University Utrecht
Debey Institute, Molecular Biophysics

Detect clustering in cells using Homo-FRET

Author:
Kyo Beyeler
3479196

Supervisors:
Dr. Gerhard A. Blab
Dr. Arnout Imhof

October 22, 2015

Abstract

In this research homo-FRET was used to detect clustering of EGFR in cells. To test this method reference constructs were made containing (2x)FKBP. A wide field microscope, polarizing beam-splitter and two cameras where used to detect polarizations. As a labeling fluorophore mCherry was used instead of the common mGFP because the autofluorescence is believed to be lower in the red spectrum. The best result obtained so far is a drop of 7.8 $\pm$ 6.8% in the case of FKBP and 11.3 $\pm$ 6.1% in the case of 2xFKBP upon adding AP. EGFR samples where also stimulated with EGF but a significant drop was not observed yet. The main problem encountered was registrating the two images.

Contents

1 Introduction

A useful trick when looking at biological samples is to attach a fluorophore to a protein or structure of interest. This way a particular protein can for example be followed in time. The resolution of this technique is however limited by the wavelength of light. Since electron microscopy can not always be used because it may damage the biological tissue, tricks have been invented to overcome this resolution limit. One example of such a trick is förster resonance energy transfer (FRET), sometimes called fluorescence resonance energy transfer. This technique uses the fact that fluorophores can transfer their energy to a neighboring fluorophore instead of emitting it in the form of light themselves. Since fluorophores need to be in close proximity to do this, FRET can be used as a distance measurement. Furthermore, as the amount of FRET increases with cluster size (since, in this case, multiple energy transfers can take place before light is emitted) it is possible to determine the cluster sizes with homo-FRET measurements. In a study by Arjen N. Bader et al. a distinction between monomers, dimers and oligomers could indeed be made.[1]

A protein which can be studied with this technique is the epidermal growth factor receptor (EGFR). EGFR is a well-studied receptor since many cancer types show overexpression or deregulation of EGFR, which makes it an attractive anticancer drug target. EGFR is usually located on the membrane in the form of monomers. When stimulated with EGF, EGFR starts clustering (form either dimers or oligomers) and internalization takes place. Internalization means the EGFR cluster is drawn inside the cell. After internalization the cluster starts signaling, thereby starting new biological reactions. The idea was to modify (parts of) the EGFR and see if clustering still took place. This way it might be possible to determine which parts of EGFR are important for this clustering behaviour, thus gaining information about the biological reaction associated with this clustering behaviour.

To summarize, our research question was: can we detect clustering of EFGR with Homo-FRET? And if so, what is important for the clustering behavior of EGFR? To answer these questions I worked with a master's student from the biology department called Reinier Damman. More details on why this research is important for biologists can be found in his thesis.[2] This first half year of this project I also worked together with a bachelor's student (from the physics department) called Margriet Oomen.[3]
This research was started by Arjen N. Bader et al., who used a confocal microscope in order to detect clustering with homo-FRET[1], but since confocal microscopy is very time consuming, this research was then continued on a wide-field microscope by Nivard Kagie[4] and Kiefer van Teutem[5]. The set-up and (MatLab) codes for data analysis they designed were the starting point of this research.
Nivard used mGFP as a labeling fluorophore, which fluoresces in the green spectrum, but the autofluorescence (a kind of background) is believed to be lower in the red spectrum compared to the green spectrum. Since autofluorescence turned out be a large factor in the uncertainty, a switch to mCherry was made, which fluoresces in the red spectrum.

2 Theory

2.1 Fluorophores and fluorescent microscopy

A fluorophore is a chemical that fluoresces (re-emits light) upon excitation. There exist many types of fluorophores, emitting a variety of colors. In fluorescent microscopy fluorophores are often attached to a (biological) tissue or protein to visualize a structure of interest. A disadvantage of fluorescent microscopy is that your resolution is limited by the wavelength of light. Many techniques have been developed to overcome this limit. One of these techniques was used in this research, and is called FRET.

2.2 FRET

Förster Resonance Energy Transfer (FRET) is the non-radiative transfer of excitation energy between fluorophores. Instead of emitting light a fluorophore transfers its excitation energy to a different (type of) fluorophore within close proximity, as is illustrated in figure 1. This is done via a dipole-dipole electrostatic interaction. The efficiency of this energy transfer is strongly dependent on the distance between fluorophores ($\propto 1/R^6$) so FRET can be used as a distance measurement. The distances are in the range of 1-10nm, way beyond the resolution limit.
Homo-FRET is FRET between two fluorophores of the same type. It can only be measured by measuring a (de)polarization. The dipoles of a sample of fluorophores are randomly orientated. When exposed to polarized light, only fluorophores with a dipole oriented in the same direction as the polarization will be excited. If the type of fluorophore is chosen such that it does not rotate within its lifetime, the light emitted from the fluorophore will have the same polarization as the incoming light. For energy transfer the dipoles are not necessarily parallel so when (Homo-)FRET takes place the polarization direction can change. Therefore Homo-FRET can be measured by a depolarization.[6]
In this research Homo-FRET is used to measure clustering. If a depolarization (with respect to a reference sample) is measured, Homo-FRET took place and the fluorophores have to be within a range of less than 10nm from each other so it can be said they have clustered. If no depolarization, and therefore no Homo-FRET, is measured it is assumed the fluorophores have not clustered.

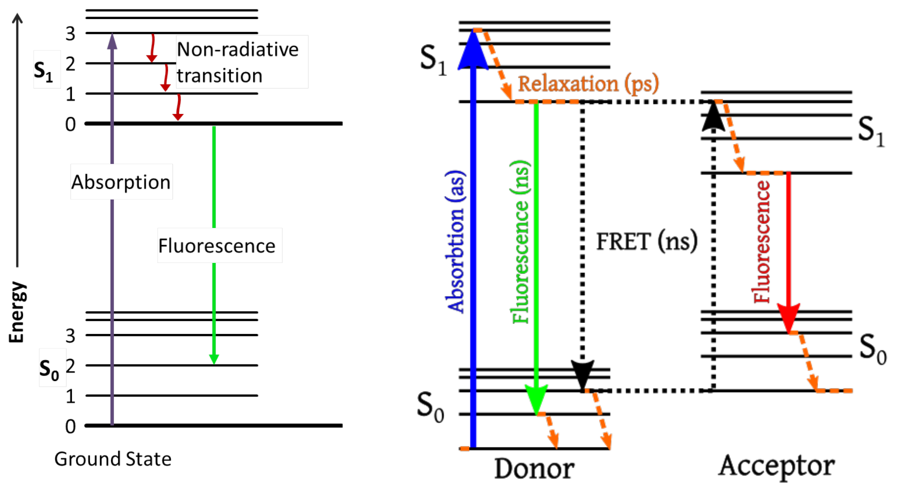


Figure 1: Jablonski Diagram of normal fluorescence (left[1]) and FRET (right[2]). In normal fluorescence a fluorophore gets excited (adsorbes a fluorophore), relaxes to the lowest excitation level and drops back to the groundstate, emitting a photon in the last step. With FRET, instead of emitting a photon the fluorophore (donor) transfers its energy to a neighbouring flourophore (acceptor). As this second fluorophore drops back to the groundstate it emits a photon.

[1]"Jablonski Diagram of Fluorescence Only" by Jacobkhed -Own work. Licensed under CC0 via Wikimedia Commons-
[2]"FRET Jabolinski Diagram" by Alex M Mooney -Own work. Licensed under CC BY-SA 3.0 via Wikimedia Commons-

2.3 Anisotropy

To easily compare different polarizations, usually the anisotropy is calculated. The anisotropy, r, of the light emitted by a fluorophore is given by equation 1.

$$r = \frac{(I_{\parallel} - A_{\parallel}) - G\,(I_{\perp} - A_{\perp})}{(I_{\parallel} - A_{\parallel}) - 2G(I_{\perp} - A_{\perp})} = \frac{\text{Difference in intensities}}{\text{Total intensity}} \tag{1}$$

Where $I_{\parallel}$ represents the parallel intensity and $I_{\perp}$ the perpendicular intensity, with respect to the polarization of the excitation beam and $A_{\parallel}$ and $A_{\perp}$ represent the background noise in these two different intensities. G stands for the G-factor, which is a correction factor introduced because a set-up will probably not treat both polarizations equally. In an ideal set-up G would be one. The two is there because of normalization, since there are two directions perpendicular to the excitation beam. These two directions are assumed to contribute equally because of the symmetry of the system so the denominator is just the total intensity.[5] The numerator is just the difference in intensity of the different orientations of the emission.
As $I_{\perp}$ increases, r decreases so a depolarization can be measured by a drop in anisotropy. The more fluorophores are present in a cluster, the more energy transfers can then take place before light is emitted. So the drop in anisotropy increases with cluster size.

Aside from energy transfer, there are other variables that influence the anisotropy which have to be taken care of and, if necessary, corrected for. Besides the factors discussed in this section, (parts of) the set up also influences the polarization. This will be elaborated on in section 3.

Non-parallel excitation/emission axes

It was stated before that only fluorophores with a dipole moment parallel to the orientation of the excitation light will be excited, but that's not entirely correct. Fluorophores with a slightly different orientation will still be excited, which decreases the anisotropy. The maximum possible anisotropy (so the anisotropy when no homo-FRET takes place) for a sample of randomly orientated fluorophores can be calculated to be 0.4, assuming the excitation and emission axes to be parallel.[4] Since parts of a set-up usually have a lowering affect on the anisotropy, the maximum possible anisotropy with a specific set-up will probably lower than this theoretical maximum.

Photobleaching

A common problem when working with fluorophores is bleaching. Bleaching is the damaging of a fluorophore due to too much or too long exposure to light, after which it will not fluoresce anymore. Bleaching reduces the number of 'active' fluorophores within a cluster and will therefore lead to less energy transfer, hence an increase in anisotropy.

2.4 Accuracy of the anisotropy

In this thesis images of cells where made and every image consists of pixels. The anisotropy was first calculated for every pixel using formula 1, then the average value for every cell and every group of cells was calculated, where a group consists of all cells who underwent the same treatment (so either non-stimulated or stimulated with AP or EGF). The standard deviation of a pixel value can be calculated with propagation of uncertainty using formula 2.

$$\sigma_r = \sqrt{\left(\frac{\partial r}{\partial I_{\parallel}}\sigma_{I_{\parallel}}\right)^2 + \left(\frac{\partial r}{\partial I_{\perp}}\sigma_{I_{\perp}}\right)^2 + \left(\frac{\partial r}{\partial G}\sigma_{G}\right)^2 + \left(\frac{\partial r}{\partial A_{\parallel}}\sigma_{A_{\parallel}}\right)^2 + \left(\frac{\partial r}{\partial A_{\perp}}\sigma_{A_{\perp}}\right)^2} \tag{2}$$

$I_{\parallel}$, $I_{\perp}$, $A_{\parallel}$, $A_{\perp}$ and G are defined as before and σ represents the standard deviation of every quantity. This σ_r is used as a weight to calculate a weighted average of the anisotropy of a cell. The standard deviation of an average cell value is calculated by looking at the variations within a cell. To be able to easily compare different treatments, an average value of all cells with the same treatment was calculated. The standard deviation was calculated using the variation between *average* cell values. It is the 'standard deviation of the mean values', or SEM (standard error of mean) as it is usually called.[7] The main contributions to the uncertainty are a fundamental uncertainty arising from photon statistics and a background called autofluorescence.

Autofluorescence

Some parts of the cell can fluoresce without the addition of a fluorophore. This is called autofluorescence, it will appear as background noise in an experiment. The autofluorescence varies between cells and may vary in different parts of the cell, e.g. different for nucleus or membrane. The autofluorescence might also depend on the excitation wavelength, i.e. it is believed to be lower in the red spectrum compared to the green spectrum which is the reason mCherry was chosen instead of mGFP, which was used in former experiments. The autofluorescence needs to be measured carefully and subtracted together with other background noise before calculating the anisotropy.

Photon statistics

The number of photons hitting the camera chip obey Poisson statistics. The uncertainty in the intensity is therefore equal to the square root of the number of photons. A camera however will not directly give the number of photons, their will be a gain factor in between. Say n is the number of photons that hit a pixel at a certain time and α is the gain factor. The number of counts given by the software is equal to αn, leading to a standard deviation in the intensity of $\sigma_I = \alpha\sqrt{n} = \sqrt{\alpha}\sqrt{\text{number of counts}}$. The gain factor can be determined by looking at the average and variance of a homogeneous image. Kiefer determined the gain factor to be: $\sqrt{\alpha} = 1.3$.[5]

3 Methods and materials

A large part of the project was characterizing the set-up. How do different parts of the set-up affect the polarization and what does that mean for the final result.
In short, measured are two images of a cell which show the intensity of the two different polarizations. One image contains the intensity of the parallel light, one the intensity of the perpendicular light (with respect to the excitation light). These two images are obtained by two cameras and analysed with software, to obtain the anisotropy with its uncertainty per pixel. Also an average value for each cell is calculated and given with its SEM. Untreated and stimulated cells can then be compared.
For analysis not only the two images of a cell are needed but also a few reference/calibration measurements. Which measurements, how they are done and how they are analysed will be explained in this section.

3.1 Experimental Setup

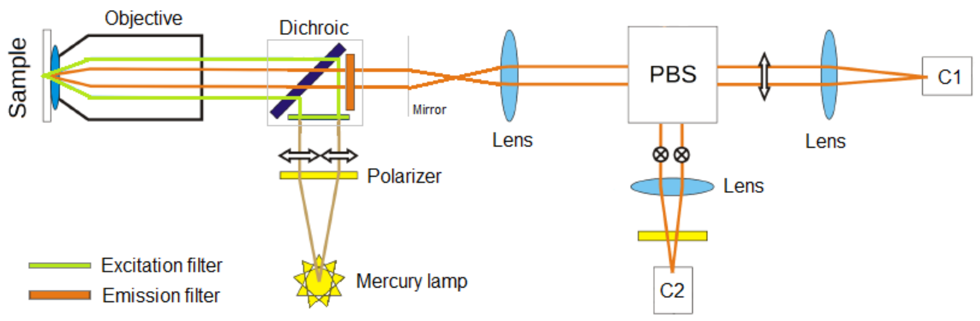


Figure 2: A schematic representation of the set-up used for this research. The excitation light is shown in green, the light emmitted from the sample in orange. PBS stands for polarizing beam splitter and C1 and C2 represent two different cameras. Figure adapted from Nivards thesis[4].

Figure 2 shows a schematic, simplified version the set-up. Excitation light is emitted by a mercury lamp, after which it goes through a polarizer, an excitation filter and dichroic mirror before reaching the sample. The light emitted from the sample gets reflected by the dichroic mirror, goes through an excitation filter into a polarizing beam splitter. Finally the intensities of the different polarizations of the emitted light are detected by two cameras. The beam splitter has a much better extinction ratio in the parallel direction then in the perpendicular direction, which means part of the parallel beam leaks through to the perpendicular beam. To filter this out, an additional polarizer is placed before the camera recording the perpendicular beam. The cameras are connected via a trigger, to make sure the images are recorded at the same time. The cameras can record an image up to 2560x2160 pixels but saving such a large image cost time and memory while cells usually aren't that big. A region of interest is therefore chosen on the chip (of 500x500 pixels) and only this region is saved.
The excitation filter, dichroic mirror and emission filter described above are all part of the dichroic cube, drawn as a box in figure 2. Two different dichroic cubes where used for this research, both with a different set of filters, one of which was specialized for mCherry and will be called 'mCherry dichroic'. The other dichroic was only used to maesure yellow-green beads and will be called 'YGbead dichroic'. The details of their filters are given below.

	excitation filter	emission filter	dichroic mirror
YGbead dichroic	460-500	516-556	505
mCherry dichroic	540-580	590-670	585

The objective, or specifically its numerical aperture (NA), influences the anisotropy. Using a high NA (>0.75) decreases the anisotropy because light can come in under a larger variety of angles.[8] Therefore an objective with NA=0.6 was used during this research.

3.2 Sample details

3.2.1 Sample 1: G-factor

In our experiment the parallel and perpendicular intensities where measured by different cameras. Both cameras are probably not equally sensitive. Furthermore, there is already an a-symmetry because of the extra polarizer and other parts of the setup, e.g. lenses or mirrors, might also treat the different polarizations differently. As mentioned in section 2, this preference can be corrected for with the G-factor. To measure how the set-up treats polarizations, a sample which emits completely depolarized light is needed, for example a dye with a lifetime longer than its rotational diffusion. In this research Rhodamine in an aqueous solution was chosen. Since the dye can rotate freely it will completely depolarize the emission, especially since the polarization filter was removed during this measurement so the dye gets excited by a depolarized beam.
Since an anisotropy of zero is expected for this sample, the G-factor can simply be calculated with: $G = \frac{I_{||}}{I_{\perp}}$. Since the value of G might be dependent on position, it needs to be calculated for every pixel separately. The G-factor is therefore an *image*, not a number.
Glass and cover slides were coated to prevent the dye from sticking to the glass and measurements were taken from the middle of the sample (height measurements were done to test how sensitive the measurement is to the distance from the edges to the focus position). For every measurement the exposure time was adapted to obtain an average number of counts (in the parallel image) around 30.000. With a concentration of $5\mu M$, this was typically around half a second. A background was measured with a sample prepared exactly the same as the rhodamine sample but without adding the rhodamine itself. The background value was measured under the same conditions as the 'real' sample, e.g. same exposure time, and its value was typically between 100 and 150. The background was subtracted per pixel before calculating the G-factor. Standard deviation was also calculated per pixel, using photon statistics and propagation of uncertainty. Multiple images were taken and averaged to average out noise fluctuations and therefore reduce the standard deviation. The number of images was chosen such that bleaching was kept below 5%.

3.2.2 Sample 2: Background

The background consists of several contributions: the camera background, which is independent of exposure time, background noise introduced by the environment, which was minimized by turning off the light and closing the door, etc., and the autofluorescence. Of these, the autofluorescence is the largest, and most unpredictable factor.
The camera background is quite constant and can be measured with high accuracy, by recording a lot of images and averaging these. Variations in background due to the environment are negligible compared to (variations in) autofluorescence.
Autofluorescence can vary a lot between cells. Kiefer proposed a method to distinguish between different regions in a cell but this has been omitted since most cells were too dim to distinguish these regions. Some samples even contained cells who were to dim to see with the naked eye (through the microscope) so the top light was used to find these cells. Autofluorescence cells were measured with exposure times of 1, 2, 3 and 10 seconds to test if the autofluorescence scales linearly. All other settings were kept the same as in the 'normal' cell measurements. Cells for these autofluorescence measurements were prepared by Reinier Damman. The details of the sample preparation can be found in his thesis.[2]

3.2.3 Sample 3: Maximum Anisotropy

It was tried to measure the maximum anisotropy possible for this set-up with a sample of fixed mCherry. When the measurements of cells are scaled to the maximum anisotropy the difference in anisotropy should be more clearly visible. However, the values found for the maximum anisotropy were much lower then expected and because no explanation could be found, it was decided to omit this. A more detailed explanation can be found in Margriets thesis.[3]

3.2.4 Sample 4: Biological samples

The fluorophore used for this study is mCherry of which the structure and spectrum can be seen in figure 3. The peak excitation wavelength is 587nm, the peak emission wavelength is 610nm.

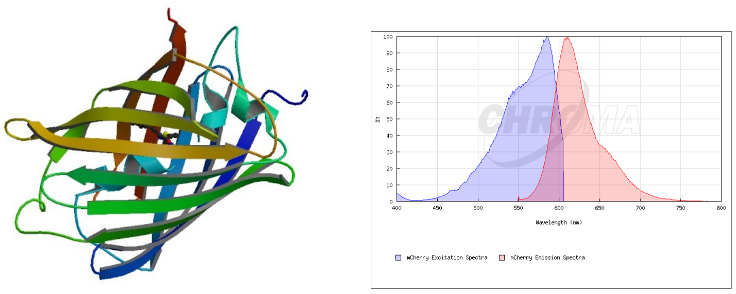
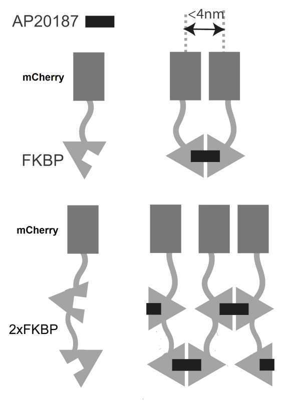
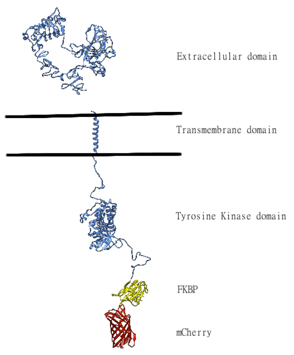


Figure 3: Properties of mCherry. The figure on the left shows the structure of mCherry. Figure has been taken from RCSB Protein Data Bank. The right image is from Chroma Spectra Viewer and shows the spectrum of mCherry.

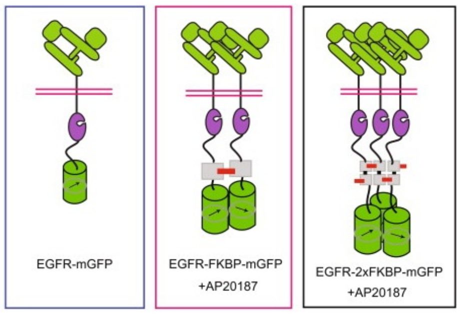
To test our method, reference constructs where made containing FKBP or 2xFKBP. FKBP is a monomer which forms dimers upon addition of AP as is shown in figure 4a. When using FKBP dimers are formed, when using 2xFKBP ogliomers are formed. Since (2x)FKBP always clusters upon adding AP this is a method to verify if this clustering behaviour can be detected by the set-up.
Figure 4b shows the protein used in this research, it is a fused protein of EGFR with FKBP and mCherry. This fused protein will cluster when stimulated with AP but it is also known that EGFR clusters upon stimulation with EGF. So our protein can cluster via FKBP upon adding AP or via EGFR upon adding EGF. Non-stimulated samples have been compared with samples stimulated with either AP or EGF, it is expected that the anisotropy is lower for the stimulated samples when compared to the non-stimulated samples, what's more this drop in anisotropy should be larger when using 2xFKBP instead of FKBP. These proteins and cells were prepaired by Reinier Damman and the details of the sample preparation can be found in his thesis.[2]



(a) Schematic representation for clustering of (2x)FKBP by stimmulation with AP.



(b) Structure of the fused protein. EGRF is shown in blue, FKBP in yellow and mCherry in red. Figure made by Reinier Damman.



(c) Schematic representation for clustering of a fused protein. AP is shown in red. For this research the mGFP was replaced by mCherry. Figure from [6].

Figure 4: The EGF Receptor and how it clusters through fusion with FKBP.

The cells varied a lot in brightness which is important since the uncertainty partially depends on photon statistics. The more counts, the less the uncertainty. The number of counts can be increased by increasing the exposure time, however when measuring too long, bleaching might come into play. The exposure time needs to be chosen with care. Usually is was chosen to be 1, 2 or 3 seconds and too dim or too bright cells where avoided (too bright means they already saturated the camera with an exposure time of 1s). This way the number of counts was on average between 10.000 and 35.000 counts for the parallel image.

3.3 Data analysis

Data analysis was done in: ImageJ, MatLab and Mathematica. Nivard[4] started designing the data analysis and Kiefer[5] further optimised it. The MatLab scripts Kiefer wrote were the starting point of this research. All scripts can be found in appendix B and C.

Registration

The first step in data analysis was overlaying the two images since, whenever measuring with two cameras, the two images are likely to be misaligned. A process called registration is needed for overlaying the images. Tuning the alignment turned out to be very important. When the two images are slightly shifted relative to each other, this can have a significant effect on the anisotropy. Finding the proper registration turned out to be one of the major challenges of this project.
One approach was to use beads as fiducial markers. This was done in a few steps.
Step 1: At first the position of the beads was determined. This was done using a threshold at a certain percentage of the mean and maximum value to filter out most noise and the built-in functions 'regionprops' and 'WeightedCentroid'. This function uses the intensity to determine the positions. For most samples this works fine but when the signal to noise ratio is very low this does not work. If this is the case it is better to determine the positions using a plugin of ImageJ called 'GDSC SMLM'. It has a fit function which uses a gaussian fit to determine the position.
Step 2: After the positions are determined in both the parallel and perpendicular image, they have to be sorted since the beads are not necessarily found in both images or found in the same order. This is done by setting a maximum distance. For every bead in the parallel image, the bead in the perpendicular image nearest to it is found, if the positions are within the maximum distance from each other, they are assumed to belong to the same bead. This maximum distance was usually set at 10 pixels. This does mean that two beads cannot be too close to each other. What's more, the larger the shift between the two images, the further away from each other two beads should be. Pay attention to this!
Step 3: After all positions are determined and matched the transformation needed to overlay them is determined. For this an 'affine' transformation was used, which means the images can be translated, scaled, sheared or rotated (the last one being a combination of the former three). This is demonstrated in figure 5. The image is transformed by a matrix which has the form:

$$\begin{pmatrix} a & b & 0 \\ c & d & 0 \\ x & y & 1 \end{pmatrix}$$

The image is then transformed using [x y 1] = [u v 1]*T, where T is the transformation matrix shown above, u and v the original position and x and y the position after transformation.
Alternative method: Since there were some difficulties with this method the registration was also done in another way. The transformation matrix could also be determined using the images of cells themselves. MatLab then determines the matrix by comparing relative intensities and shifting the images until it finds the best fit. This is a built-in function of MatLab which used the same affine transformation.

G-factor

The next step was to determine the G-factor. First the multiple images of rhodamine and its background were averaged. This was done in ImageJ using 'Z-projection'. Usually a series of 10 images was taken and this function creates an image where every pixel has the average value of all values this pixel has in the 10 different images. This way random fluctuations are averaged out. The background images are then subtracted from the Rhodamine images (per pixel, in MatLab) before calculating G using $G = I_{\parallel}/I_{\perp}$. The uncertainty in the intensity is calculated using photon statistics and with this, the standard deviation in G is calculated using photon statistics and propagation of uncertainty, using the differential method. Again, everything is calculated per pixel, since G might be position dependent

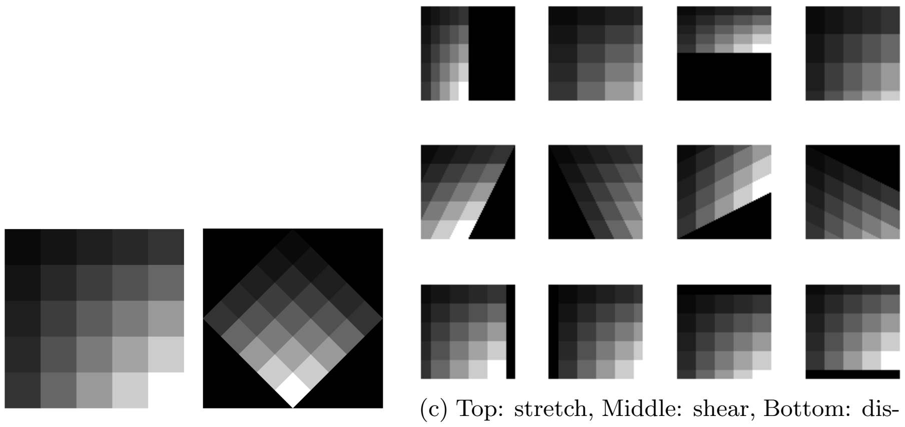

(a) Original (b) Rotation (c) Top: stretch, Middle: shear, Bottom: displacement

Figure 5: The transformations possible with an affine tansformation. Additional pixels, not in the original image, are given the value zero.

Background

Cells were measured with different exposure times, but the background is also (partially) dependent on this exposure time. The way this was solved was by keeping track of the exposure time by putting it in the file name. This way when MatLab reads in the images or cells, it also links a exposure time to each cell. Now the camera background, which is independent of exposure time, was subtracted first and after this, the background value of 1s times the exposure time was subtracted. This way the right background value was subtracted, assuming the background is linear (which was tested, luckily this was the case, as will be shown in section 4).
Practically all background dependent on exposure time was autofluorescence. Since the autofluorescence cells are dim, it was difficult to distinguish them. The signal to noise ration was too low to reliably let software distinguish the cells from the background noise so therefore this was done by hand, with the freehand selection tool in ImageJ. With this selection a mask was created. In MatLab this mask was converted to a binary image and multiplied with the original image to extract the autofluorescence values within the cell. The average and standard deviation of each cell was then calculated.

Cells

Now that the right corrections have been applied, only one step is left aside from calculating the anisotropy: determine what part of the image is 'cell' and which part is 'background'. This was done in two different ways. A simple method is setting a threshold on intensity. A threshold of 10.000 counts (in the parallel image) was found to work fine. Another method was based on uncertainty. First the anisotropy and standard deviation were calculated for every pixel in the image. Then all pixels with a standard deviation below a certain threshold were set to zero, typically this threshold was 0.02. (Since the standard deviation mainly depends on intensity, both results do not deviate a lot.) In both methods pixels with a value of 53000 were set to zero since that is the saturation value of the camera. Both methods give an image with the anisotropy values for every pixel within the cell and an image of standard deviations per pixel (all values outside the cell are set to zero).
These methods are illustrated in figure 6. The top images show there is almost no difference between the two methods. This relation between intensity and standard deviation can be seen in the bottom images: the standard deviation per pixel increases at the edges of the cell. This is because the intensity drops at the edges. Also the cell with a higher uncertainty had a lower intensity than the other cells. In the middle figure it can also be seen that the background has a much higher uncertainty than the pixels belonging to a cell.
After the anisotropy was determined per pixel, the average value of a cell and each group of cells were calculated with their SEM, as already described in section 2. A detailed protocol on how to name the images and how to use the scripts in MatLab is given in appendix A.

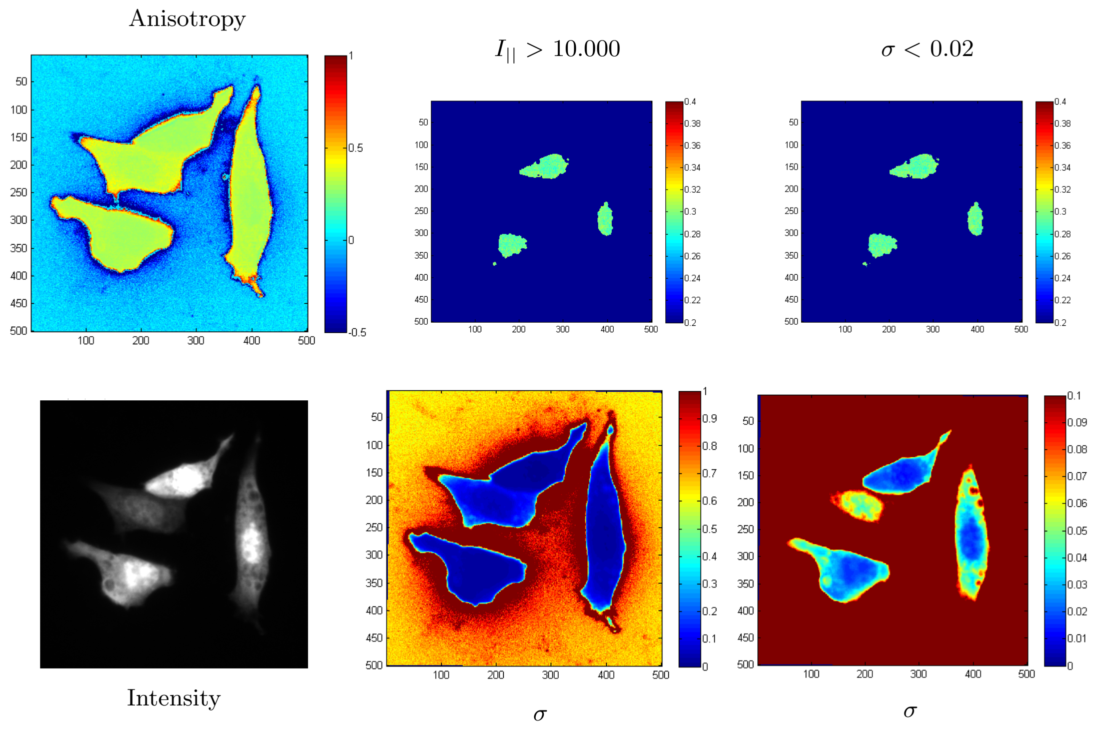


Figure 6: The top three images show all the anisotropy of a few cells. The most left image shows the anisotropy for every pixel, without any threshold. On the middle and right images a threshold has been applied using intensity and standard deviation respectively. As can be seen, the images with different thresholds are almost identical. Bottom: One image showing the intensity (contrast has been enhanced for clarity) and two images showing the standard deviation of every pixel. The two images showing the standard deviation only differ in their range on the scalebar. These images show, very clearly, the correlation between intensity and standard deviation.

4 Results

4.1 Calibrating & Investigating the setup

4.1.1 Registration

The first step in the process of calculating the anisotropy, was to determine the registration. As is illustrated in figure 7, a fault in registration can have a significant effect on the anisotropy and the effect is even larger in the cells with clusters. It does seem logical that a homogeneous images is less sensitive to registration then an image with sharp difference. As was said in section 3, there are two ways to determine the registration, either from a sample of beads or from the cell images themselves. These two methods did not work equally well. When aligning the cell images with the transformation matrix determined with beads, a small misalignment was still seen, as is shown in figure 7. Registrating with the cell images themselves works but it is quit slow (∼ a minute instead of a second). So it was investigated why this method with beads works less well.

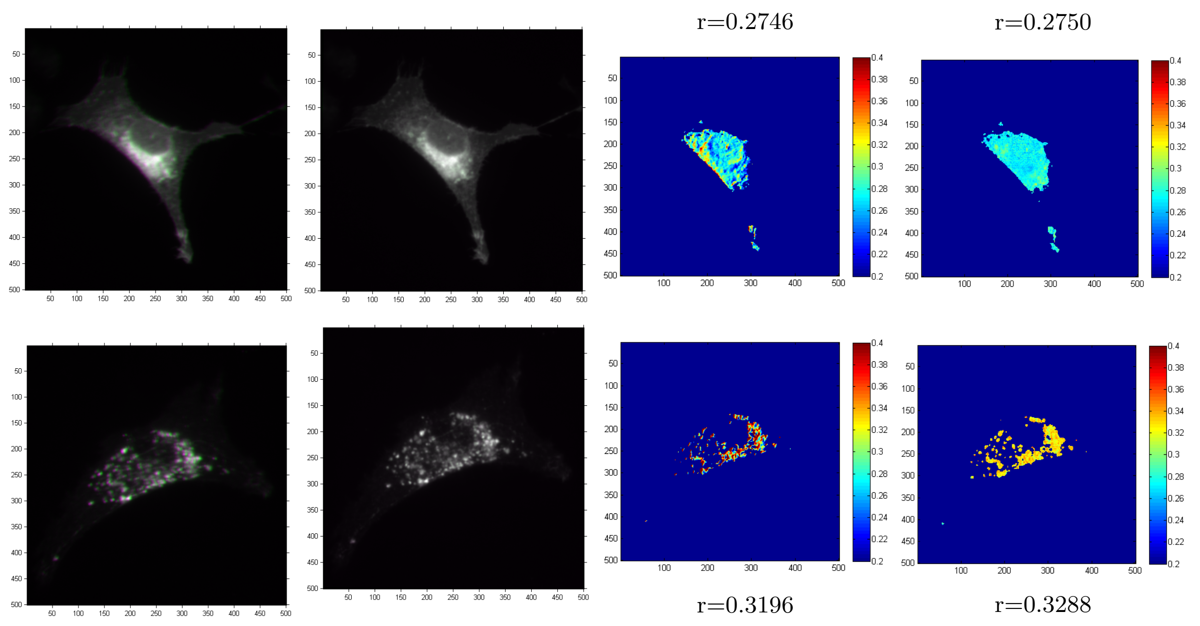


Figure 7: Effect of registration. The leftmost images show the overlay of the two intensity images, without any correction. Green and pink colors indicate a mismatch, white images indicate a correct overlay. The images second from the left show the intensity images after proper registration. The images second from right show the anisotropy of the cell when the transformation matrix calculated from beads is used to overlay the images. The rightmost images show the anisotropy of the cell when the transformation matrix calculated from relative intensities of the cell images themselves are used to overlay the images.

One idea was that it might be the dichroic, because the registration was first done with yellow-green beads which need a different dichroic than mCherry. So a sample of red beads was prepared but this did not solve the problem.
Also, it was checked if the focus position of the camera was the same for both cameras, by making a z-stack of a sample of beads and using a gaussian fit to determine the amplitude and full width half maximum (FWHM) of each bead in every image. The focus position is then found by finding a maximum amplitude or minimum FWHM. The results are shown in figure 8. The first two images show the fit of one bead. The rightmost image was obtained by averaging fits of 54 beads. Numbers corresponding to this graph are given in table 1. As can be seen, the focus position is slightly different for both cameras but the difference is very small, especially considering it is placed and focused by hand.
Next, it was checked if the registration varied over the chip. The registration seemed to consist mainly of a rotation, so it was tried to find the center of this rotation. This was done by making an image of the full chip and plotting the distances of the beads in the parallel and perpendicular image. If the

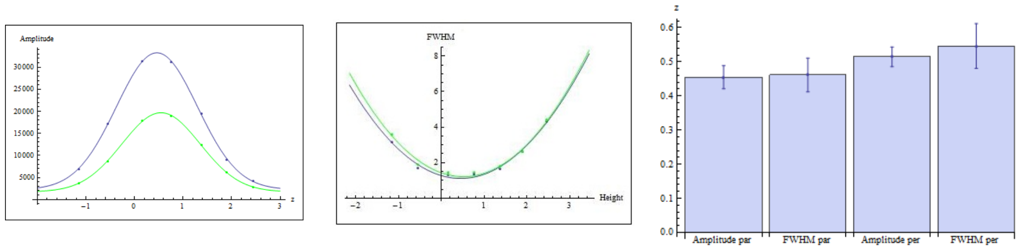


Figure 8: Determine the z-position of the focus: A Gaussian fit was used to find the FWHM (middle) and amplitude (left) as a function of z. The left and middle image show a fit of one bead. Of 54 beads the maximum amplitude and minimal FWHM was determined. The average z value is shown in the graph on the right.

	Mean from amplitude	σ
Par	0.454996 μm	0.0337102 μm
Per	0.51473 μm	0.0290469 μm
	Mean from FWHM	σ
Par	0.461544 μm	0.0496222 μm
Per	0.546309 μm	0.0657887 μm

Table 1: z-position of the focus. Numbers have been obtained by fitting a Gaussian through an intensity image of a bead and extracting a FWHM and an amplitude. By calculating at which height they had their minimum or maximum respectively, the focus position in z of this particular bead was found. This was done for 54 beads. The average values are given here.

registration truly consists of a rotation, their should be a point where the distance is (very close to) zero and from this point the distance should increase going away from this point radially. The results are shown in figure 9.
As can be seen these image seem to indicate such a center of rotation can indeed be found. A new region of interest was determined with this center of rotation in the middle. Images of beads where taken with this new region of interest, of which an example is shown in figure 10 (left image). From the bead images it was clear the registration was indeed a rotation. To illustrate this a bit more clearly, a homogeneous image was transformed with the same registration matrix needed to transform the beads of figure 10.
Measurements of red beads and cells have been done with this new region of interest to test if the registration works better and, unfortunately the transformation matrix calculated from the position of the beads is still slightly different from the transformation matrix of the cells. The registration of cells is about twice as fast though, since the starting point is better.
Finally, it was tested how the software transforms the image. What value does MatLab take for the 'new' pixel after transforming the image? This was tested with a simple line image, which was rotated or stretched as is illustrated in figure 11. The transformed image contains many values in between 1 and 3, indicating MatLab calculates a weighted average for the new pixel values.

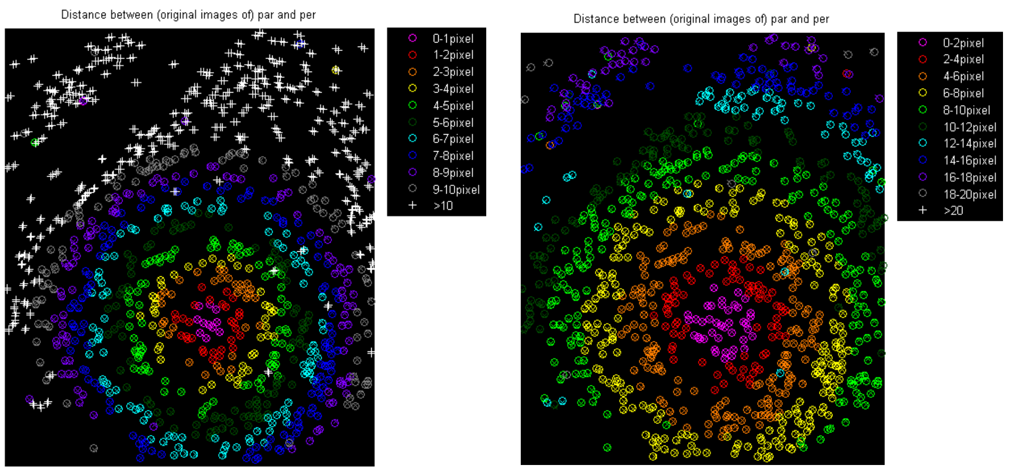


Figure 9: Plotted in each image are the distances between the positions of a bead in the parallel and perpendicular image, in order to find the center of the rotation needed to overlay two images. As can be seen their is an area where the distance is almost zero and going radially outward from this area, the distances increase, indicating indeed a rotation around this area.

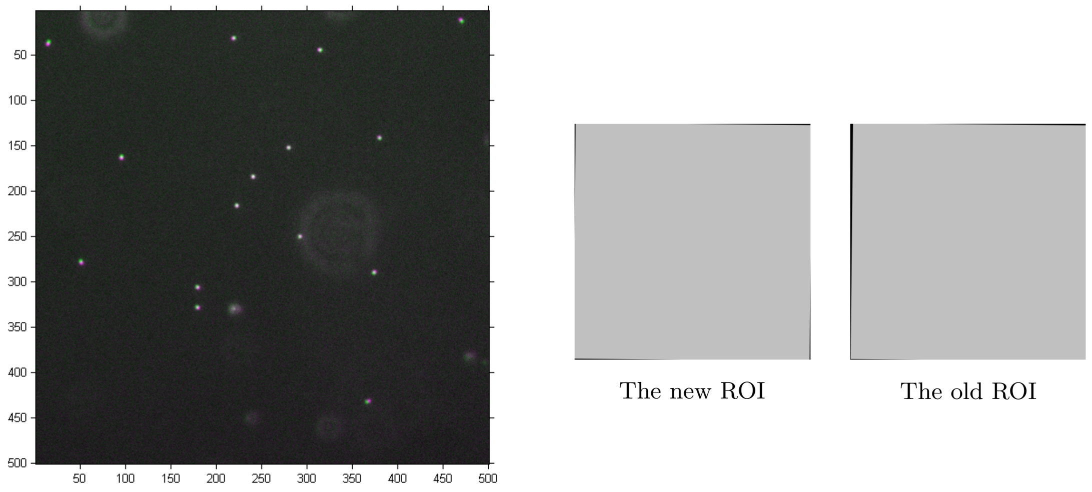


Figure 10: The shift needed to overlay the images with the old and new ROI. The left image shows the two intensity images of a sample of beads, overlayed without any correction, taken with the new region of interest. The middle and right image show a homogeneous image which is shifted with the transformation matrix of the new and old region of interest respectively.

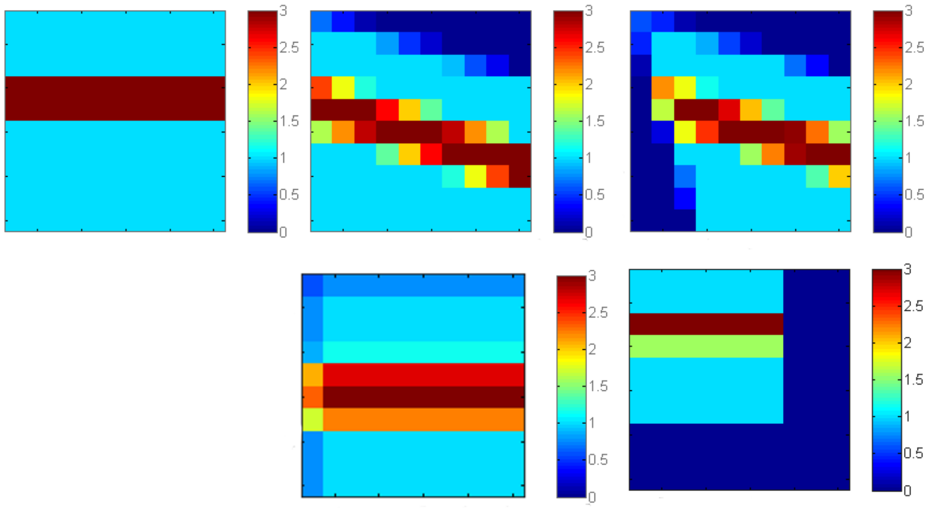


Figure 11: Testing how Matlab determines the values of the new, transformed image. Image on the left show the original images. Top middle: shear in one direction $\left(\begin{smallmatrix}1 & 0.3 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{smallmatrix}\right)$, Top right: shear in two directions $\left(\begin{smallmatrix}1 & 0.3 & 0\\ 0.3 & 1 & 0\\ 0 & 0 & 1\end{smallmatrix}\right)$, Bottom middle: stretch in x and y, $\left(\begin{smallmatrix}1.3 & 0 & 0\\ 0 & 1.3 & 0\\ 0 & 0 & 1\end{smallmatrix}\right)$, Bottom right: shrink (negative stretch) in x and y, $\left(\begin{smallmatrix}0.7 & 0 & 0\\ 0.7 & 1 & 0\\ 0 & 0 & 1\end{smallmatrix}\right)$. As can be seen MatLab uses weighted averages to calculate a new pixel value and extra pixel to fill up the remaining space are given the value zero.

4.1.2 G-factor

A typical image of the G-factor can be found in figure 12. It is clear that the value of the G-factor varies over the ROI and therefore needs to be calculated per pixel. The ROI in the middle of the chip, had an average value of 1.379, the optimized ROI an average value of 1.220. The average value of the standard deviation per pixel is 0.004, for both ROI's, taking an average of 20 or 10 images respectively. Since the image is quite homogeneous (no sharp edges/sudden changes) a small change in the registration matrix does not have a significant influence on the G-factor.

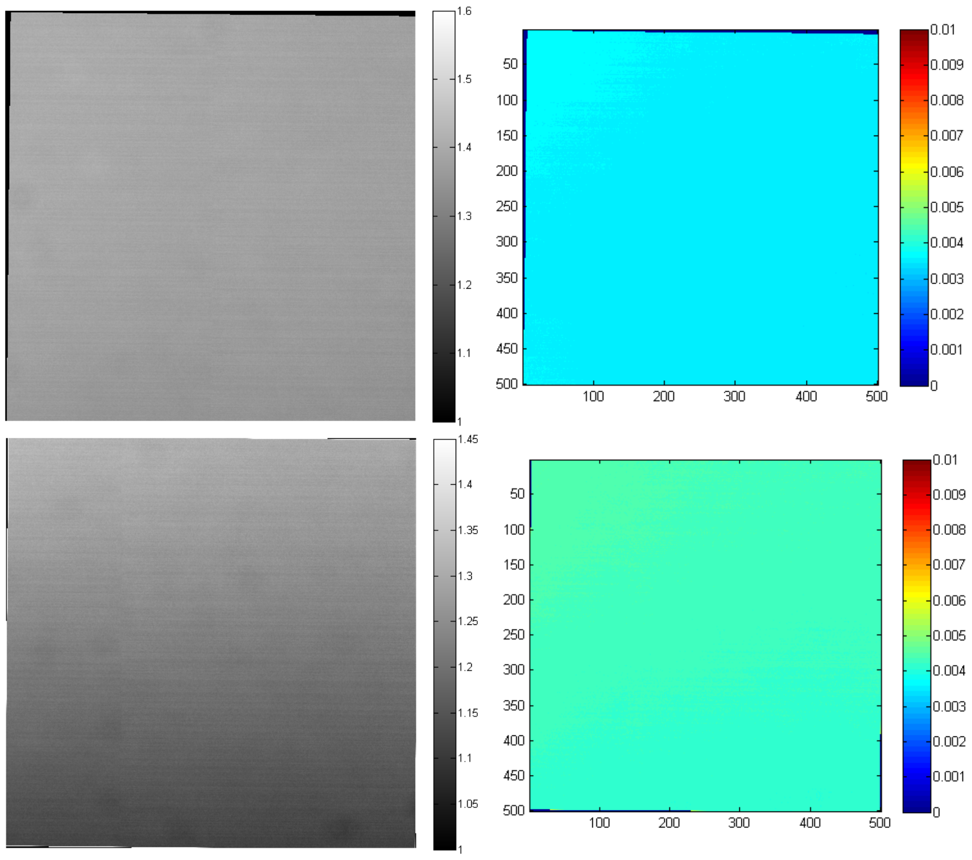


Figure 12: A typical example of the G-factor with its uncertainty per pixel. Left images show the G-factor, right images the standard deviation. Top images are taken from the old ROI (middle of the chip), the bottom images from the new ROI (center of rotation).

Because measurements on different days varied a lot more than what was expected, a few sensitivity test where done. First, it was checked how important it is to measure the Rhodamine (far) away from the edges, the G-factor was measure at different heights. Focusing was done with the top light on the cover glass, there the z-value was set to zero. The results are given in table 2. The average value of the G-factor and σ are given. σ is the average value of the uncertainty per pixel, so this does not take into account the variance over the image. The sample was about $100\mu m$ thick. As you can see the average values of the G-factor do not differ a lot and are well within each others uncertainty.
Secondly, different camera settings or switches on the microscope where tested, how much do they influence the G-factor? The standard settings are given below. In table 3 is given which settings were changed and what the average G-factor was with that particular setting.

Standard camera settings	**Switches**
Rolling (a readout setting)	Autofocus: default is turned off
readout: 16-bit	1x magnification, could also be set on 1.5x
560MHz	

As expected, wrong camera settings give a completely different G-factor. Something a little more surprising is the fact that the autofocus also has a quite significant effect on the G-factor. This is because the autofocus uses an extra dichroic, which apparently has an effect on polarization. This value of 1.28 was exactly the same as some deviating previous measurements, we now believe this probably was because the autofocus must have been on accidentally, especially since it was easy to miss on this set-up.

height	G	σ
0.300	1.377	0.015
5.150	1.377	0.015
10.400	1.380	0.016
20.625	1.376	0.016
29.475	1.379	0.016
42.075	1.381	0.017
50.650	1.380	0.017
61.675	1.379	0.017
72.425	1.378	0.017
81.300	1.381	0.017
92.175	1.383	0.018
95.125	1.383	0.018

Table 2: 'z-stack' of the G-factor

Different from standard	G	σ
Standard settings	1.378	0.005
Both camera's Global	1.369	0.008
Camera 1: Global, Camera 2: Rolling	0.510	0.002
Camera 1: Rolling, Camera 2: Global	3.690	0.018
Camera 2: 11 bit (low noise)	17.88	0.183
Camera 2: 11 bit (high well capacity)	35.92	0.541
Camera 1: 11 bit (low noise)	0.076	0.001
Camera 1: 11 bit (high well capacity)	0.054	0.001
With top light on	1.375	0.005
1.5x	1.391	0.007
Autofocus on	1.285	0.005

Table 3: The average G-factor for different settings.

Dichroic

Another test measurement which was done to check how well our set up worked was the measurement of the transmittance of (the filters in) the dichroic. The result is shown in figure 13. In blue is shown the transmittance of the excitation filter, in red of the dichroic mirror, in green of the emission filter. This was to to check if excitation light might 'leak through' to the emission path. As can be seen the dichroic works very well.

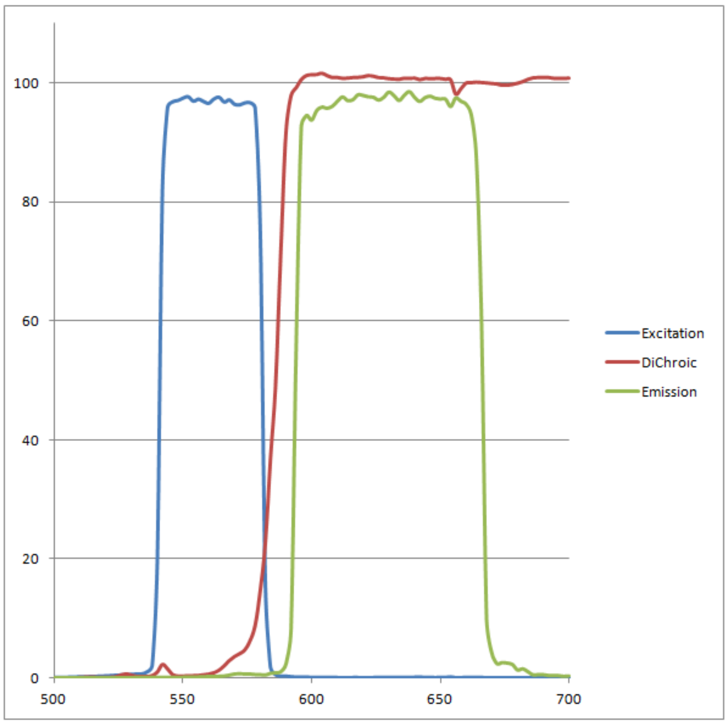


Figure 13: The transmittance of the filters in the dichroic. Numbers on the vertical axis indicate the percentage of the amount of transmittance. On the horizontal axis, the wavelength is shown.

4.1.3 Background/Autofluorescence

The autofluorescence was measured a few times, the results are shown in table 4. For each day an average value of all cells measured that day is given with its SEM (called σ in the table). In the bottom of the table, the average of all average values is given. Values given are the total background values, since the camera background was 101 counts, the true autofluorescence was 287 $\pm$ 223 and 137 $\pm$ 123 for the parallel and perpendicular intensity respectively. The values given are obtained with an exposure time of 1s. As can be seen their is a lot of variation between average values on different days. What's more the variation between cells measured on the same day can also be very different. Some measurements show almost no variation, while cells measured the first three days differ quite a lot.

Day	Par mean	σ	Per mean	σ
27/02 (10 cells)	542.177	293.655	277.491	121.720
13/03 (10 cells)	674.164	236.196	441.316	138.141
20/03 (4 cells)	518.815	289.743	279.504	102.210
17/04 (20 cells)	167.329	11.811	128.707	3.482
23/04 (20 cells)	120.084	2.880	108.703	0.912
18/09 (10 cells)	307.483	37.696	186.675	12.217
mean	388		237	
SEM (std of mean)	223		123	

Table 4: Autofluorescence values, averaged per day, without subtraction of any background (eg. camera background). As can be seen there is a lot of fluctuation in both the average values and the σ.

Linearity

Because the cells were measured with different exposure times, it was tested how linear the autofluorescence is. This was done by measuring the same cells with different exposure times. Results are given in table 5 and 6. Table 5 shows the average value of each cell and the average value of all cells with the SEM, measured with an exposure time of 1 and 10 seconds. For clarity, table 6 only shows the average value of all cells with the average standard deviation each cell had (σ). The SEM is again the standard deviation calculated using all (mean) cell values.
When looking at the difference between 1 and 10s, the differences between the average values are much smaller than the variations within the sample. When looking at the differences between 1, 2 and 3s they are comparable to the SEM, but the average values are well within each others uncertainty. Therefore it can be concluded that the autofluorescence is linear as expected.

18-09-2014	Parallel Intensity		Perpendicular Intensity	
	10s	1s	10s	1s
Average cell values	265.761	278.928	106.758	111.966
	316.603	315.637	125.861	127.189
	275.891	292.765	111.283	117.691
	262.731	284.003	106.342	114.367
	307.409	321.294	119.936	125.389
	326.675	332.493	127.317	130.106
	255.330	258.998	107.196	109.417
	313.139	331.581	124.139	131.152
	106.829	113.156	52.052	52.183
	120.439	122.700	56.731	55.059
	139.746	143.674	63.080	63.565
	144.627	147.128	64.831	65.100
	135.491	136.567	62.530	61.754
	166.918	173.676	72.934	76.059
	145.066	146.778	65.221	64.995
	183.875	197.897	77.552	82.875
	149.720	155.803	66.196	68.066
	136.020	131.432	62.840	59.850
	129.347	126.215	60.019	57.305
	152.037	150.900	68.765	68.226
Mean	201.683	208.081	85.079	87.116
Difference	6.4		2.0	
SEM (std of mean)	77.852	82.260	27.073	29.524

Table 5: Autofluorescence values with different exposure times. Of these values the camera background has been subtracted first, after which they were divided by their exposure time. As can be seen, the difference is much smaller than the SEM.

23-04-2014	Exposure time 1s		Exposure time 2s		Exposure time 3s	
	$(I_{\parallel} - Cbg)$	$\sigma_{I_{\parallel}}$	$(I_{\parallel} - Cbg)/2$	$\sigma_{I_{\parallel}}$	$(I_{\parallel} - Cbg)/3$	$\sigma_{I_{\parallel}}$
mean (20 cells)	18.708	8.936	21.117	14.904	22.516	20.793
SEM (std of mean)	2.880		3.156		3.449	
	$(I_{\perp} - Cbg)$	$\sigma_{I_{\perp}}$	$(I_{\perp} - Cbg)/2$	$\sigma_{I_{\perp}}$	$(I_{\perp} - Cbg)/3$	$\sigma_{I_{\perp}}$
mean (20 cells)	18.708	8.936	21.117	14.904	22.516	20.793
SEM (std of mean)	2.880		3.156		3.449	

Table 6: Autofluorescence values with different exposure times. Of these values the camera background (Cbg) has been subtracted first, after which they were divided by their exposure time.

4.2 Applied measurements: Biological cells

A typical example of a measurement is given in figure 14. Reference measurements with FKBP and EGF have been done and the first modification called, 'kinase dood' has been measured (what this modification is, was not important for this research and will therefore not be explained, more about this can be found in Reiniers thesis[2]). To obtain the anisotropy values, the registration has been done on the cell images themselves, for every cell apart and a cutoff value of 0.02 has been used as threshold on the standard deviation. Average values of a number of cells are given, the number at the bottom of each bar indicates how many cells have been measured. The error bars indicate the SEM. As can be seen there is a drop in anisotropy upon adding AP, but the errorbars are still very large, especially for FKBP. Drop in anisotropy upon adding AP is 7.8 ± 6.8% in the case of FKBP and 11.3 ± 6.1% in the case of 2xFKBP. For EGF a drop was not observed.

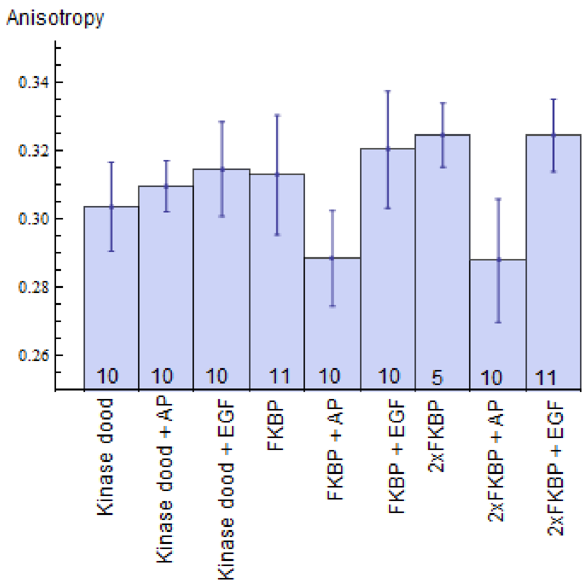
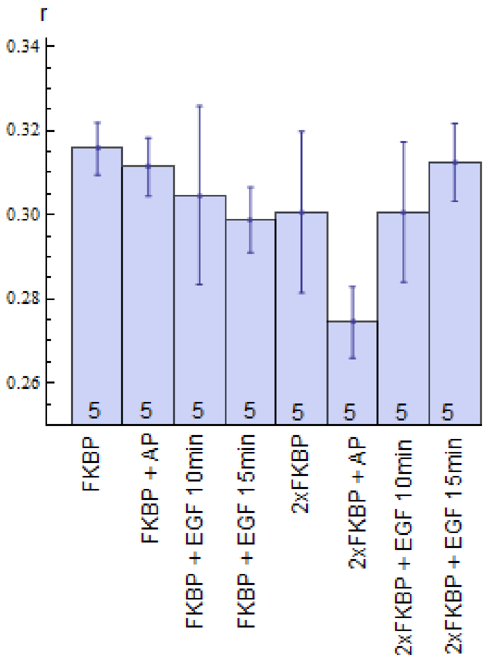


Figure 14: Data has been obtained on 11 april 2014. The registration was done on the images of cells themselves. The threshold was based on the standard deviation, using a cutoff of 0.02. The numbers at the bottom indicate how many cells have been measured with that particular treatment.

Figure 15 shows data of two other measurements. Mainly reference constructs have been measured, and the first modification. Again, registration has been done on the cell images themselves, for every cell apart and a cuttoff value of 0.02 has been used as threshold on the standard deviation. Since a drop in anisotropy upon adding EGF was not found, samples where treated a different amount of time: EGF "10min" and "EGF 15min". As can be seen, a drop in anisotropy can be seen when comparing FBKP and EGF 15min but for 2xFKBP this drop was not found. Actually for most measurements either a drop was not observed at all or the errorbars overlapped. Striking is the small errorbar for '2xFKBP' on the right of figure 15. Usually the standard deviation decreases when the number of measurements increases but for this dataset this seems not the be the case.



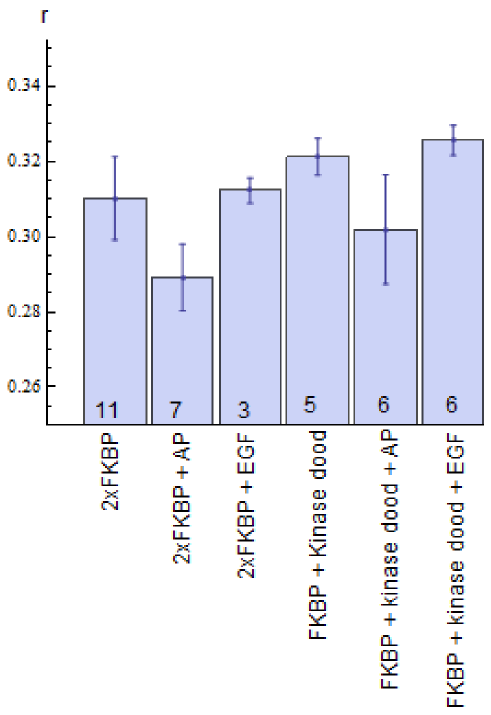


Figure 15: For all samples, the registration was done on the images of cells themselves. The threshold was based on the standard deviation, using a cutoff of 0.02. The numbers at the bottom indicate how many cells have been measured with that particular treatment. The average value of all these cells is given here, with the SEM.

5 Conclusion & discussion

The research question of this thesis was: can we detect clustering of EFGR with Homo-FRET? And if so, what is important for the clustering behavior of EGFR? Before answering these questions, first a few side questions encountered along the way will be discussed.

Registration

The registration process turned out to be one of the major chalanges of this project. So far, it was found the registration mainly consists of a rotation and choosing the middle of this rotation as the middle of the ROI does fasten the registration process but it does not completely solve it. There is still a mismatch in registration on beads or on cells. A possible explanation might be that the registration is z-dependent. Beads and cells are not recorded at the same height since focusing is done in between. It was briefly looked upon but fluctuations are too high to draw conclusions yet. If the z-dependence would be the problem, one way to solve this would be to put beads inside the cells. This way registration could be done on the position of the beads inside a cell.
Another issue was the pixel overlay, is the weighted average MatLab calculates the correct value? Why this might not be the case, can be best explained by looking at figure 16. When imposing a new raster (purple) on the old image (green) after the proper registration, the new pixel values contain information of several original pixels. Especially when there are sharp edges within the image, and therefore large differences between pixels, this might not be wanted. This can be improved by merging pixels as is shown in the bottom of figure 16. A disadvantage of this, is that it lowers your resolution. Be aware, this is very different from pixel binning, where the pixels are merged beforehand!

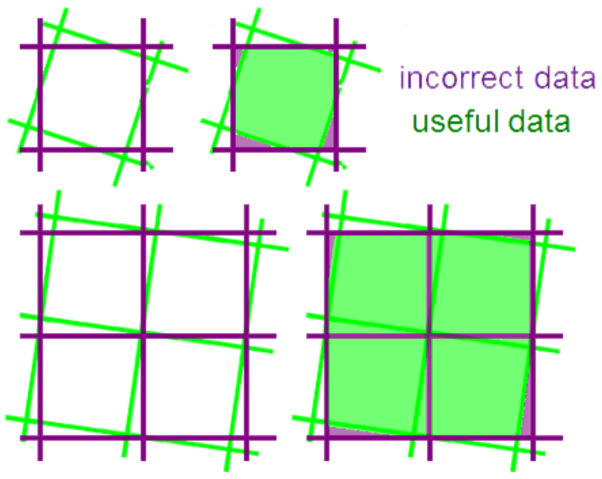


Figure 16: Overlaying pixels might give incorrect data. The green lines represent rotated, original pixels (perpendicular image), purple lines show the new raster imposed on the image. The latter is done to make the image fit to the size of the other (in our case parallel), non rotated image. As can be seen, this might result in some incorrect data.

G

Turning on the autofocus (even without using it) has a huge influence! Pay attention to these kind of settings. High intensity and multiple images give an accurate result. Their is some fluctuation in the G-factor so it does need to be recorded for every set of measurements. Luckily a small variation in registration does not have a large effect on the G-factor since it is quite homogeneous within a ROI. It is not very sensitive to where in the sample is measured either.

Anisotropy of cells

Going back to the research question, so far in some measurements a drop in anisotropy upon adding AP or EGF can be found but the standard deviation is still very high. Furthermore, this drop is not very consistent, in many measurements it is not found, so it seems too early to say this method works. Since it is found more often when adding AP then when adding EGF, Reinier suggested this might have a more biological reason. As can be seen from figure 17, when FKBP clusters upon adding AP, the mCherry's are automaticaly close to each other, but when EGFR clusters the MCherry's have more freedom to move around because they are attached to the carboxy terminal tail or, C-terminal tail (indicated by the green circle) which has some freedom to move, eventhough EGRF has clustered. Could it be that the mCherry's are too far from each other to measure homo-FRET even though EGFR has clustered? If this would be the case, it would be a huge drawback for this project. Measurements have been done with constructs only containing EGFR and mCherry (so FKBP was left out), but this did not improve the result.

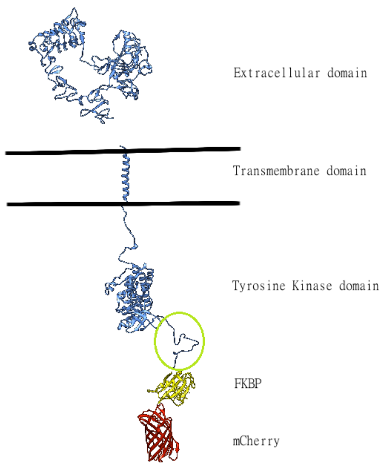


Figure 17: Structure of the fused protein. EGRF is shown in blue, FKBP in yellow and mCherry in red. The green circle indicated the C-terminal tail. Figure made by Reinier Damman.

In our best experiment a drop of 7.8 ± 6.8% was found upon adding AP when using FKBP and a drop of 11.3 ± 6.1% when using 2xFKBP. When using 2xFKBP, Nivard found a drop of either 6.5 ± 1.2% or 6.7 ± 1.2% depending on the autofluorescence value he subtracted.[4] The drop we found is larger so in a sense using mCherry improved the measurement but Nivards errorbars are much smaller so in that sense it worsened the result. All results are much lower than the drop of 14.9% found with confocal microscopy.[1] A difference between Nivard and what has been done for this thesis, is how the ROI was selected. To fasten the process a simple threshold was used for this research while Nivard used a manual selection on top of this. He did not include the nucleus of the cell, when calculating the average cell values. Since the EGF receptor is located in the membrane this is preferable but to do this by hand takes a lot of time, especially since the number of cells measured increased. For further research this might be looked into: is there a way to improve the selection of the ROI aside from doing it by hand for every individual cell?

Another part of this research was improving the signal to noise ratio by using mCherry instead of mGFP. Nivard reports autofluorescence values of 210/150 or 300/200 (parallel/perpendicular) and typical pixel values of 3000 to 6000 counts. In this research, autofluorescence values are comparable 388/237, but typical pixel values ranged from 10.000 to 35.000 so the signal to noise ratio did indeed improve. The overall variance did not decrease however. A possible reason might be that Nivard was more careful in selecting cells with more or less the same brightness. He used the same exposure time for all cells where we changed the exposure time to increase the signal therefore selecting cells which Nivard would have rejected. Or he simply did not measure enough cells to find so much fluctuations. He only measured four cells of a particular treatment where we measured at least 10.

6 Acknowledgements

First of all I would like to thank Hans Gerritsen for supervising during a part of this project. You always answered my questions with a lot of patience and a clear explanation or a nice discussion.
I would like to thank Gerhard Blab, for taking over the supervision of Hans Gerittsten and for helping with the data analysis. Your example of the script you wrote to analyse the beads taught me a lot.
Furthermore I would like to thank Dave van den Heuvel for always helping me out when I found another part of the set-up I did not understand. You always took the time to answer my numerous questions with a lot of patience.
For the first half year of this project I worked with Margriet Oomen. Especially when Hans left it was very nice to be able to discus the results of our measurements together, aside from this we always had a lot of fun. Thank you for both.
Reinier Damman and Cilia de Heus are the biologists I worked with. It was very nice to work with both of you. You taught me a lot about biology and the way biologists look at physics, which I found very interesting. An extra thank you goes out to Reinier for making all samples and helping/thinking along with the data analysis.

Last but not least, I would like to thank everyone in the MBFgroup for always thinking along when we encountered a new problem and of course for the fun during coffee breaks and the delicious cakes on thursday mornings.

References

[1] Arjen N. Bader et al. Homo-fret imaging enables quantification of protein cluster sizes with subcellular resolution. *Biophysics Journal*, 97, 2009.

[2] Reinier Damman. To be published. Master's thesis, University Utrecht, probably 2015.

[3] Margriet Oomen. Homo-fret detection in wide-field, steady-state microscopy, with mcherry as labeling fluorophore. Bachelorthesis, University Utrecht, 2014.

[4] Nivard Kagie. Homo-fret detection by fluorescence ploarization anisotropy in wide-field microscopy. Master's thesis, University Utrecht, 2013.

[5] Kiefer van Teuten. On using mcherry as a label for use in wide-field homo-fret induced fluorescence anisotropy microscopy. Bachelorthesis, University Utrecht, 2014.

[6] Cecillia de Heus et al. *Methods in Cell Biology*, volume 117, chapter 16: Analysis of EGF receptor oligomerization by Homo-FRET. Elsevier, 2013.

[7] David S. Fay and Ken Gerow. A biologist's guide to statistical thinking and analysis. *Wormbook*, 2013.

[8] Gabrielle S. Kaminski Schierle Fiona T.S. Chan, Clemens F.Kaminski. Homofret fluorescence anisotropy imaging as a tool to study molecular self-assembly in live cells. *ChemFhysChem*, 12.

A Appendix 1

To properly read in all files the folders and images need to have the correct names and all images should be tif images.
Make two folders named 'Parref' and 'Perref'. In the first folder the parallel reference images (so images of beads and rhodamine) are saved and in the second folder the perpendicular reference images. Reference images are named in the following way: bead images are named 'beads1', 'beads2', 'beads3', etc. (no limit on how many). Rhodamine images are named 'rhod' or 'AVG_rhod', their background images are named 'rhod_BG' or 'AVG_rhod_BG' (which of the two does not matter, but choose the same type for both). For the cells name two folders 'CAM1' and 'CAM2'. Save the parallel images in 'CAM1' and the perpendicular images in 'CAM2'. The name of the cell images contain a 'cellnumber' and the exposure time in the following way: 'cellnumber _ exposuretime s.tif'. An example is: '8_2s.tif' which means the 8th cell is measured with an exposure time of 2 seconds. If no exposure time is given it is assumed to be 1 second (so '6.tif' is the same as '6_1s.tif').
Analysis:

1. Read in the camera background, either drag the images into your workspace of write a script like 'BGinlezenF', This was done separately from the other reference measurements since the camera background is constant, so it is (probably) not necessary to measure it again, every time you measure.

2. Reference measurements. '[T, G, sigmaG] = allesrefF(link, N);' The input 'link' is a link to the directory where the folders 'Parref' and 'Perref' are located, make sure to end with a backslash. 'N' is the number of image you took and averaged over when measuring the G-factor and its background. The output T is the transformation matrix, calculated from the positions of the beads. G is the G-factor and sigmaG its uncertainty, both calculated per pixel.

3. Analyse cells and calculate anisotropy. Their are a few ways to do this. As said a threshold can be based on either intensity or σ, and either the registration matrix of the beads can be used, or it can be calculated for every cell apart.

   (a) '[groepmatrix, groepmatrixN, celmatrixN, anirow, sig_ani, Tcellen,x] = cellenallesin1_roisig_TpercelF(link, n, G, sigmaG,camerabgPAR,camerabgPER,treshold, path);' This script uses σ to determine a threshold and calculates the transformation matrix for every cell apart, using relative intensities.

   (b) '[celmatrix, groepmatrix, groepmatrixN, celmatrixN, anirow, sig_ani, x] = cellenallesin1_roisigF(link, n, G, sigmaG, T,camerabgPAR,camerabgPER,treshold, path);' This script uses σ to determine a threshold and uses one transformation matrix for all cells.

   (c) '[groepmatrix, groepmatrixN, celmatrixN, anirow, sig_ani, Tcellen,x] = cellenallesin1TpercelF(link, n, G, sigmaG, camerabgPAR, camerabgPER, path);' This script uses the intensity to determine a threshold ($I_{||} > 10.000$) and calculates the transformation matrix for every cell apart, using relative intensities.

   (d) '[celmatrix, groepmatrix, groepmatrixN, celmatrixN, anirow, sig_ani] = cellenallesin1F(link, n, T, G, sigmaG, camerabgPAR, camerabgPER, path); 'This script uses the intensity to determine a threshold ($I_{||} > 10.000$) and uses one transformation matrix for all cells.

   Input: 'link' is the directory which contains the folders 'CAM1' and 'CAM2' (make sure to end with a backslash), 'n' indicates which cells had the same treatment, for example if n = [0,5,10]; cell 1 till 5 belong to the same group (they underwent the same treatment) and cell 6 to 10 belong to another group. The fist number should always be 0. G and sigmaG are the G-factor and its uncertainty respectively as calculated from the reference measurements. camerabgPAR and camerabgPER are the camera background of the parallel and perpendicular camera respectively. Threshold is the threshold used to determine what part of the cell is background, usually this was set to 0.02. 'path' is the directory where MatLab saves the average values for every cell and group (again make sure to end with a backslash).
   Output: 'celmatrix' is a matrix with the average anisotropy with its uncertainty for every cell. 'groupmatrix' is a matrix with the average anisotropy with its uncertainty for every group. 'anirow'

is a row of images, where every image consists of the anisotropy values, per pixel, of a cell. 'sig_ani' is a row of images, where every image consists of the standard deviation in the anisotropy values, per pixel, of a cell. Standard deviation is based on photon statistics and propagation of uncertainty. 'x' is a matrix where the first column contains the exposure time of a cell, the second column the parallel image of a cell and the third column the perpendicular image.

B Appendix 2, references in MATLAB

In this appendix all MATLAB scripts can be found with which the background and reference measurements have been analyses.

```
function [camerabgPAR,camerabgPER] = BGinlezenF()

camerabgPAR = double(imread('D:\Kyo\oude (wide-field) opstelling\'
        'DATA\AVG_cameraBG_60x1spar.tif'));
camerabgPER = double(imread('D:\Kyo\oude (wide-field) opstelling\'
        'DATA\AVG_cameraBG_60x1sper.tif'));

        Error: File: D:\Kyo\oude (wide-field) opstelling\MATLAB\Uiteindelijke vers
        Expression or statement is incorrect--possibly unbalanced (, {, or [.
```

Published with MATLAB® R2013a

```
function [T,G,sigmaG] = allesrefF(link, N)
%Does all reference measurements: determines T (from beads) and G.

if(nargin<2)
    N = 1;
    %N = number of measurements done for G-factor,
    %of how many images is this the avarage?
end

%--------read in files of beads images--------------------------
beads = cell(1,2);
i=1;
while exist( strcat(link,'Parref\beads',num2str(i),'.tif') )==2
        try
    beads{i,1}=imread(strcat(link,'Parref\beads',num2str(i),'.tif'));
    beads{i,2}=imread(strcat(link,'Perref\beads',num2str(i),'.tif'));
        catch
    error(strcat('beads ', num2str(i), ' have not been found'))
        end
        i=i+1;
end

%--determine position of the beads in the images and registrate on them--
allpPAR = [];
allpPER=[];
maxdistance = 10;
%If the distance beween the positions in par and per is less than the
%maxdistance, the positions are assumed to be from the same bead.
for j=1:i-1
[pPAR,pPER] = positionF(beads{j,1}, beads{j,2}, maxdistance);
allpPAR = [allpPAR;pPAR];
allpPER = [allpPER;pPER];
end

T = alim_points_meerdereF(allpPAR, allpPER, 1);

%---------G-factor-------------------------------------------------------
try
    rhodPAR=imread(strcat(link,'Parref\AVG_rhod.tif'));
    rhodPER=imread(strcat(link,'Perref\AVG_rhod.tif'));
    rhodPAR_BG=imread(strcat(link,'Parref\AVG_rhod_BG.tif'));
    rhodPER_BG=imread(strcat(link,'Perref\AVG_rhod_BG.tif'));
catch
    try
        rhodPAR=imread(strcat(link,'Parref\rhod.tif'));
        rhodPER=imread(strcat(link,'Perref\rhod.tif'));
        rhodPAR_BG=imread(strcat(link,'Parref\rhod_BG.tif'));
        rhodPER_BG=imread(strcat(link,'Perref\rhod_BG.tif'));
    catch
    error(strcat('The rhodamine images have not been found.'))
    end
end
```

```
[G, sigmaG] = calgaangepastF(rhodPAR, rhodPER, rhodPAR_BG, rhodPER_BG, T,N);

end
```

```
        Error using allesrefF (line 13)
        Not enough input arguments.
```

Published with MATLAB® R2013a

```matlab
function [pPAR,pPER] = positionF(imPAR, imPER, maxdistance)
%Determines the position of the beads in the images.

%maxdistance is the maximum distance the beads are allowed to be separated
%to form a pair.

%pos1 &pos2 are the lists of positions of the beads found in each image
%seperatly. pPAR and pPER the lists of positions left after matching
%between the images (left out are beads which do not have a corresponding
%bead in the other image).

%gives comment when given the wrong imput
if (nargin<2)
    help position
end
if (nargin<3)
     error('You forget to give the maximum distances' ...
     'positions can be apart and still belong to the same bead.');
end
%---------------------------------------------------
%define variables
pPAR = []; pPER = [];
Nfound=0;

original = imPAR; orig_max = max(original(:));
orig_background=mean(original(:));
distorted = imPER; dist_max = max(distorted(:));
dist_background=mean(distorted(:));

%set threshold: normal: 0.2*max; change to 0.25*max for low S/N ratio.
orig_thresh = 0.2*orig_max + 0.8*orig_background;
dist_thresh = 0.2*dist_max + 0.8*dist_background;

%make a binary map of the image (mask)
orig_map = bwmorph(bwmorph(bwmorph(original>orig_thresh,'fill'),...
    'thicken'),'dilate');
dist_map = bwmorph(bwmorph(bwmorph(distorted>dist_thresh,'fill'),...
    'thicken'),'dilate');

%list of positions (x en y) for both images
pos_orig = regionprops(orig_map, original, {'WeightedCentroid'});
pos_dist = regionprops(dist_map, distorted, {'WeightedCentroid'});

%fprintf(1,'there are %d points in the original (par) and
    %d points in the distorted (per) image\n', ...
%   length(pos_orig), length(pos_dist))

pos1 = reshape([pos_orig.WeightedCentroid ], [2, length(pos_orig)])';
pos2 = reshape([pos_dist.WeightedCentroid ], [2, length(pos_dist)])';

%Match beads from images. Left out are beads which do not have a
%corresponding bead in the other image
```

```
for ii = 1:length(pos_orig)
   distance2 = sqrt((pos2(:,1)-pos1(ii,1)).^2 + ...
       (pos2(:,2)-pos1(ii,2)).^2);
   ind = find(distance2 == min(distance2));
   if (distance2(ind(1)) < maxdistance)
       Nfound = Nfound + 1;
       pPAR(Nfound,:) = pos1(ii,:);
       pPER(Nfound,:) = pos2(ind(1),:);
   end
end

%-------plots---------------------------------------------------------------

figure
%subplot(1,2,1),
imagesc(imPAR), title('Par'); colormap(gray), colorbar; axis equal tight, hold on
label1 = cellstr( num2str([1:length(pPAR)]') );  %'#labels correspond to their ord
plot(pPAR(:,1), pPAR(:,2), 'ro')
plot(pPAR(Nfound,1), pPAR(Nfound,2), 'yo')
text(pPAR(:,1), pPAR(:,2), label1, 'VerticalAlignment','bottom', ...
                             'HorizontalAlignment','right', 'Color',[1 1 1])
%subplot(1,2,2),
figure
imagesc(imPER), title('Per'); colormap(gray), colorbar; axis equal tight, hold on
label2 = cellstr( num2str([1:length(pPER)]') );  %'#labels correspond to their ord
plot(pPER(:,1), pPER(:,2), 'ro')
plot(pPER(Nfound,1), pPER(Nfound,2), 'yo')
text(pPER(:,1), pPER(:,2), label2, 'VerticalAlignment','bottom', ...
                             'HorizontalAlignment','right', 'Color',[1 1 1])

%fprintf(1,'there are now %d points left and indexed\n', ...
%    Nfound)

%-----------------------------------------------------------------------------

end
```

```
        Error: File: D:\Kyo\oude (wide-field) opstelling\MATLAB\Uiteindelijke vers
        Unexpected MATLAB expression.
```

Published with MATLAB® R2013a

```
function [pPARsorted,pPERsorted,R] =
    position_plugin_sideF(imPAR, imPER, pPAR, pPER)
%Sorteert de ingegeven posities en laat ze zien op de image.

%zorgt voor commentaar als je de verkeerde imput geeft
if (nargin<4)
    help positio_plugin
end

%-----------------------------------------------------
%deeltjes bij elkaar zoeken
r = zeros(length(pPER),1);
Nfound =0;
pPERsorted =[];
pPARsorted =[];
for i=1:size(pPAR,1)
    for j=1:size(pPER,1)
        r(j,1) = sqrt((pPAR(i,1) - pPER(j,1))^2 + (pPAR(i,2) - pPER(j,2))^2);
    end
    k = find(r == min(r));
    if r(k,1) < 20
        Nfound = Nfound+1;
        pPERsorted(Nfound,:) = pPER(k,:);
        pPARsorted(Nfound,:) = pPAR(i,:);
    end
end
%--------------------------------------------------------------------------------

fprintf(1,'there are now %d points left and sorted\n', ...
    Nfound)

figure
%subplot(1,2,1),
imagesc(imPAR), title('Par'); colormap(gray), colorbar; axis equal tight, hold on
%for i=1:length(pos1), plot(pos1(i,1), pos1(i,2), 'bo'); end
label1 = cellstr( num2str([1:length(pPARsorted)]') );
        %labels correspond to their order
plot(pPARsorted(:,1), pPARsorted(:,2), 'ro')
plot(pPARsorted(Nfound,1), pPARsorted(Nfound,2), 'yo')
text(pPARsorted(:,1), pPARsorted(:,2), label1, 'VerticalAlignment','bottom', ...
                             'HorizontalAlignment','right', 'Color',[1 1 1])
%subplot(1,2,2),
figure
imagesc(imPER), title('Per'); colormap(gray), colorbar; axis equal tight, hold on
%for i=1:length(pos2), plot(pos2(i,1), pos2(i,2), 'bo'); end
label2 = cellstr( num2str([1:length(pPERsorted)]') );
        %labels correspond to their order
plot(pPERsorted(:,1), pPERsorted(:,2), 'ro')
plot(pPERsorted(Nfound,1), pPERsorted(Nfound,2), 'yo')
text(pPERsorted(:,1), pPERsorted(:,2), label2, 'VerticalAlignment','bottom', ...
                             'HorizontalAlignment','right', 'Color',[1 1 1])
```

```
R = distanceF(pPARsorted,pPERsorted);

end
```

```
        Error: File: D:\Kyo\oude (wide-field) opstelling\MATLAB\Uiteindelijke vers
        Expression or statement is incomplete or incorrect.
```

Published with MATLAB® R2013a

```
function [T, wrong, R_new, allpPAR,pPERnew] =
    alim_points_meerdereF(allpPAR, allpPER, maxdistance)
%Determines the transformation by alligning points from MULTIPLE measurements.

%allpPAR and allpPER are the position of the beads in the two images,
%maxdistance is the maximum distance you want the beads to have after
%transformation.
%Old coordinates are pPAR & pPER, new coordinates: pPAR & pPERnew

if (nargin<2)
    help alim
    T = [];
end
if size(allpPAR)~=size(allpPER)
        error('Houston, we have the wrong matrices!');
end
if (nargin<3)
    maxdistance = 1;
end

%Determine the transformationmatrix with a affine transformation.
T= estimateGeometricTransform(allpPER,allpPAR,'affine');

%New position of the beads
pPER_n=[];
for i=1:length(allpPER)
pPER_n(i,:) = [allpPER(i,:),1]*T.T;
end
pPERnew = pPER_n(:,1:2);


%old distances:
R = distanceF(allpPAR,allpPER);
%new distances:
R_new = distanceF(allpPAR,pPERnew);

%all beads separated more than the maximum distance are put into the
%variable 'wrong' (positions after transformation) and 'wrongoud'
%(positions before transformation). Thisway they can be plotted on the
%image to check if something goes wrong in certain parts of the image (eg.
%the edges). They are not left out when calculating the transformation!

wrong=[];
wrongoud=[];
for i=1:length(R_new)
if R_new(i) > maxdistance
    fout = [allpPAR(i,:),pPERnew(i,:),R_new(i),i];
    wrong = [wrong; fout];
    foutoud = [allpPAR(i,:),allpPER(i,:),R_new(i),i];
    wrongoud = [wrongoud; foutoud];
end
end
```

```
%-----------plotting------------------
    figure,

    %original image
    l=1; b=2;
    subplot(l,b,1), imshow(0.5*ones(500)),
    title('original (blue = par, yellow = per)'); axis equal tight, hold on
    for i=1:length(allpPAR), plot(allpPER(i,1), allpPER(i,2), 'yo'),
        plot(allpPAR(i,1), allpPAR(i,2), 'bo'); end
if ~isempty(wrongoud)
    for i=1:length(wrongoud(:,1)), plot(wrongoud(i,1),wrongoud(i,2), 'gx')
        ,plot(wrongoud(i,3),wrongoud(i,4), 'yx'); end
    else
end
    %image after transformation
    subplot(l,b,2), imshow( imwarp(0.5*ones(500,500), T,'OutputView',
    imref2d(size(ones(500,500)))) ),
    title('transformed (blue = par, yellow = per)'); axis equal tight, hold on
    for i=1:length(pPERnew), plot(pPERnew(i,1), pPERnew(i,2), 'yo'),
        plot(allpPAR(i,1), allpPAR(i,2), 'bo'); end
if ~isempty(wrong)
    for i=1:length(wrong(:,1)), plot(wrong(i,1),wrong(i,2), 'gx'),
        plot(wrong(i,3),wrong(i,4), 'yx'); end
else fprintf(1,'No distances longer then %g pixels!\n', ...
        maxdistance)
end

end
```

```
        Error: File: D:\Kyo\oude (wide-field) opstelling\MATLAB\Uiteindelijke vers
        Expression or statement is incomplete or incorrect.
```

Published with MATLAB® R2013a

```
function [distances] = distanceF(p1,p2)
%Calculates the distances between all points in the lists.

if size(p1)~=size(p2)
        error('Houston, we have the wrong matrix!');
end

distances = sqrt((p1(:,1)-p2(:,1)).^2 + (p1(:,2)-p2(:,2)).^2);
end
```

```
        Error using distanceF (line 4)
        Not enough input arguments.
```

Published with MATLAB® R2013a

```
function [G, sigmaG]=
    calgaangepastF(rhodPAR, rhodPER, rhodPARbg, rhodPERbg, T,N)
%CALG Calculate your G-factor from images,
%their background and transform-matrix.

if(nargin<6)
    N = 1;
end

rhodPARzonderbg = rhodPAR - rhodPARbg;
rhodPERzonderbg = rhodPER - rhodPERbg;

rhodPERzonderbg=imwarp(rhodPERzonderbg, T,'OutputView',...
    imref2d(size(rhodPARzonderbg)));

G = double(rhodPARzonderbg)./double(rhodPERzonderbg); %calculate G

G(isnan(G))=0;
G(isinf(G))=0;
G(G>1.45)=0;
%nesecarry to set 'strange' values (due to non overlapping images because
%of registration) at the edge at 0.

figure; imshow(G); ggf; title('G-factor');

sigmaG = sigmaGfactorF(rhodPAR, rhodPER, T, G, rhodPARbg, rhodPERbg, N);
%calculate standard deviation of G

gemm = mean(nonzeros(G(50:450,50:450)));
gemsig1 = mean(nonzeros(sigmaG(50:450,50:450)));
%at the edges strange values can occur so it is better not to take the
%edges into account when calculating the average.


fprintf(1,'The mean value of G is %g \n', ...
    gemm)
fprintf(1,'The mean value sigma G is %g \n', ...
    gemsig1)

end
```

```
Error: File: D:\Kyo\oude (wide-field) opstelling\MATLAB\Uiteindelijke vers
Expression or statement is incomplete or incorrect.
```

Published with MATLAB® R2013a

```matlab
function [sigmaG]=sigmaGfactorF(rhodPAR, rhodPER, T, G, bgpar, bgper, N)
%Calculate the uncertainty in G-factor

if(nargin<7)
    N = 1;
end

bgpar = double(bgpar);
bgper = double(bgper);
Iper = double(imwarp(rhodPER, T,'OutputView',imref2d(size(rhodPAR))));
Ipar = double(rhodPAR);
a=1.69; %sqrt(alpha) = 1.3, determinde by Kiefer.

%sigma I = (sqrt(number of counts)*sqrt(factor alpha)) /sqrt(number of
%measurements used for averaging)
sigmaIpar = ((Ipar.*a)./N).^(1/2);
sigmaIper = ((Iper.*a)./N).^(1/2);
sigmabgpar = ((bgpar.*a)./N).^(1/2);
sigmabgper = ((bgper.*a)./N).^(1/2);

%derivatives
dG_ipar = 1./(Iper - bgper);
dG_iper = (bgpar - Ipar)./(bgper - Iper).^2;
dG_bgpar = 1./(bgper - Iper);
dG_bgper = (-bgpar + Ipar)./(bgper - Iper).^2;

%remove infinities and NaN (=not a number)
dG_ipar(isinf(dG_ipar))=0;
dG_ipar(isnan(dG_ipar))=0;
dG_iper(isinf(dG_iper))=0;
dG_iper(isnan(dG_iper))=0;

sigmaG = ((dG_ipar.*sigmaIpar).^2 + (dG_iper.*sigmaIper).^2 + ...
    (dG_bgpar.*sigmabgpar).^2 + (dG_bgper.*sigmabgper).^2).^(1/2);

%remove nonsense values at the edges
sigmaG(isnan(sigmaG))=0;
sigmaG(isinf(sigmaG))=0;
for i = 1:length(G)
    for j= 1:length(G)
        if G(i,j)==0
        sigmaG(i,j)=0;
        end
    end
end

if N==1 %usefull because large N reduces the sigma quite a lot.
    colormax = 0.025;
else
    colormax =0.01;
end
```

```
figure; imagesc(sigmaG); axis equal tight; caxis([0,colormax]);...
    colorbar; title('Standard deviation of the G-factor, per pixel')
end
```

```
        Error using sigmaGfactorF (line 8)
        Not enough input arguments.
```

Published with MATLAB® R2013a

```
function ggf
%GF Sets figure properties for G-factor figures.
caxis([1.00,1.45]); colormap(gray);
set(gcf,'Color',[1,1,1]); colorbar; truesize;
end

function afF
% properties of the colored images (anisotropy)
caxis([0.2,0.4]); colormap(jet); s
et(gcf,'Color',[1,1,1]); colorbar; axis equal tight %truesize;
end
```

```
        Error using truesize>ParseInputs (line 369)
        No images or texturemapped surfaces in the figure.

        Error in truesize (line 30)
        [axHandle, imHandle, imSize, ...

        Error in ggf (line 4)
        set(gcf,'Color',[1,1,1]); colorbar; truesize;
```

Published with MATLAB® R2013a

C Appendix 3, cells in MATLAB

In this appendix all MATLAB scripts can be found with which the cells have been analyses.

```
function [groepmatrix, groepmatrixN, celmatrixN, anirow, sig_ani, Tcellen,x]
= cellenallesin1_roisig_TpercelF(link, n, G, sigmaG,
    camerabgPAR,camerabgPER,treshold, path)
%Analyses cells, region of interest  based on standard deviation,
%calculates the T-matrix for every cel apart.

%-----------------------------------------------------------------------
%anirow     = row of images, every image contains the anisotropy values
%            (per pixel) of a cell.
%sig_ani    = row of images, every image contains the standard deviation of
%            the anisotropy values, per pixel.
%celmatrix  = matrix of the weighted average and standard deviation of
%            every cell.
%groepmatrix= matrix of the weighted average and standard deviation of
%            every group of cells.
%T          = row of registration matrices (2d affine objects), every
%            instance contains the registration needed for that particular
%            cell.


%--BACKGROUND VALUES------------------------------------------------------
%camera background, which is independent of you exposure time
bgpar = double(camerabgPAR);
bgper = double(camerabgPER);

%autofluorescence: determined seperately and filled in here, dependent
%on exposure time.
auto_par = double(388 - mean(bgpar(:)) );
sig_auto_par = double(223);
auto_per = double( 237 - mean(bgper(:)) );
sig_auto_per = double(123);
%-------------------------------------------------------------------------

aantalcellen = n(end);
[x]=cellen_inlezen_mexptF(link, aantalcellen);%reads in cell images
anirow = cell(1,aantalcellen);
sig_ani = cell(1,aantalcellen);
Tcellen = cell(1,aantalcellen);

for i=1:aantalcellen
    T = alimF(x{i,2},x{i,3}); %registration

%---calculate anisotropy and its standard deviation (of the whole image)---
    ani = anisotropy_1celF(x{i,2},x{i,3},x{i,1}, T, G, auto_par,...
        auto_per,bgpar,bgper);
    sigani = sigma_ani_cellen_zroiF(x{i,2}, x{i,3}, T, G, sigmaG, bgpar, ...
        bgper, auto_par, auto_per, sig_auto_par, sig_auto_per);

%-------------make a ROI --------------------------------
    %treshold = 0.03;

    for j=1:500
```

```
        for k=1:500
            if sigani(j,k)>treshold
            ani(j,k)=0;
            sigani(j,k)=0;
            end
            if x{i,2}(j,k)==53000 || x{i,3}(j,k)==53000;
                %saturated pixels have a value of 53000
            ani(j,k)=0;
            sigani(j,k)=0;
            end
        end
    end
%-------------------------------------------------------------
    sigani(isnan(sigani))=0;
    sigani(isinf(sigani))=0;

    %-- Put found values into a 'tabel' ---------------------
    anirow{1,i} = ani;
    sig_ani{1,i} = sigani;
    Tcellen{1,i} = T;

%     figure, %Plot anisotropy and its standard deviation
%     subplot(1,2,1), imagesc(anirow{1,i}),afF, ...
%       title(strcat('anisotropy cel ',num2str(i)));
%     subplot(1,2,2), imagesc(sig_ani{1,i}), colorbar, ...
%       caxis([0,treshold]),axis equal tight, title('sigma in anisotropy, per pixe
end

%---calculate mean and variance per cel and per group--------------
if (nargin>7)
[groepmatrix, celmatrix] = cellen_groeperenF(anirow, sig_ani, n, path);
[groepmatrixN, celmatrixN] = cellen_groeperen_simpelerF(anirow, sig_ani, n, path);
else
[groepmatrix, celmatrix] = cellen_groeperenF(anirow, sig_ani, n);
[groepmatrixN, celmatrixN] = cellen_groeperen_simpelerF(anirow, sig_ani, n);
end

        Error: File: D:\Kyo\oude (wide-field) opstelling\MATLAB\Uiteindelijke vers
        Expression or statement is incomplete or incorrect.
```

Published with MATLAB® R2013a

```
function [celmatrix, groepmatrix, groepmatrixN, celmatrixN, anirow, sig_ani, x]
= cellenallesin1_roisigF(link, n, G, sig_G, T,camerabgPAR,camerabgPER
            ,treshold, path)
%Analyses cells, bases the region of interest (roi) on the standard
%deviation.

%anirow     = row of images, every image contains the anisotropy values
%            (per pixel) of a cell.
%sig_ani    = row of images, every image contains the standard deviation of
%            the anisotropy values, per pixel.
%celmatrix  = matrix of the weighted average and standard deviation of
%            every cell.
%groepmatrix= matrix of the weighted average and standard deviation of
%            every group of cells.

%--BACKGROUND VALUES---------------------------------------------------
%camera background, which is independent of the exposure time
bgpar = double(camerabgPAR);
bgper = double(camerabgPER);

%autofluorescence: determined seperately and filled in here, dependent
%on exposure time.
auto_par = double(388 - mean(bgpar(:)) );
sig_auto_par = double(223);
auto_per = double( 237 - mean(bgper(:)) );
sig_auto_per = double(123);
%--------------------------------------------------------------------

aantalcellen = n(end);
[x]=cellen_inlezen_mexptF(link, aantalcellen);%read in cell images
anirow = cell(1,aantalcellen);
sig_ani = cell(1,aantalcellen);

for i=1:aantalcellen

%---calculate anisotropy and its standard deviation (of the whole image)---
    ani = anisotropy_1celF(x{i,2},x{i,3},x{i,1}, T, G, auto_par, ...
        auto_per,bgpar,bgper);
    sigani = sigma_ani_cellen_zroiF(x{i,2}, x{i,3}, T, G, sig_G, ...
        bgpar, bgper, auto_par, auto_per, sig_auto_par, sig_auto_per);

%---make ROI-----------------------------------------------
%    treshold = 0.03;

    for j=1:500
        for k=1:500
            if sigani(j,k)>treshold
            ani(j,k)=0;
            sigani(j,k)=0;
            end
            if x{i,2}(j,k)==53000 || x{i,3}(j,k)==53000;
                %saturated pixels have a value of 53000
            ani(j,k)=0;
            sigani(j,k)=0;
            end
        end
    end
%------------------------------------------------------------
    sigani(isnan(sigani))=0;
    sigani(isinf(sigani))=0;

    %-- Put found values into a 'tabel' ---------------------
    anirow{1,i} = ani;
    sig_ani{1,i} = sigani;

end

%---calculate mean and variance per cel and per group--------------
if (nargin>8)
[groepmatrix, celmatrix] = cellen_groeperenF(anirow, sig_ani, n, path);
[groepmatrixN, celmatrixN] = cellen_groeperen_simpelerF(anirow, sig_ani, n, path);
else
[groepmatrix, celmatrix] = cellen_groeperenF(anirow, sig_ani, n);
[groepmatrixN, celmatrixN] = cellen_groeperen_simpelerF(anirow, sig_ani, n);
end

        Error: File: D:\Kyo\oude (wide-field) opstelling\MATLAB\Uiteindelijke vers
        Expression or statement is incomplete or incorrect.
```

Published with MATLAB® R2013a

```
function [celmatrix, groepmatrix, groepmatrixN, celmatrixN, anirow, sig_ani]
= cellenallesin1F(link, n, T, G, sig_G, camerabgPAR, camerabgPER, path)
%Analyses cells, bases the region of interest (roi) on intensity

%anirow     = row of images, every image contains the anisotropy values
%            (per pixel) of a cell.
%sig_ani    = row of images, every image contains the standard deviation of
%            the anisotropy values, per pixel.
%celmatrix  = matrix of the weighted average and standard deviation of
%            every cell.
%groepmatrix= matrix of the weighted average and standard deviation of
%            every group of cells.

%--BACKGROUND VALUES----------------------------------------------------
%camera background, which is independent of the exposure time
bgpar = double(camerabgPAR);
bgper = double(camerabgPER);

%autofluorescence: determined seperately and filled in here, dependent
%on exposure time.
auto_par = double(388 - mean(bgpar(:)) );
sig_auto_par = double(223);
auto_per = double( 237 - mean(bgper(:)) );
sig_auto_per = double(123);
%----------------------------------------------------

aantalcellen = n(end);
x = cellen_inlezen_mexptF(link, aantalcellen);%Read in cells
roi = maakroizpF(x, aantalcellen);%determine the roi

%HERE THE CALCULATIONS START------------------------------
anirow = cell(1,aantalcellen);
sig_ani = cell(1,aantalcellen);

for i=1:aantalcellen
    roicel = double(roi{i});

    %---calculate anisotropy and its standard deviation -------
    ani = anisotropy_1celF(x{i,2},x{i,3},x{i,1}, T, G, auto_par, ...
        auto_per,bgpar,bgper);%anisotropy of the whole image
    anirow{i}=ani.*roicel;%anisotropy of a cel
    sigma_ani = sigma_ani_cellen_zroiF(x{i,2}, x{i,3}, T, G, sig_G, ...
        bgpar, bgper, auto_par, auto_per, sig_auto_par, sig_auto_per);
    %gives standard deviation of the whole image
    sig_ani{1,i} = sigma_ani.*roicel;

end

%---calculate mean and variance per cel and per group--------------
if (nargin>7)
[groepmatrix, celmatrix] = cellen_groeperenF(anirow, sig_ani, n, path);
[groepmatrixN, celmatrixN] = cellen_groeperen_simpelerF(anirow, ...
    sig_ani, n, path);
else
[groepmatrix, celmatrix] = cellen_groeperenF(anirow, sig_ani, n);
[groepmatrixN, celmatrixN] = cellen_groeperen_simpelerF(anirow,...
    sig_ani, n);
end

figure, bar(groepmatrix(1,:))
```

```
        Error: File: D:\Kyo\oude (wide-field) opstelling\MATLAB\Uiteindelijke vers
        Expression or statement is incomplete or incorrect.
```

Published with MATLAB® R2013a

```
function [groepmatrix, groepmatrixN, celmatrixN, anirow, sig_ani, Tcellen,x]
= cellenallesin1TpercelF(link, n, G, sig_G, camerabgPAR, camerabgPER, path)
%Analyses cells, bases the region of interest (roi) on intensity,
%calculates the registration (T-matrix) for every cell apart.
%
%anirow      = row of images, every image contains the anisotropy values
%             (per pixel) of a cell.
%sig_ani     = row of images, every image contains the standard deviation of
%             the anisotropy values, per pixel.
%celmatrix   = matrix of the weighted average and standard deviation of
%             every cell.
%groepmatrix= matrix of the weighted average and standard deviation of
%             every group of cells.
%T           = row of registration matrices (2d affine objects), every
%             instance contains the registration needed for that particular
%             cell.

%--BACKGROUND VALUES-----------------------------------------------------
%camera background, which is independent of the exposure time
bgpar = double(camerabgPAR);
bgper = double(camerabgPER);

%autofluorescence: determined seperately and filled in here, dependent
%on exposure time.
auto_par = double(388 - mean(bgpar(:)) );
sig_auto_par = double(223);
auto_per = double( 237 - mean(bgper(:)) );
sig_auto_per = double(123);
%----------------------------------------------------

aantalcellen = n(end);
[x]=cellen_inlezen_mexptF(link, aantalcellen);%Read in cells
roi = maakroizpF(x, aantalcellen);%Determine the roi

%HERE THE CALCULATIONS START------------------------------
anirow = cell(1,aantalcellen);
sig_ani = cell(1,aantalcellen);
Tcellen = cell(1,aantalcellen);

for i=1:aantalcellen
    Tcellen{1,i}=alimF(x{i,2} , x{i,3}); %registration
    roicel = double(roi{i});

    %---calculate anisotropy and its standard deviation -------
    ani = anisotropy_1celF(x{i,2},x{i,3},x{i,1}, Tcellen{1,i}, G,...
        auto_par, auto_per,bgpar,bgper);%anisotropy of the whole image
    anirow{1,i}=ani.*roicel;%anisotropy of a cel
    sigma_ani = sigma_ani_cellen_zroiF(x{i,2}, x{i,3}, Tcellen{1,i},...
        G, sig_G, bgpar, bgper, auto_par, auto_per, sig_auto_par, sig_auto_per);
    %standard deviation of the whole image
    sig_ani{1,i} = sigma_ani.*roicel;
    %----------------------------------------------------------
```

```
%     figure, %Plot anisotropy and its standard deviation
%     subplot(1,2,1), imagesc(anirow{1,i}),afF, title(strcat('anisotropy cel ',num
%     subplot(1,2,2), imagesc(sig_ani{1,i}), colorbar, caxis([0,0.04]),axis equal
end

%---calculate mean and variance per cel and per group--------------
if (nargin>6)
[groepmatrix, celmatrix] = cellen_groeperenF(anirow, sig_ani, n, path);
[groepmatrixN, celmatrixN] = cellen_groeperen_simpelerF(anirow, sig_ani, n, path);
else
[groepmatrix, celmatrix] = cellen_groeperenF(anirow, sig_ani, n);
[groepmatrixN, celmatrixN] = cellen_groeperen_simpelerF(anirow, sig_ani, n);
end
```

```
        Error: File: D:\Kyo\oude (wide-field) opstelling\MATLAB\Uiteindelijke vers
        Expression or statement is incomplete or incorrect.
```

Published with MATLAB® R2013a

```
function [x]=cellen_inlezen_mexptF(link, aantalcellen)
%Reads in images of cells (with different exposure times). It returns a
%matrix x, where the first collum contains the exposure time, the second
%collum the parallel image, the third collum the perpendicular image.

%Possible exposure times: 1s (default), 2s, 3s, 4s, 5s, 10s, 0.3, 0.4, 0.5,
%0.6, 0.7, 0.75, 0.8, 0.9

x = cell(aantalcellen,3);
for i=1:aantalcellen
        try%default, when not given the exposure time is assumed to be 1s
    x{i,2}=imread(strcat(link,'CAM1\',num2str(i),'.tif'));
    x{i,3}=imread(strcat(link,'CAM2\',num2str(i),'.tif'));
    x{i,1} = 1;
        catch
        try      %read in cells with an exposure time of 1s
     x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_1s.tif'));
     x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_1s.tif'));
     x{i,1} = 1;
        catch
        try      %read in cells with an exposure time of 2s
     x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_2s.tif'));
     x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_2s.tif'));
     x{i,1} = 2;
        catch
        try      %read in cells with an exposure time of 3s
     x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_3s.tif'));
     x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_3s.tif'));
     x{i,1} = 3;
        catch
        try      %read in cells with an exposure time of 4s
     x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_4s.tif'));
     x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_4s.tif'));
     x{i,1} = 4;
        catch
        try      %read in cells with an exposure time of 5s
     x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_5s.tif'));
     x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_5s.tif'));
     x{i,1} = 5;
        catch
        try      %read in cells with an exposure time of 10s
     x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_10s.tif'));
     x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_10s.tif'));
     x{i,1} = 10;
        catch
        try      %read in cells with an exposure time of 0.9s
     x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_09s.tif'));
     x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_09s.tif'));
     x{i,1} = 0.9;
        catch
        try      %read in cells with an exposure time of 0.9s
     x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_0.9s.tif'));
```

```
x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_0.9s.tif'));
x{i,1} = 0.9;
   catch
   try      %read in cells with an exposure time of 0.8s
x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_08s.tif'));
x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_08s.tif'));
x{i,1} = 0.8;
   catch
   try      %read in cells with an exposure time of 0.5s
x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_0.8s.tif'));
x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_0.8s.tif'));
x{i,1} = 0.8;
   catch
   try      %read in cells with an exposure time of 0.75s
x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_075s.tif'));
x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_075s.tif'));
x{i,1} = 0.75;
   catch
   try      %read in cells with an exposure time of 0.75s
x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_0.75s.tif'));
x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_0.75s.tif'));
x{i,1} = 0.75;
   catch
   try      %read in cells with an exposure time of 0.5s
x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_07s.tif'));
x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_07s.tif'));
x{i,1} = 0.7;
   catch
   try      %read in cells with an exposure time of 0.5s
x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_0.7s.tif'));
x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_0.7s.tif'));
x{i,1} = 0.7;
   catch
   try      %read in cells with an exposure time of 0.6s
x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_06s.tif'));
x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_06s.tif'));
x{i,1} = 0.6;
   catch
   try      %read in cells with an exposure time of 0.6s
x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_0.6s.tif'));
x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_0.6s.tif'));
x{i,1} = 0.6;
   catch
   try      %read in cells with an exposure time of 0.5s
x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_05s.tif'));
x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_05s.tif'));
x{i,1} = 0.5;
   catch
   try      %read in cells with an exposure time of 0.5s
x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_0.5s.tif'));
x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_0.5s.tif'));
x{i,1} = 0.5;
   catch
   try      %read in cells with an exposure time of 0.4s
```

```
    x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_04s.tif'));
    x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_04s.tif'));
    x{i,1} = 0.4;
       catch
       try      %read in cells with an exposure time of 0.4s
    x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_0.4s.tif'));
    x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_0.4s.tif'));
    x{i,1} = 0.4;
       catch
       try      %read in cells with an exposure time of 0.3s
    x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_03s.tif'));
    x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_03s.tif'));
    x{i,1} = 0.3;
       catch
       try      %read in cells with an exposure time of 0.3s
    x{i,2}=imread(strcat(link,'CAM1\', num2str(i), '_0.3s.tif'));
    x{i,3}=imread(strcat(link,'CAM2\', num2str(i), '_0.3s.tif'));
    x{i,1} = 0.3;
       catch
   error(strcat('no image of cell ' , num2str(i), ' has been found'))
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
       end
end %end for loop

end
```

```
        Error using cellen_inlezen_mexptF (line 9)
        Not enough input arguments.
```

Published with MATLAB® R2013a

```
function [roi] = maakroizpF(x, aantalcellen)
%Determines a region of interest, based on a cuttof level of 10.000.
%Returns a binary image which is 1 whithin the cell and zero outside.

roi=cell(aantalcellen,1);
cutofflevel = 10000;

for i=1:aantalcellen
immask = x{i,2};
immask(immask>52999)=0;%sets saturated values (=53000) to zero
immask(immask<cutofflevel)=0;
immask(immask>cutofflevel)=1;
immask = bwmorph(immask,'majority',10);
roi{i,1} = immask;
end

end
```

```
Error using maakroizpF (line 5)
Not enough input arguments.
```

Published with MATLAB® R2013a

```
function [T] = alimF(imPAR, imPER)
%Image regristration: Determines the transformation between the cameras by
%regristration of the whole image. Uses relative intensities to compare the
%images.

if (nargin<2)
    help alim
    T = [];
end

optimizer = registration.optimizer.RegularStepGradientDescent;
optimizer.MaximumIterations=100;
metric = registration.metric.MattesMutualInformation;

%imPER is the moving image while imPAR is kept fixed.
T = imregtform(imPER, imPAR, 'affine', optimizer, metric);

figure;
imshowpair(imPAR,imPER)
figure;
imshowpair(imPAR,imwarp(imPER,T,'OutputView',imref2d(size(imPAR))))

end
```

```
 ALIM Alpha limits.
    AL = ALIM             gets the alpha limits of the current axes.
    ALIM([AMIN AMAX])     sets the alpha limits.
    ALMODE = ALIM('mode') gets the alpha limits mode.
    ALIM(mode)            sets the alpha limits mode.
                             (mode can be 'auto' or 'manual')
    ALIM(AX,...)          uses axes AX instead of current axes.
 
    ALIM sets or gets the Alim or AlimMode property of an axes.
 
    See also ALPHA, ALPHAMAP, CAXIS, COLORMAP.

    Reference page in Help browser
       doc alim

Error using alimF (line 16)
Not enough input arguments.
```

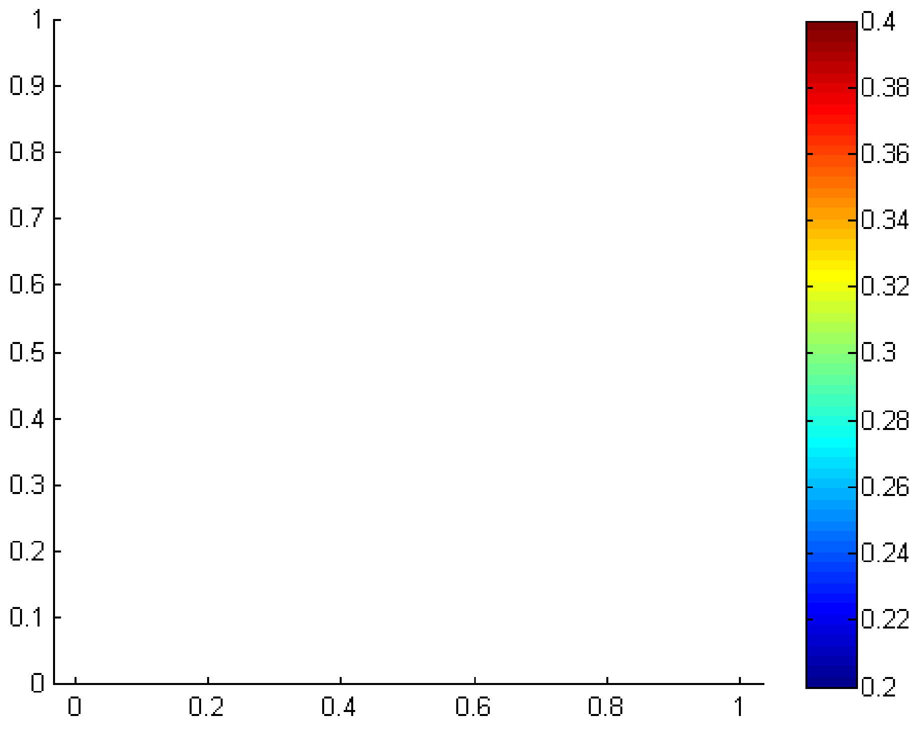
Published with MATLAB® R2013a

```
function [ani] =
anisotropy_1celF(impar,imper,exptime, T, G, auto_par, auto_per,bgpar,bgper)
%Calculates the anisotropy of a cell. Takes into acount the exposure time
%when subtrackting the background!!

%Subtract background
imPAR = double(impar) - double(bgpar) - auto_par.*exptime;
imPER = double(imper) - double(bgper) - auto_per.*exptime;
%Registration
imPER = imwarp(imPER, T,'OutputView',imref2d(size(imPAR)));
%Calculate anisotropy
ani = (imPAR - G.*imPER)./(imPAR + 2*G.*imPER);
ani(isnan(ani))=0;
ani(isinf(ani))=0;
ani(ani>0.99)=0;
%Setting edges to 0 (on the edges G is 0, so the anisotropy 1).

end
```

```
        Error: File: D:\Kyo\oude (wide-field) opstelling\MATLAB\Uiteindelijke vers
        Expression or statement is incomplete or incorrect.
```

Published with MATLAB® R2013a

```
function afF
% properties of the colored images (anisotropy)
caxis([0.2,0.4]); colormap(jet); set(gcf,'Color',[1,1,1]); ...
        colorbar; axis equal tight %truesize;
end
```



Published with MATLAB® R2013a

```matlab
function [groepmatrix, celmatrix] =
        cellen_groeperen_simpelerF(anirow, sig_ani, n, path)
%Calculates (weighted) avarage and variance of anisotropy of each cell and
%of each group. Adding a directory (path) saves the cel and group values.

aantalcellen = n(end);
%cells
gewogengmcel=zeros(aantalcellen,1);
stdcel=zeros(aantalcellen,1);
%groups
meancels = zeros(1,length(n)-1);
stdmean = zeros(1,length(n)-1);


% i = celnumber, j = groupnnumber
for i = 1:n(end)   %---cells------------------
        weigth = 1./(sig_ani{1,i}.^2); %weight is 1/sigma
        weigth(isnan(weigth))=0;
        weigth(isinf(weigth))=0;
        aniweight = weigth.*anirow{1,i};
        gewogengmcel(i,1)=(sum(aniweight(:)))./sum(weigth(:));
        stdcel(i,1)=std(nonzeros(anirow{1,i})); %variation within cell
end
celmatrix = [gewogengmcel,stdcel];

gewogengmcel(isnan(gewogengmcel))=0;
gewogengmcel(isinf(gewogengmcel))=0;
stdcel(isnan(stdcel))=0;
stdcel(isinf(stdcel))=0;

for j=1:length(n)-1 %---for every group-------

    meancels(1,j) = mean(nonzeros(gewogengmcel( n(j)+1:n(j+1),1 )));
    stdmean(1,j) = std(nonzeros(gewogengmcel( n(j)+1:n(j+1),1 )));

end
groepmatrix = [meancels;stdmean];


if (nargin>3)
    if isdir(path) %if path is a directory
pathname = fileparts(path);
%----saving the group values----------------------
testsave=fullfile(pathname,'meanofcels.mat');
save(testsave,'meancels')
testsave=fullfile(pathname,'stdofmean.mat');
save(testsave,'stdmean')
%---saving cell values---------------------------
testsave=fullfile(pathname,'gemcellen.mat');
save(testsave,'gewogengmcel')
testsave=fullfile(pathname,'sigcellen.mat');
save(testsave,'stdcel')
```

```
    else
        error('Wrong directory for saving!')
    end
end
```

```
        Error: File: D:\Kyo\oude (wide-field) opstelling\MATLAB\Uiteindelijke vers
        Expression or statement is incomplete or incorrect.
```

Published with MATLAB® R2013a

```
function [sig_ani] = sigma_ani_cellen_zroiF(I_par, I_per, T, G, sig_G,
bgpar, bgper, auto_par, auto_per, sig_auto_par, sig_auto_per)
%Calculates the standard deviation of the anisotropy per pixel, no region
%of interest is used, it returns the 'whole image'. The standard deviation
%is calculated by propagation of uncertainty with the differential methode.

%Derivatives used here are calculated with Mathematica.

a=1.69;
G = double(G);
I_par = double(I_par);
I_per = double(imwarp(I_per, T,'OutputView',imref2d(size(I_par))));

%---derivatives and uncertainties------------------------------
denom = (auto_par + bgpar + 2.*auto_per.*G + 2.*bgper.*G - I_par - ...
    2.*G.*I_per).^2;%denominator

d_ani_I_par = double( ( 3.*G.*(I_per - bgper - auto_per))./denom ); %derivative
sig_I_par = double((I_par.*a).^(1/2)); %uncertainty

d_ani_I_per = double( (3.*G.*(-I_par + bgpar + auto_par))./denom );%derivative
sig_I_per = double((I_per.*a).^(1/2));%uncertainty

d_ani_G = double( (-3.*(auto_par+bgpar-I_par).*(auto_per+bgper-I_per))./denom );%d
sig_G = double(sig_G);%uncertainty

d_ani_auto_par = double( ( 3.*G.*(auto_per+bgper - I_per))./denom );%derivative
d_ani_auto_per = double( (-3.*G.*(auto_par+bgpar - I_par))./denom );%derivative

N=30; %number of images taken for the background measurement
d_ani_bgpar = double( (3.*G.*(auto_per + bgper - I_per))./denom);%derivative
d_ani_bgper = double( -(3.*G.*(auto_par + bgpar - I_par))./denom);%derivative
sig_bgpar = double( ((bgpar.*a)./N).^(1/2) );%uncertainty
sig_bgper = double( ((bgper.*a)./N).^(1/2) );%uncertainty
%---------------------------------------------------------------

sig_ani = ((              (d_ani_I_par.*sig_I_par).^2 ...
                        + (d_ani_I_per.*sig_I_per).^2 ...
                        + (d_ani_auto_par.*sig_auto_par).^2 ...
                        + (d_ani_auto_per.*sig_auto_per).^2 ...
                        + (d_ani_G.*sig_G).^2 ...
                        + (d_ani_bgpar.*sig_bgpar).^2 ...
                        + (d_ani_bgper.*sig_bgper).^2 ...
                         ).^(1/2));

sig_ani(isnan(sig_ani))=0;
sig_ani(isinf(sig_ani))=0;

end

        Error: File: D:\Kyo\oude (wide-field) opstelling\MATLAB\Uiteindelijke vers
        Expression or statement is incorrect--possibly unbalanced (, {, or [.
```

Published with MATLAB® R2013a