%%%%%% README %%%%%%%

This repository contains all the code and data necessary to replicate the results of eMSIM simulations as well as a pattern estimation function.

The PSFs provided use a pixel size of 32nm and were calculated using the work of Xie et al. whose code, originally for modelling STED PSFs is freely available through their publication. Also included are the resolution targets used.

%%%%% compare\_reconstructions.m %%%%%%%

This is the primary program that compares diffraction limited, MSIM and eMSIM imaging techniques. It can also be used to compare with FED microscopy simply by showing the mean projection of the D\_IWS image stack. One side-effect of how the patterns are calculated in the eMSIM process is a boarder artefact at the edges of the image. This results from incomplete illumination at the edges of the image and can be easily can be negated by zero-padding or cropping the image after reconstruction.

%%%%%% find\_maxima\_gpu.m %%%%%%%

A very useful GPU accelerated function to find and return the coordinates of the local maxima in any image. The function takes the input arguments (image, OTF, threshold). Image is a double array of pixel values. OTF is the expected OTF of the image and threshold defines the sensitivity to peaks. The function works by filtering the image using the OTF to remove high frequency noise. It then subtracts the threshold from the image, rejects negative values and uses the function imregionalmax to give a binary image of local maxima. It then uses imerode(image) and find(image) to find the centre coordinates of the maxima.

%%%%%% rebuilt\_estimator.m %%%%%%%

This is an updated version of the pattern estimator and is faster and more stable than previous versions. The process is outlined in the eMSIM publication. This is also under continuous development and will be periodically updated.

The function starts by filtering the raw images using the find\_maxima\_gpu.m function. Although this is an unusual filtering method we have found it very successful and surprisingly quick to implement. Every Mth image in the stack (where M is the spacing) is used to determine the spacing and rotation. The spacing is determined by shifting the filtered images over themselves and looking for a minimum RMS error. It shifts +/- 5 of the estimated spacing although this can simply be changed if the spacing is not known to this accuracy, though this will obviously increase computation time. From the estimated spacings an ideal complete grid is created using the generate\_illum\_patterns.m function. The rotation for every Mth image is determined by rotating the ideal grid and optimising shift for that rotation. The rotations are then averaged with outliers removed and the negative of this rotation is applied to the image data. After rotation has been applied the spacings are re-calculated to account for any change due to the rotation. Correction for spherical aberration is then done using the optimise\_skew.m function although this is not required for the test data.