Implementation of an Automatic Image Registration Tool

Pavel A. Koshevoy, Tolga Tasdizen, and Ross T. Whitaker

UUSCI-2006-020

Scientific Computing and Imaging Institute
University of Utah
Salt Lake City, UT 84112 USA

May 2, 2006

Abstract:

This paper outlines the basic steps in the design and implementation of an intensity based automated Transmission Electron Microscopy (TEM) image mosaicing application, and highlights some of the implementation details, such as the tile matching, mosaic layout, and tile distortion correction.

Implementation of an automatic image registration tool

Pavel A. Koshevoy, Tolga Tasdizen, Ross T. Whitaker

May 2, 2006

Abstract

This paper outlines the basic steps in the design and implementation of an intensity based automated Transmission Electron Microscopy (TEM) image mosaicking application, and highlights some of the implementation details, such as the tile matching, mosaic layout, and tile distortion correction.

1 Motivation

The goal of this project is to provide a fully automatic tool for image registration and mosaicking of several hundred high-resolution images. This tool is primarily aimed at researchers working with Transmission Electron Microscopy images. A microscope rarely has a large enough field of view to cover the area of interest to the scientist with reasonable detail. Therefore, the area of interest has to be imaged as a sequence of tiles, following some overlapping tile pattern. The original area of interest is later reconstructed by laying out the image tiles into a mosaic. One problem particular to the microscopy images arises from the fact that the imaging process introduces distortion into the images. Thus, even if the exact layout is known for the image tiles, the tiles may not match perfectly in the overlap region. When the number of tiles is more than just a few, the task of laying out the mosaic quickly becomes daunting, and is a prime candidate for automation.

2 Problem statement

Given a large number of tiles specified in no particular order, a mosaic must be constructed and individual tiles must be corrected for distortion. This is the global problem that can be split up into slightly more manageable sub-problems:

- Find pairs of matching tiles.
- Build a rough estimate of the mosaic without distortion correction.
- Iteratively refine the mosaic by un-warping and adjusting the position of each tile simultaneously.

3 Description of the mathematics and algorithms

3.1 Matching pairs of tiles

Finding matching tiles amounts to finding tiles with highest cross-correlation. The method for finding matching tiles implemented in this application is based on a technique described by Girod and Kuo[1]. The technique is very straight forward, but it has an important prerequisite - it requires that the width and height of the two tiles must match. If that is not the case, one or both of the tiles must be padded on the bottom and on the right side with zeros until both of the tiles have matching dimensions as follows: given unpadded tiles U_0 and U_1, padded tiles S_0 and S_1 are generated such that $width\left(S_0\right) = width\left(S_1\right) = \max\left(width\left(U_0\right), width\left(U_1\right)\right)$ and $height\left(S_0\right) = height\left(S_1\right) = \max\left(height\left(U_0\right), height\left(U_1\right)\right)$.

Having satisfied the prerequisite by padding the tiles, the tiles are transformed into the frequency domain by Discrete Fourier Transform $F_0 = F\left\{S_o\right\}$ and $F_1 = F\left\{S_1\right\}$. The Discrete Fourier Transform functionality is provided by the FFTW[4] library. Once the tiles have been transformed, the cross-correlation Φ_{10} between S_1 and S_0 is calculated as

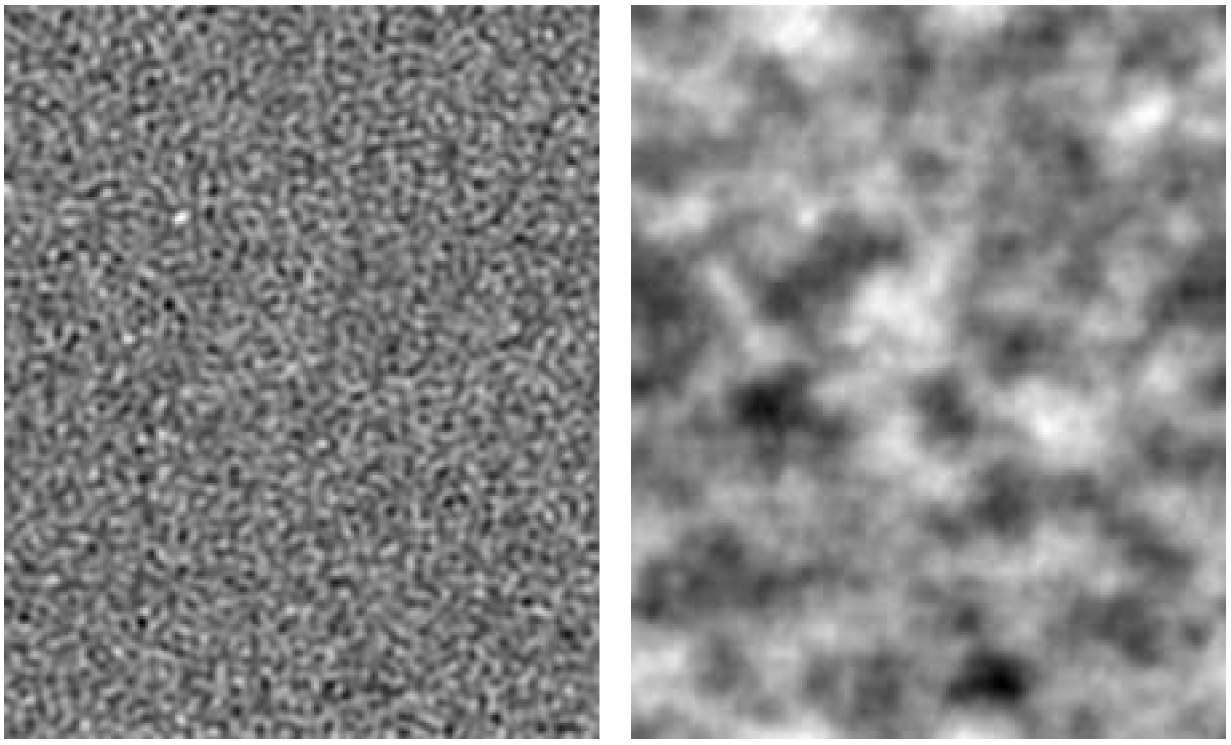
$$\Phi_{10} = F_1 \times F_0^*$$

where F_0^* is the complex conjugate of F_0. The auto-correlation terms $\Phi_{00} = F_0 \times F_0^*$ and $\Phi_{11} = F_1 \times F_1^*$ are used to enhance the cross-correlation term as follows

$$P = \frac{\Phi_{10}}{\sqrt{\Phi_{00} \times \Phi_{11}} + \epsilon}$$

where ϵ is a small number greater than zero added to avoid division by zero. The Girod and Kuo paper addresses a slightly different problem than the one targeted by our application. The technique described in the paper is intended for tracking a moving object. One of the difficulties of the tracking problem is that the background behind the object changes. The mosaicking problem typically does not suffer from this obstacle. During early experimentation we attempted to use the auto-correlation directly as $P = \Phi_{10}$. This was found to be unacceptable, therefore the current implementation of the mosaicking application follows exactly the technique described by Girod and Kuo. The comparison of the enhanced auto-correlation and plain auto-correlation can be seen in figure 1.

Figure 1: enhanced auto-correlation vs. plain auto-correlation


enhanced auto-correlation plain auto-correlation

The inverse Fourier transform of the cross-correlation

$$PDF\,(x, y) = \Re\left(F^{-1}\{P\}\right)$$

corresponds to the probability density function (PDF) that tile S_1 matches with tile S_0 displaced by vector $[x\,y]^T$. We will refer to this function as the displacement PDF. Thus, in order to find the displacement vector it is necessary to find the coordinates $[x_{max}\,y_{max}]^T$ of the global maximum of this function.

Finding the maximum of the displacement PDF is non-trivial. This is due to the fact that for most electron microscopy images the PDF is usually very noisy. Also, the PDF of two mismatched images may contain several maxima, or none at all. The technique described in the Girod and Kuo paper mentions a simple thresholding method used to suppress the negative and insignificantly small values of the PDF. The method currently implemented in the mosaicking application is similar, but has several important features that are worth pointing out.

Early experimentation with the PDF has shown that identifying the maxima becomes significantly easier after blurring the PDF to remove the high-frequency noise. The blurring is carried out in the Fourier domain, where it corresponds to a multiplication by a low-pass filter

$$PDF\left(x,y\right)=\Re\left(F^{-1}\left\{P\times Filter\left(r,s\right)\right\}\right)$$

where $r\in\left[0,\sqrt{2}\right]$ and $s\in\left[0,r\right]$. When $s=0$ the filter behaves exactly like the ideal low-pass filter, passing unaffected frequencies in the range $\left[0,r\right]$ and attenuating completely frequencies in the range $\left(r,\infty\right)$. When $s>0$ the filter passes frequencies in the range $\left[0,r-s\right]$ completely unaffected, frequencies in the range $\left(r+s,\infty\right)$ are completely attenuated, and frequencies in the range $\left(r-s,r+s\right]$ are attenuated according to the function

$$attenuation\left(f\right)=\frac{1+\cos\left(\pi\frac{f-(r-s)}{2s}\right)}{2}$$

which provides a smooth transition from zero attenuation at $f=r-s$ to full attenuation at $f=r+s$. This low-pass filter results in zero total power loss in the frequency range $\left[0,r\right]$, because the attenuation incurred in range $\left[r-s,r\right]$ is canceled out by the power leakage from range $\left[r,r+s\right]$ due to aliasing.

More experimentation has shown that blurring the tiles prior to calculating their corresponding PDF reduces the number of false maxima in the PDF. The tiles are blurred in the Fourier domain as follows

$$\begin{aligned}F_0 &= F\left\{S_0\right\}\times Filter\left(r,s\right)\\ F_1 &= F\left\{S_1\right\}\times Filter\left(r,s\right)\end{aligned}$$

and the rest of the calculations are carried out as described above. The parameters r and s used for blurring the tiles and the PDF can be tuned. In the current implementation the values $r=0.5$ and $s=0.1$ are used for the tiles, and $r=0.4$ and $s=0.1$ for the PDF.

Having blurred the PDF, it is necessary to select a good threshold value in order to isolate a set of pixels corresponding to the global PDF maximum. We assume that the number of pixels belonging to the maximum is approximately 1% of the total number of PDF pixels, but it may not be less than 5 pixels or greater than 64 pixels. The lower bound restriction is imposed in order to avoid thresholding values where only one maximum pixel is left. One pixel does not carry enough information about the rest of the structure of the PDF. When 5 pixels are grouped together, it is fairly obvious that there is only one strong maximum in the PDF. If the pixels are scattered across the PDF, it is likely the PDF does not have a strong maximum. The lower bound on the number of pixels belonging to the PDF maximum is necessary in order to deliver the information regarding the distribution of these pixels within the PDF. One or two pixels do not carry enough information. The upper bound on the number of pixels applies to larger images. If too many pixels are allocated to the PDF maxima, the computational burden involved in the classification of the clusters increases. The upper limit of 64 pixels guarantees that no PDF could ever contain more than 64 maxima. Thus

$$pixels_{maxima}=\min\left(64,\max\left(5,\frac{area\left(PDF\right)}{100}\right)\right)$$

where $area\left(PDF\right)$ corresponds to the total number of pixels in the PDF image.

To find the threshold value that would provide this number of pixels, it is necessary to build a cumulative histogram of the PDF pixel values. The current implementation uses 1024 histogram bins. Although the importance of this parameter has not been explored in the context of our application, we can assume that more bins will give us a more accurate estimate of the threshold value. The cumulative histogram is searched for the bin containing at least

$$area\left(PDF\right)-pixels_{maxima}$$

number of pixels. The minimum pixel value associated with that bin is the optimal threshold value that we need.

Once the PDF is thresholded, a small fraction of the pixels belonging to the maxima are isolated into one or more clusters. Next, pixels are classified into clusters based on an 8-connected neighborhood stencil. Once all of the clusters have been identified, the clusters that are broken up across the PDF boundary are merged together. This step is required because the Discrete Fourier Transform assumes that the signal is periodic; therefore, the PDF is also periodic. After all of the pixel clusters are identified, the coordinates of the PDF maxima are calculated as the centers of mass of the corresponding clusters. The value of each maximum is calculated as the total mass of the cluster divided by the number of pixels in that cluster. This process results in a list of several maxima with varying coordinates and values. The list is sorted in descending order, so that the highest maximum is at the head of the list.

Given a list of maxima points present in a particular PDF, a simple heuristic is applied to decide whether the tiles that produced this PDF in fact match. Matching tiles would ideally produce only one maximum. However, due to the inaccuracy in the selection of the thresholding value, it is very likely that there will be several maxima. This is also the case when the tiles being matched have undergone a distortion. During experimentation an important observation was made that mismatching tiles produce a PDF with several maxima points at roughly the same value, while the PDF of two matching tiles produces one maximum significantly higher than the rest. This result suggests a very simple algorithm to decide whether the PDF corresponds to two matching tiles. The dissimilarity of the PDF maxima with respect to the best PDF maximum is calculated as

$$dissimilarity = \frac{\max_{best}(PDF)}{\max_i(PDF)} - 1$$

The *dissimilarity* of two perfectly similar maxima is equal to 0. Whenever *dissimilarity* exceeds a given threshold the corresponding maximum is removed from the list. In current implementation, the *dissimilarity* threshold is set to 1; thus, maxima which are more than 2 times smaller than the highest maxima in the list are discarded. If the list contains only one maximum, we assume that the tiles match and proceed to calculate the corresponding displacement vector. If there is more than one maximum left in the list after this filtering, it is very likely that the tiles do not match, or one of the tiles is self-similar and may match the other tile in several places. Due to distortion, it is possible that no matching tiles will be found with exactly one maximum. In that case the match with the fewest number of maxima is considered. Significantly radially distorted tiles typically have 2 to 4 valid maxima corresponding to small shifts from the true displacement vector. The current implementation of the mosaicking application considers at most 3 maxima per match.

In order to find the displacement vector, it is not enough to simply find the maximum of the displacement PDF. The coordinates $[x_{max}\, y_{max}]^T$ are always positive, yet the displacement vector may very well have negative coordinates. As mentioned earlier, the Discrete Fourier Transform assumes that the signal is periodic, therefore the cross-correlation between the tiles corresponds to cross-correlation of two periodic tiles. Once the coordinates of the maximum $[x_{max}\, y_{max}]^T$ are known, there are four possible permutations of the displacement vector that could produce the corresponding high cross-correlation between the tiles. The permutations are

$$\begin{aligned}
T_{00} &= \begin{bmatrix} x_{max} \\ y_{max} \end{bmatrix} \\
T_{10} &= \begin{bmatrix} x_{max} - width(S_0) \\ y_{max} \end{bmatrix} \\
T_{01} &= \begin{bmatrix} x_{max} \\ y_{max} - height(S_0) \end{bmatrix} \\
T_{11} &= \begin{bmatrix} x_{max} - width(S_0) \\ y_{max} - height(S_0) \end{bmatrix}
\end{aligned}$$

The current implementation of the application chooses the best permutation based on the normalized squared image differences metric. This metric is calculated as the sum of squared pixel differences within the overlap region, divided by the area of the overlap region. The best permutation corresponds to the lowest metric value (the least mismatch between the tiles). The metric is evaluated against unpadded tiles

U_0 and U_1, yet the displacement permutations are based on the dimensions of the padded tiles S_0 and S_1, which means that some of the permutations may not overlap the unpadded tiles at all. In consequence, permutations can be discarded early based on the amount of overlap between the tiles. The amount of overlap is computed as the ratio of the area of the overlap region to the area of the smaller of the two tiles. Thus, when one tile overlaps another entirely, the overlap is equal to 1. Displacement vectors resulting in less than 5% of overlap are discarded without further consideration. This decision is based on the fact that typical tiles will have 20% to 30% of overlap along the edges of the tile, and approximately 10% to 5% of overlap at the corners.

3.2 Initial mosaic layout

Prior to deducing the tile ordering it is necessary to find pairs of matching tiles. The runtime complexity of the current algorithm for finding the matching tiles is $O\left(n^2\right)$. The performance of this algorithm may be improved, but not without sacrificing some robustness in finding the correct tile matches and rejecting the mismatches. Why this is the case will become more clear after the current algorithm is explained in greater detail.

The algorithm tries to find the best possible mapping from the image space of one tile into any other tile. This is accomplished by cascading the mappings via intermediate tiles. For example, there may exist a mapping $U_0 : U_1$ between tiles U_0 and U_1, and another mapping $U_1 : U_4$ between tiles U_1 and U_4. A mapping $U_0 : U_1 : U_4$ between tiles U_0 and U_4 can be created via the intermediate tile U_1. The number of intermediate steps in a mapping from one tile to another will be referred to as the cascade length from now on. Given n tiles, there may be at most $n-2$ intermediate steps in a mapping between any 2 tiles. Of course, this is only the upper bound on the cascade length. There are no guarantees that a mapping with a given cascade length exists between any 2 tiles. However, the fact that there may be redundant mappings between any 2 tiles presents a great opportunity to select the best mapping possible.

The algorithm proceeds as follows. First, pairs of matching tiles are found. Finding just one match for every tile is not enough, because that does not provide any redundant mappings between the tiles. This is the reason why the algorithm has $O\left(n^2\right)$ run time complexity. One way to speed up the algorithm is to limit the number of redundant mappings to some fixed maximum number per tile. Allowing a maximum of just 2 mappings per tile may introduce enough redundancy to correct for mismatches while also speeding up the matching process.

The mappings between the tiles are stored as connections in a graph of tiles. Each mapping (connection) is weighed according to the normalized squared image differences metric mentioned earlier. Next, redundant mappings with cascade length 1 to $n-2$ are found. There may be more than one such mapping, therefore it is useful if the process is explained with an example. Assume there exists a function

$$C\left(U_i : U_j\right) = cost$$

that evaluates the cost of a mapping between tiles U_i and U_j. Given the following sample mappings

$$\begin{aligned}
C\left(U_0 : U_1\right) &= 278 \\
C\left(U_0 : U_2\right) &= 311 \\
C\left(U_1 : U_4\right) &= 160 \\
C\left(U_2 : U_4\right) &= 121 \\
C\left(U_0 : U_4\right) &= 3419
\end{aligned}$$

it is most likely that the mapping $U_0 : U_4$ is mismatched. There are 2 possible alternative mapping from tile U_0 to U_4. The cost is set to the maximum cost of the intermediate mapping costs. In the context of this example, this means that

$$C\left(U_0 : U_1 : U_4\right) = \max\left(C\left(U_0 : U_1\right), C\left(U_1 : U_4\right)\right) = 278$$

$$C\left(U_0 : U_2 : U_4\right) = \max\left(C\left(U_0 : U_2\right), C\left(U_2 : U_4\right)\right) = 311$$

The mapping with the least cost (in this case $U_0 : U_1 : U_4$) is preferred even when it has greater cascade length.

In order to generate the mosaic, it is necessary to select the target tile into which every other tile will be mapped. This is done by considering the total cost of the target tile candidates. The total cost is calculated as the cumulative cost of the mapping from the target tile to every other tile in the mosaic. The candidate with the lowest total cost becomes the target tile.

3.3 Distortion correction

In order to correct for distortion each tile has to be un-warped. During the various stages of the development of this application, several transform types have been explored. The transform that is currently used is a bivariate cubic Legendre polynomial, defined as follows

$$\begin{aligned} x(u,v) &= X_{max} \times \sum_{i=0}^{N}\sum_{j=0}^{i} a_{j,i-j} \times P_j\left(\frac{u-u_c}{X_{max}}\right) \times P_{i-j}\left(\frac{v-v_c}{Y_{max}}\right) \\ y(u,v) &= Y_{max} \times \sum_{i=0}^{N}\sum_{j=0}^{i} b_{j,i-j} \times P_j\left(\frac{u-u_c}{X_{max}}\right) \times P_{i-j}\left(\frac{v-v_c}{Y_{max}}\right) \end{aligned}$$

where $[u_c \, v_c]^T$ is the center of distortion. The transform is parameterized by coordinates $u_c \, v_c$, normalization constants X_{max} and Y_{max} and polynomial coefficients $a_{i,i-j}$ and $b_{i,i-j}$. In order to simplify the computational burden, it is assumed that X_{max} and Y_{max} correspond to the half-width and half-height of the tile. The location of the center of distortion $[u_c \, v_c]^T$ is unknown, therefore it is assumed to be at the center of the tile. The gross tile displacement $[T_x \, T_y]^T$ estimated from the tile matching is incorporated in $[u_c \, v_c]^T$ as follows

$$\begin{aligned} u_c &= \frac{width(U_i)}{2} - T_x \\ v_c &= \frac{height(U_i)}{2} - T_y \end{aligned}$$

The polynomial coefficients are found iteratively by the ITK[3] optimization framework. The standard ITK image registration framework consists of the following components

- Two images that must be matched (fixed image and moving image).
- A metric that quantifies the quality of the match between the images.
- A transform.
- An optimizer.

This framework is not directly applicable to simultaneous registration of more than 2 images, therefore an alternative method had to be developed. Since more than one tile may overlap the same pixel, the average intensity variance within overlapping regions was chosen as the tile mismatch metric, shown below

$$V = \frac{1}{A}\sum_{u=0}^{W-1}\sum_{v=0}^{H-1}\left(\frac{1}{N(u,v)}\sum_{i=0}^{N(u,v)-1} P_i\left(x_i\left(u,v,a_{i;0,0},...\right), y_i\left(u,v,b_{i;0,0},...\right)\right) - \mu(u,v)\right)$$

where V is the average variance. A is the area (pixel count) of the overlapping regions. W and H are the dimensions of the mosaic. $N(u,v)$ is the number of tiles overlapping a pixel at the given mosaic coordinates $[u,v]^T$. $x_i(u,v,a_{i;0,0},...)$ and $y_i(u,v,b_{i;0,0},...)$ compute the tile coordinates given mosaic coordinates $[u,v]^T$ and transform parameters $a_{i;0,0},...,b_{i;0,0},...$ for tile U_i. $P_i(x,y)$ is the intensity value for tile U_i at the computed tile coordinates, and $\mu(u,v)$ is the mean intensity value at the specified mosaic coordinates

$$\mu(u,v) = \frac{1}{N(u,v)}\sum_{j=0}^{N(u,v)-1} P_j\left(x\left(u,v,a_{0,0},...\right), y\left(u,v,b_{0,0},...\right)\right)$$

Thus, a transform that maps from the mosaic space into the tile space is computed for each image. In order to estimate the bounding box of the mosaic, a transform from the tile space to the mosaic space must be used. Since the corresponding transform is unavailable, the inverse mapping is calculated numerically via the Newton's method[2].

Within the ITK image optimization framework, the optimizer manipulates the parameters of the transform for each tile in order to minimize the average variance within the overlapping regions of the mosaic. Currently, we use a modified version of the *itk::RegularStepGradientDescentOptimizer*, where the relaxation criteria has been altered to be independent of the derivative direction to rely solely on the function value. The original ITK implementation of *itk::RegularStepGradientDescentOptimizer* diverged near the minima of the metric function.

The optimization proceeds in 2 stages. First, we assume that all of the tiles have been warped similarly, therefore optimize all transform parameters of (except the fixed parameters $u_c, v_c, X_{max}, Y_{max}$) of one tile and share the changes with all other tile transforms. This compensates for large scale radial distortion common to all tiles. Next, we assume that the remaining variance in the mosaic is due to unique distortions present in each tile. Therefore, we restart the optimization with the shared parameters. This time we optimize each tiles transform without sharing the parameters with other tiles. This produces the unique transform parameters for each tile.

The variance minimization iterates until it converges or exceeds the maximum number of iterations (specified by the user). The resulting transform parameters define the un-distortion transforms which best match the neighboring tiles.

4 Demonstration of the correctness of implementation

The tile matching and tile ordering examples were computed using an earlier version of the un-warping transform defined as follows

$$\begin{aligned}
x(u,v) &= u_c + (u-u_c) \times S(u,v) \\
y(u,v) &= v_c + (v-v_c) \times S(u,v) \\
S(u,v) &= \sum_{n=0}^{N-1} k_n \times \left(\frac{R(u,v)}{R_{max}}\right)^{2n} \\
R(u,v) &= \sqrt{(u-u_c)^2 + (v-v_c)^2}
\end{aligned}$$

where $[u_c\, v_c]^T$ is the center of radial distortion. The transform is normalized by R_{max}. Thus, the radial distortion transform is parameterized by coordinates $u_c\, v_c$, normalization constant R_{max} and polynomial coefficients $k_0 ... k_{N-1}$. In order to simplify the computational burden, it is assumed that R_{max} corresponds to the maximum distance from the center of distortion to the corners of the tile. The location of the center of distortion is unknown, therefore it is assumed to be at the center of the tile. Additionally, the number of polynomial coefficients is limited to $N = 2$. Thus, only k_0 and k_1 are needed to define the transform.

4.1 Tile matching

Figure 2 on the next page shows two matching image tiles. These tiles have undergone a mild radial distortion with parameters $k_0 = 0.95$ and $k_1 = 0.05$. The overlap area between these tiles is roughly 8%. Figure 3 shows the displacement PDF corresponding to these two tiles, as well as the isolated pixel clusters corresponding to the PDF maxima. There are a total of 19 maxima isolated in the PDF. Filtering the maxima leaves only one eligible maximum for consideration, which indicates that the tiles are well matched.

Figure 2: matching tiles

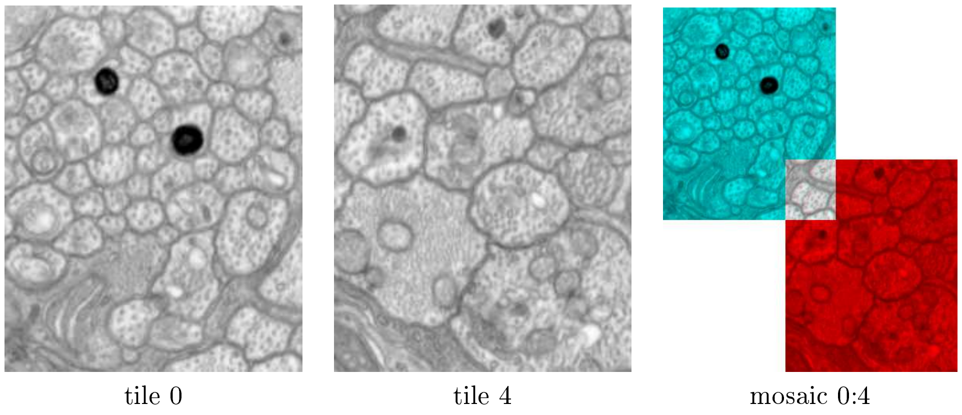


Figure 3: displacement PDF for matching tiles

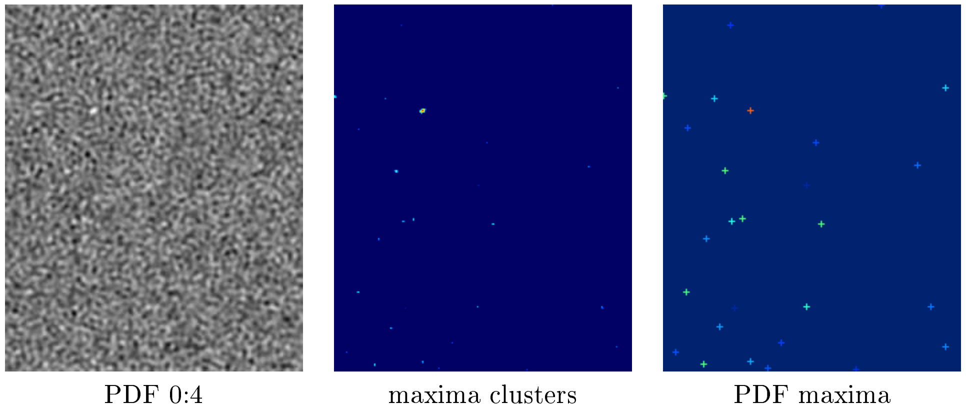
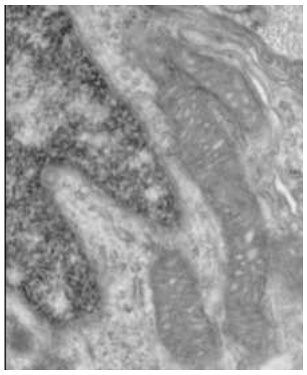


Figure 4 on the following page shows two mismatched tiles. Figure 5 shows the corresponding displacement PDF and PDF maxima. There are 30 maxima isolated in this PDF. After filtering there are still 5 maxima left. Ideally there would be only one maximum left, therefore this PDF indicates that the tiles do not match.

Figure 4: mismatched tiles


tile 6

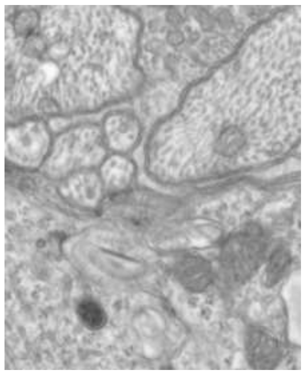
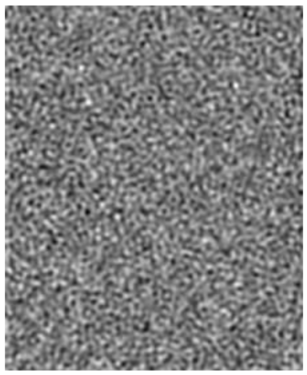
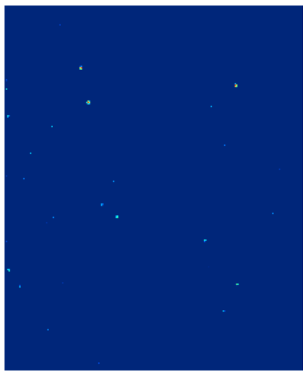
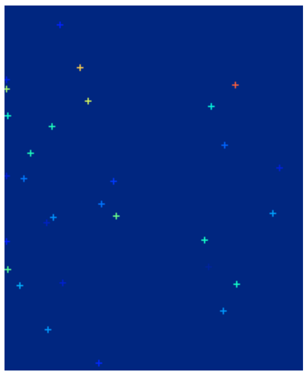

tile 8

Figure 5: displacement PDF for mismatched tiles


PDF 6:8

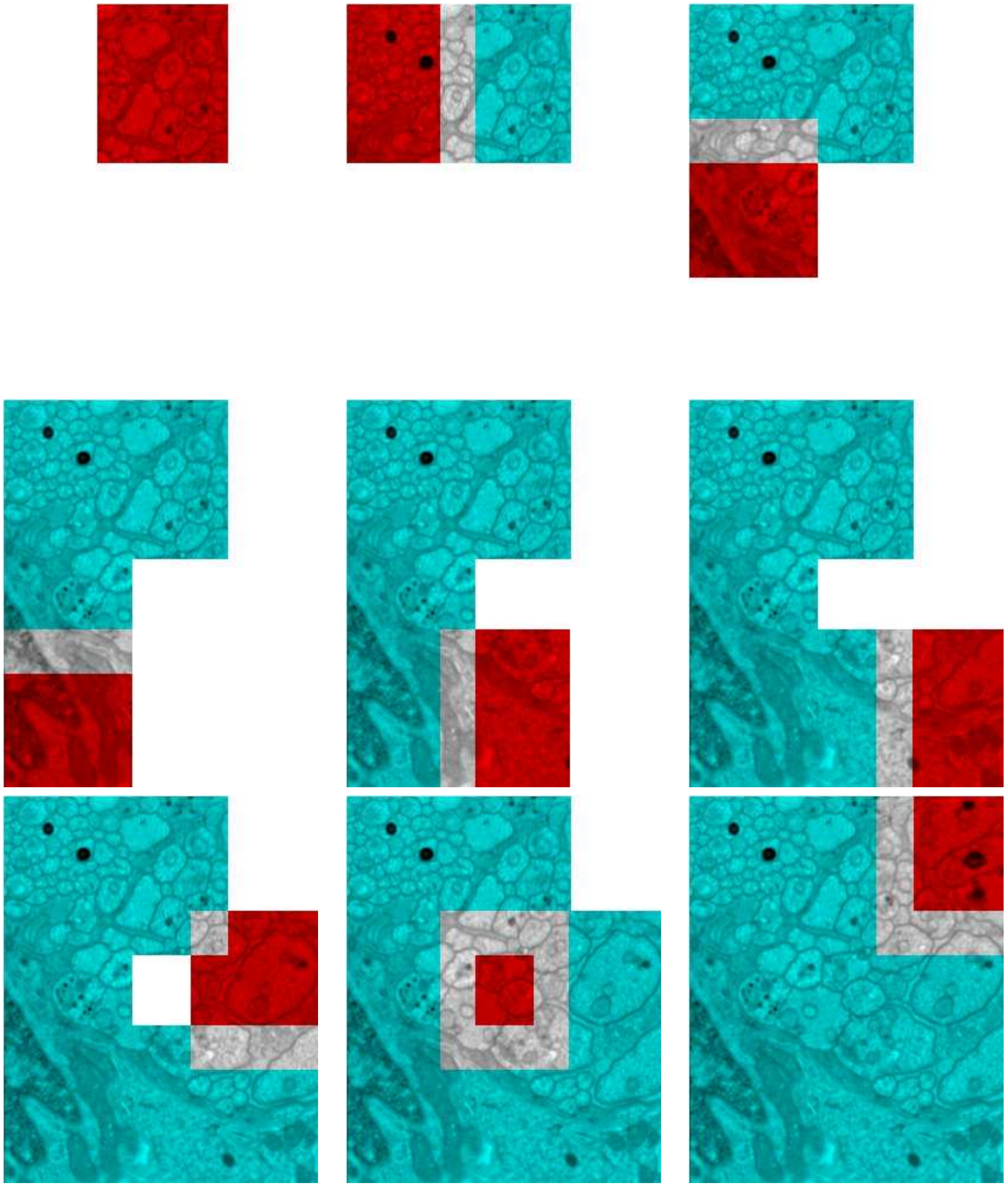

maxima clusters


PDF maxima

4.2 Tile ordering

Figure 6 on the next page illustrates the order in which the tiles are added to the mosaic. As can be seen, the algorithm lays out the red tiles such that they have significant overlap with previous tiles (shown in blue).

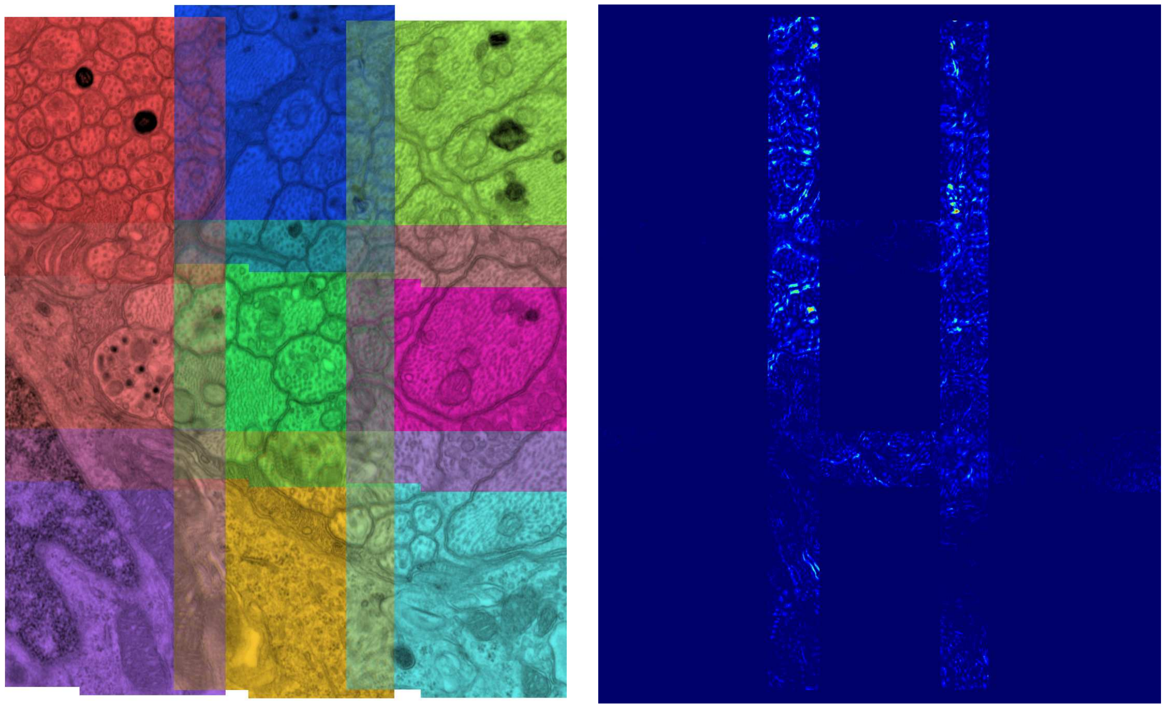
Figure 6: tile ordering



4.3 Distortion correction

To verify the un-warping capabilities of the application a set of 9 artificially warped tiles was generated. Each tile was warped by a radial distortion transform with parameters $k_0 = 0.95 \pm 0.05 \times drand()$ and $k_1 = 0.05 \pm 0.05 \times drand()$. This ensures that each tile has been uniquely warped. Figure 7 on the following page shows the result of displacement estimation for each tile, as well as the variance within the overlapping regions of the mosaic.

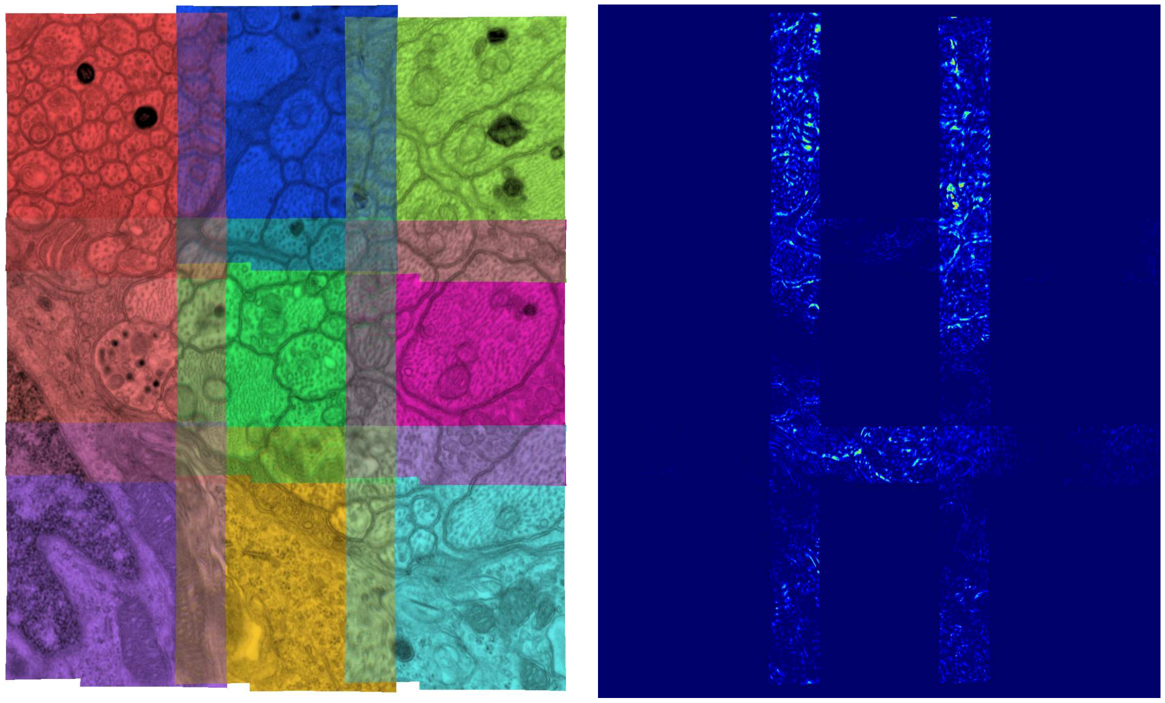
Figure 7: initial mosaic



This figure illustrates the tiling of the mosaic and initial variance within the overlapping regions of the mosaic. Here, the maximum variance is 7750, and the mean variance is 144.

The initial mosaic is first un-warped using shared transform parameters across all transforms. This is meant to compensate for any common global distortion present in all tiles. This stage of un-warping reduces the average variance from 144 to 112, as illustrated in figure 8 on the next page. The image on the bottom demonstrates variance within the overlapping regions of the mosaic.

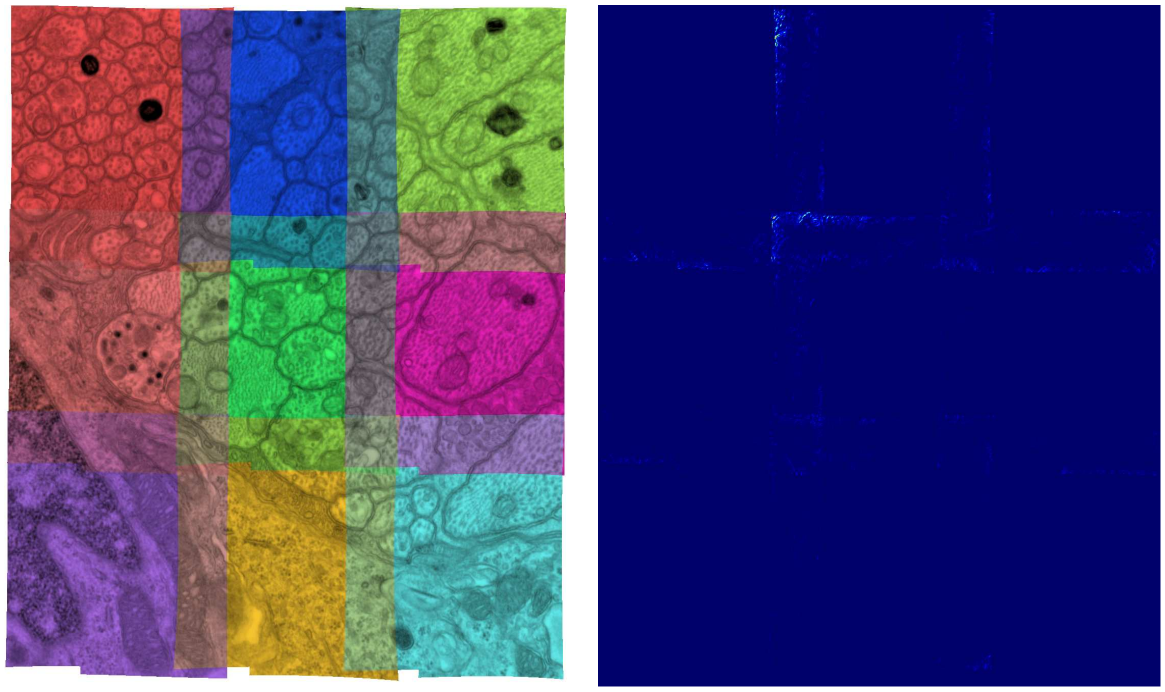
Figure 8: shared parameters optimization results



The result of shared transform parameters optimization. Here, the maximum variance is 4940, and the mean variance is 112.

Following the optimization using the shared transform parameters, the process is repeated with unique transform parameters for each tile. This stage of un-warping reduces the average variance from 112 to 2.71, as illustrated in Figure 9 on the following page .

Figure 9: unique parameters optimization results

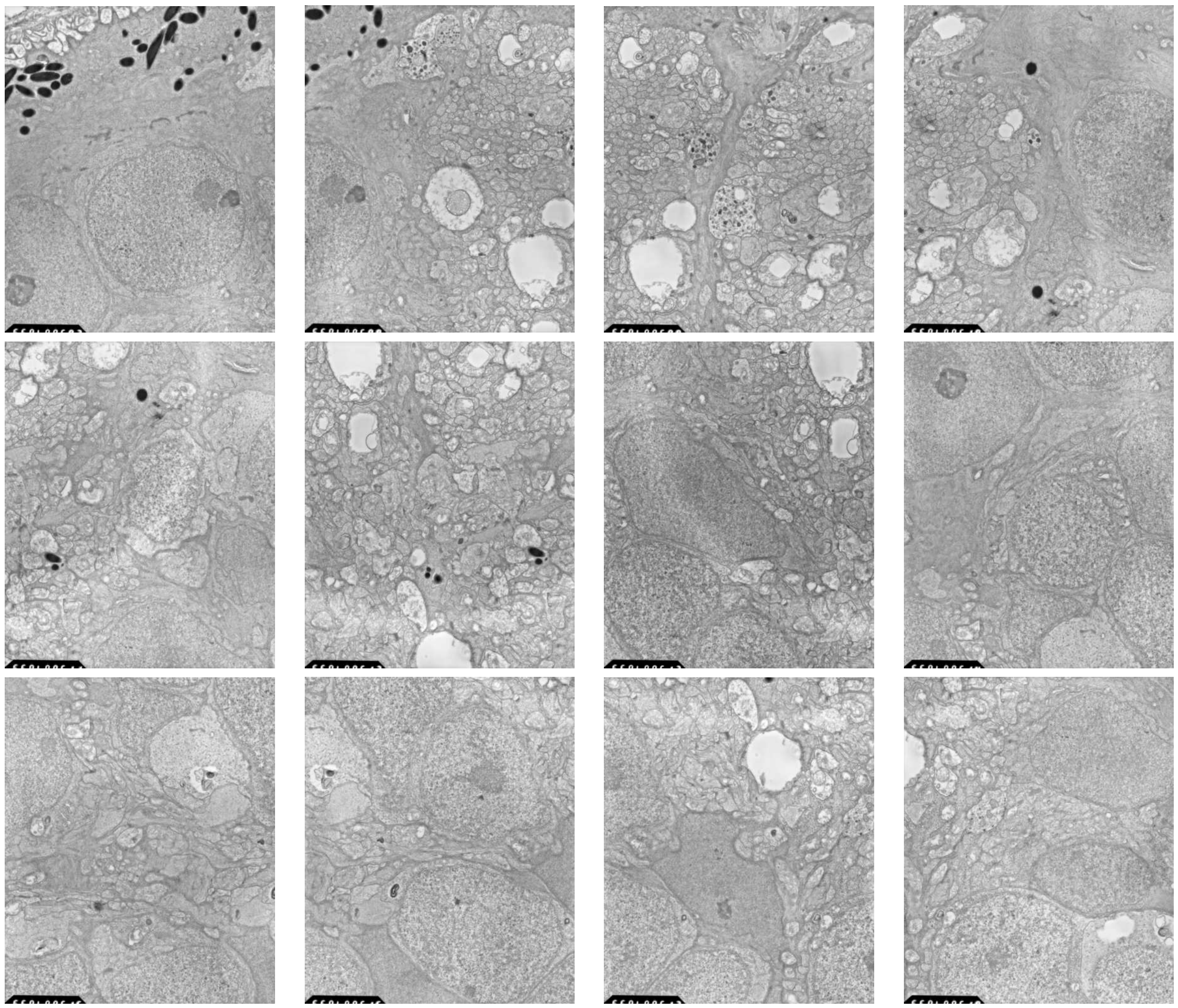


The result of unique transform parameters optimization. Here, the maximum variance is 80.7, and the mean variance is 2.71.

5 Results

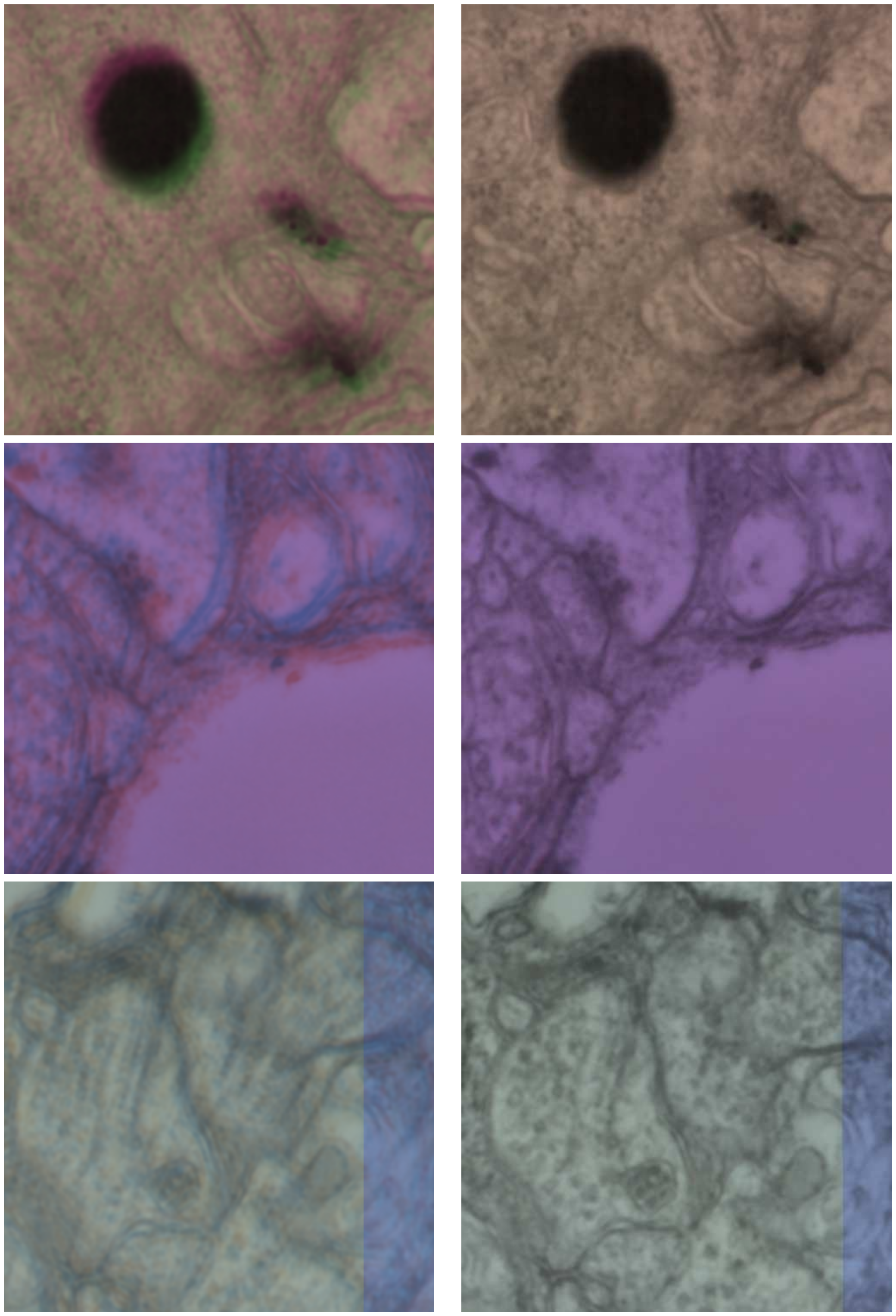
Figure 10 on the next page shows 12 tiles of one mosaic. These tiles were matched to each other resulting in initial mean variance of 100. Following the shared transform parameters optimization, the mean variance was reduced down to 52.8. The unique transform parameters optimization reduced the mean variance down to 43. The remaining variance may be due to higher order distortion, or differences in tile illumination inherent in each tile or contributed by the Contrast Limited Adaptive Histogram Equalization (CLAHE) preprocessing that was applied to each tile. A close up demonstration of the achieved variance reduction can be seen in figure 11 on page 15.

Figure 10: sample electron microscopy tiles



These are the sample Transmission Electron Microscopy tiles from one slice of a rabbit retina tissue. Each tile has been enhanced with Contrast Limited Adaptive Histogram Equalization (CLAHE) algorithm. The tag at the lower left corner of each tile disrupts the initial tile matching in the frequency domain, therefore a bottom portion of the image containing the tag has to be cropped out prior to transforming the image via FFT. During variance minimization the tag is masked out, leaving the rest of the image in tact. The effect of masking out the tag can be seen in the mosaics shown in figures 7, 8 and 9.

Figure 11: variance reduction



These images illustrate the variance reduction due to tile un-warping within the overlap regions of the mosaic. The images on the left are from the initial mosaic prior to un-warping, while the images on the right are from the final mosaic where each tile has been un-warped with unique transform parameters.

References

[1] Girod, B. and Kuo, D. 1989. Direct estimation of displacement histograms. In Proceedings of the Optical Society of America Meeting on Understanding and Machine Vision, 73–76.

[2] Newton-Raphson Method for Nonlinear Systems of Equations. Numerical Recipes in C, second edition, 379–382.

[3] NLM Insight Segmentation & Registration Toolkit, http://www.itk.org/

[4] Fastest Fourier Transform in the West, http://www.fftw.org/