Fast Stitching of Large 3D Biological Datasets

Stephan Preibisch, Stephan Saalfeld and Pavel Tomancak

Max Planck Institute of Molecular Cell Biology and Genetics, Dresden, Germany

ABSTRACT

In order to study anatomy of organisms with high-resolution there is an increasing demand to image large specimen in three dimensions (3D). Confocal microscopy is able to produce high-resolution 3D images, but these are limited by relatively small field of view compared to the size of large biological specimens. To overcome this drawback, motorized stages moving the sample are used to create a tiled scan of the whole specimen. The physical coordinates provided by the microscope stage are not precise enough to allow reconstruction (”Stitching”) of the whole image from individual image stacks.

We developed an algorithm, as well as an ImageJ plug-in, based on the Fourier Shift Theorem that computes all possible translations (x, y, z) between two 3D images at once, yielding the best overlap in terms of the cross correlation measure. Apart from the obvious gain in computation time it has the advantage that it cannot be trapped in local minima as it simply computes all possible solutions. Computing the overlap between two adjacent image stacks is fast (12 seconds for two 512x512x89 images on a Intel® Core2Duo with 2.2 GHz) making it suitable for real time use, i.e. computing the output image during acquisition of the individual image stacks. To compensate the possible shading- and brightness differences we apply a smooth linear intensity transition between the overlapping stacks. Additionally we extended the approach to generic 3D registration using gradient based rotation detection in combination with the phase correlation method. We demonstrate the performance of our 3D stitching plug-in on several tiled confocal images and show an example of its application for 3D registration.

Keywords: 3D Stitching, 3D Registration, Fourier Analysis, Global Optimization

1. INTRODUCTION

There is an increasing demand to image complete biological specimen with high resolution in two and three dimensions (2D, 3D). Biologist use Confocal(3D), Widefield(2D) or Electron(2D) microscopes equipped with motorized stages to stepwise image large areas using high magnifications. The acquired image tiles have to be combined into one final output image by a process usually referred to as *Stitching*. Motorized stages used for such acquisitions operate with high precision, however, in many cases the physical coordinates from the stage are not precise enough to provide a proper reconstruction of the whole acquisition on pixel level.

The physical coordinates provided by the microscope telemetry are imprecise due to the limited precision of the motorized stages and the nondeterministic shaking of the stage during acquisition. With our microscope hardware we observed changes up to 10% from the anticipated position of an image tile. To overcome those problems we present an ImageJ[1] plug-in capable of stitching two and three dimensional images starting from approximate initial configuration. The plug-in uses a *phase correlation* based algorithm to find optimal registration parameter for each pair of stacks and global optimization to stitch multi-tile acquisitions. Additionally we implemented linear blending to compensate brightness changes between the image tiles.

This paper is organized as follows. We will first describe the algorithm for stitching two adjacent tiles. In the following chapter we extend the algorithms to the stitching of many tiles and describe the linear blending. Afterwards we present the ImageJ plug-in and finish with the summary and outlook of further applications of phase correlation.

Further author information:
Stephan Preibisch: E-mail: preibisch@mpi-cbg.de, Telephone: +49 351 210 2758

2. ALGORITHM

2.1 Problem Description

In the generic case, the acquisition results in set of n image tiles where a subset overlaps so that all images form a connected graph. However, in most cases the tiles are arranged in a $p \times q$ grid and overlap by a 10–30%. The user provides an initial arrangement of the image tiles, i.e. which image tiles overlap and the algorithm determines the correct overlap for all tiles and creates a properly registered output image containing all image tiles that are connected.

We assume all transformations between overlapping tiles to be translation only. This approximation is reasonable for most commonly used microscopy techniques where the static, fixed sample rests on a level microscope slide. Translation only transformation models have the advantage that they have a closed form solution, that means no iterative optimization methods are necessary to compute all possible translations in n dimensions.

2.2 Phase Correlation

The phase correlation method (PCM)[2] is a popular algorithm to compute translational offsets between images. It is based on the well known Fourier shift theorem which states that a shift between two images results in a linear phase difference in the Fourier transform of both images. If two images A and B are related by translation the shift (x, y, z) can be computed from the *phase correlation matrix* Q of both images.

$$Q(k,l,m) = \frac{B(k,l,m) \cdot A^*(k,l,m)}{|B(k,l,m) \cdot A^*(k,l,m)|} \quad {}^* \text{ denotes complex conjugate}$$

Note that (x, y, z) refers to positions in real space while (k, l, m) defines coordinates in frequency space. The inverse Fourier transform of $Q(k, l, m)$ yields $Q(x, y, z)$ which is a delta function $\delta(x, y, z)$ in the ideal case, i.e. a single peak whose position defines the shift between the images.

In real images, however, $Q(x, y, z)$ contains several peaks marking different translations with high correlation. Moreover, each peak describes eight different possible translations (in 3D) due to the periodicity of the Fourier space. For example in two $100 \times 100 \times 100$ images a detected shift of $(x, y, z) = (10, 10, -10)$ refers to one of the following translations: $(10, 10, -10)$; $(-90, 10, -10)$; $(10, -90, -10)$; $(-90, -90, -10)$; $(10, 10, 90)$; $(-90, 10, 90)$; $(10, -90, 90)$; $(-90, -90, 90)$.

To determine the correct shift we select the n highest local maxima ($3 \times 3 \times 3$ neighborhood) from the inverse phase correlation matrix and evaluate their 8 possible meanings using the correlation coefficient r_{AB} of the cross correlation on the overlapping area of the images A, B.

$$r_{AB} = \frac{Cov(A,B)}{\sqrt{Var(A)} \cdot \sqrt{Var(B)}} = \frac{\frac{1}{xyz} \sum_x \sum_y \sum_z \left(A(x,y,z) - \overline{A}\right) \cdot \left(B(x,y,z) - \overline{B}\right)}{\sqrt{\frac{1}{xyz} \sum_x \sum_y \sum_z \left(A(x,y,z) - \overline{A}\right)} \cdot \sqrt{\frac{1}{xyz} \sum_x \sum_y \sum_z \left(B(x,y,z) - \overline{B}\right)}}$$

The correlation coefficient gives a normalized value between -1 and 1 describing their similarity. The peak with the highest r_{AB} is selected as translation between the two images. If none of the r_{AB} is above a certain limit (e.g. 0.3) the tiles are assumed to be non overlapping.

2.3 Fast Fourier Transform

The Fast Fourier Transform (FFT) is computed using the Prime Factor Fast Fourier Transform (PFFFT)[3] implemented in the Java Mines Toolkit (JTK).[4] It supports a wider variety of input sizes and is faster than typical Radix-2 FFT algorithms. As JTK implements only one dimensional PFFFT, we added multithreaded methods for 2- and 3-dimensional Fourier transforms.

Prior to performing the PFFFT the input images have to be extended to a size supported by that algorithm. However, a simple zero-padding is not optimal as it produces sharp edges between the input image and the zero-padded part (Fig. 1 rightmost image). These edges cause high correlation independent of the image content making the detection of all translations impossible.

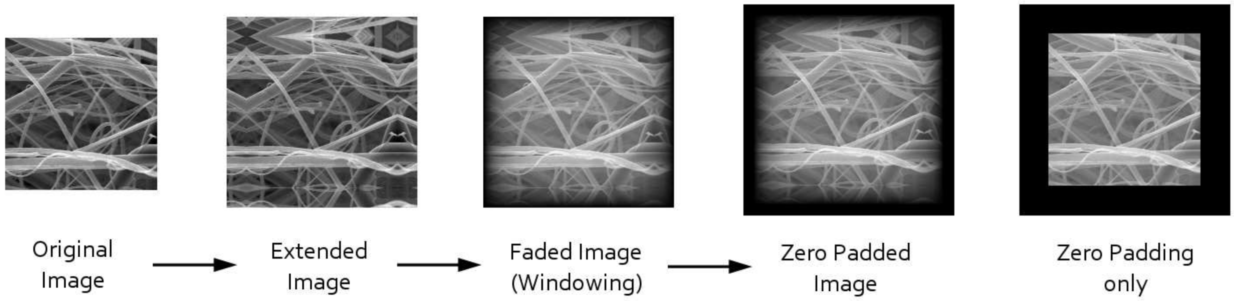

Figure 1. shows the process of extending an image to a size accepted by the PFFFT. Hard edges as shown in Zero Padding only are prevented and the image content is preserved.

To overcome those issues as well as the problem of periodicity of the Fourier Transform (it assumes that the function continues from right to left and bottom to top) the images are first extended by 1/4 of their size. The empty space is then filled by the mirrored content of the images and afterwards faded from their real intensity to black using an exponential windowing function (Fig. 1). In this way, the periodicity is not broken and real image information is preserved as only the mirrored parts are faded out. The resulting images are zero-padded to dimensions supported by the PFFFT algorithm without introducing any artifacts.

2.4 Global Optimization

After the correct alignment between overlapping image pairs has been identified the location of each image tile in the final output image has to be determined. To compute a location for an image tile it must have at least one overlapping neighbor, however, typically an image tile has up to eight neighbors (Fig. 4). Such configurations require a global optimization which ensures that possible alignment errors are not propagated. Furthermore a global optimization can deal with cases where no translation between overlapping images was found but these extant images are still connected indirectly over other images. This situation occurs when for instance the overlapping area in both images is uniform (e.g. black), a tile is missing or the correlation coefficient for two overlapping tiles is below a certain threshold (0.3).

To solve the global optimization we use the generic groupwise landmark registration that has been formalized and implemented in the diploma thesis of Stephan Saalfeld.[5] It computes the final location of each image and the minimal, average and maximal displacement of all tiles relative to the initially computed pairwise alignments. In the ideal case the displacements are zero, for real images displacements in the range of 1 pixel are typical (Table 1). To identify significant misalignments which occur if similar structures in overlapping tiles are actually not the same we use an approach inspired by robust regression. We detect outliers by checking the ratio between maximal and average displacement. As long as the maximal displacement is more than 2.5 times higher than the average displacement the alignment of the image pair with the highest displacement is removed and the optimization is re-calculated. If tiles become disconnected from all their neighbors they will not contribute to the output image. These are usually tiles without content.

2.5 Output Image Computation using Linear Blending

Almost all microscopic acquisitions suffer from shading. Shading is an position dependent additive term to the pixel intensity, in microscopy also often called background image. In most cases this results in brightness differences in the overlapping areas which are clearly visible as lines (Fig. 3) even if the registration is correct.

We compute the linear blending in the overlapping area by computing a weightening factor for each contributing tile T_i at each location which is based on the distance to its own border. The closer the location of a pixel $l_{i,dim}$ in an overlapping area is to its respective image border, the smaller is its contribution to the overall intensity. We start computing the minimal distance to the image border in each dimension

$d_{i,dim}^{min} = min(l_{i,dim}, size(T_{i,dim}) - l_{i,dim})$. The initial weightening factor is computed as $v(T_i) = \prod_{dim} d_{i,dim}^{min}$ and the final weights $w(T_i)$ for each tile at that location are given by:

$$w(T_i) = v(T_i)^\alpha \quad \text{and} \quad \sum_i w(T_i) = 1$$

The parameter α influences how the intensities are blended in the overlapping area (Fig. 2). With higher values of α the steepness of the blending increases. For $\alpha < 1$ edges remain visible but are less prominent, $\alpha = 0$ is identical to averaging the pixel intensities. For all images shown in this publication we used $\alpha = 1.5$.

3. IMAGEJ PLUG-IN

The algorithm described in this paper is implemented as several ImageJ plug-ins available for download on the homepage `http://fly.mpi-cbg.de/~preibisch/software.html` together with a HowTo.

The plug-ins *Stitching2D* and *Stitching3D* align two currently opened images. The user can define rectangular ROIs in the input images which mark the area used to compute the alignment. The ROI is an optional parameter that decreases the computation time. The plug-ins will create an output image using the fusion method defined by the user. Note that these plug-ins support creation of an output image as two channel overlap (red/cyan) to visualize the alignment quality. When only two images are stitched, global optimization is unnecessary. The

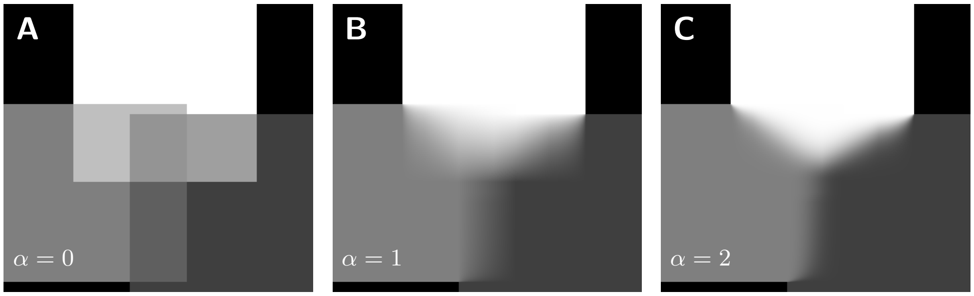

Figure 2. illustrates the results of linear blending and the effect of α. For illustration purposes each panel contains three images, one in white, gray and dark gray. (A) shows the resulting image for $\alpha = 0$ which is identical to a simple averaging in the overlapping areas. (B) and (C) show the output image when using linear blending and an α of 1 and 2, respectively.

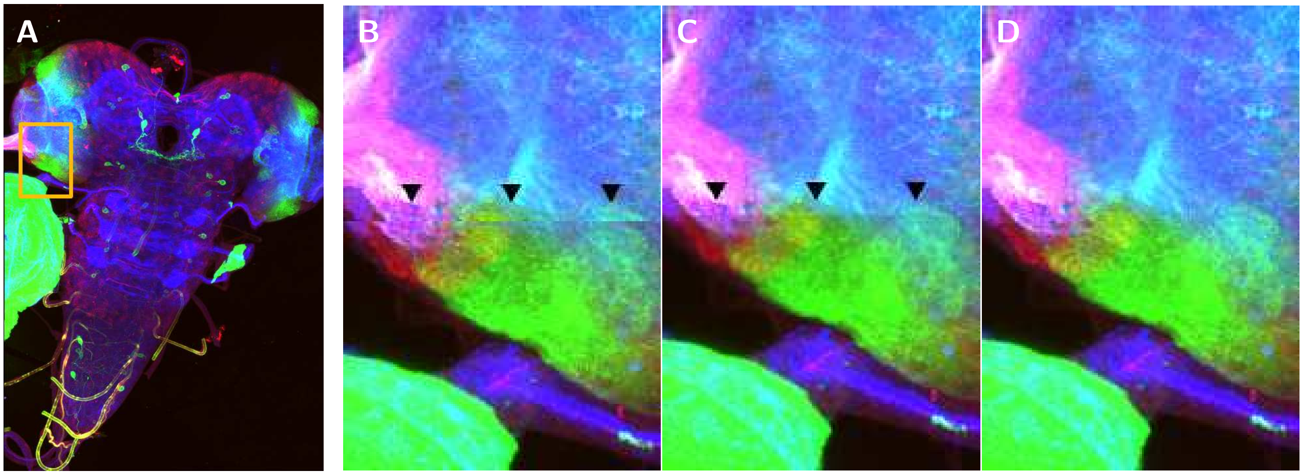

Figure 3. shows effect of linear blending on the output image. (A) Maximum intensity projection of a stitched image consisting of 6 tiles. The orange rectangle shows the area B,C,D are zoomed into. (B) shows the stitching based on coordinates provided by the stage. (C) shows the alignment using PCM without linear blending, whereas in D the linear blending was applied.

tiles	individual tile dimension	output image dimension	output image size	computation time	min/avg/max displacement
3	$1024 \times 1024 \times 42$	$1097 \times 2345 \times 43$	108 MB (8 Bit)	0 : 42 min	0.00 / 0.00 / 0.00 px
6	$512 \times 512 \times 86$	$975 \times 1425 \times 86$	350 MB (RGB)	1 : 20 min	0.60 / 0.77 / 1.05 px
24	$1024 \times 1024 \times 68$	$3570 \times 5211 \times 70$	1200 MB (8 Bit)	22 : 43 min	0.49 / 0.76 / 0.99 px
72	$512 \times 512 \times 122$	$3391 \times 3847 \times 145$	1850 MB (8 Bit)	43 : 10 min	0.00 / 0.39 / 0.64 px
63	$1024 \times 1024 \times 92$	$6088 \times 7667 \times 119$	5424 MB (8 Bit)	178 : 57 min	0.00 / 0.66 / 1.18 px

Table 1. examples of stitched data computed on an Intel® Quad-Core CPU machine with 2.67 GHz and 24 GB of RAM. The stitched images of row 1,2 and 3 are shown in figure 4. The global alignments of all stitchings have an average error below 1 px, the displacements in row 1 are zero because the two alignments are independent of each other. Note that the computation time roughly scales linear with the output image size.

plug-in *Stitch Collection of Images* stitches an arbitrary collection of input images. The configuration of how the input images are arranged has to be provided in a text file. The individual tile files are loaded using the LOCI bioformats importer (`http://www.loci.wisc.edu/ome/formats.html`) for microscopic files. The plugin computes the alignments between all overlapping images and uses global optimization to compute the output image. The related plugin *Stitch Grid of Images* creates automatically the configuration for an arbitrary grid of images without need for user defined text file. Typically the memory consumption of the plug-ins are in the range of 2-3× the size of the output image. It can be minimized by setting smaller ROIs or overlapping area and choosing maximum projection as fusion method.

4. RESULTS

We used the presented algorithm to reconstruct different tiled acquisitions from confocal microscopes (Table 1, Fig. 4). The images show visually no stitching artifacts, neither from misalignments nor from intensity leaps in the overlapping areas. The global optimization finishes with low average and maximal displacement and does not discard any non-uniform image tiles. The computation time is rather fast, small collections of 3D images can be reconstructed in less than a minute. Registration represents typically 25% from the overall computation time while the remaining 75% are spent on the creation of the output image. This delay is due to the linear blending computation which is subject to further optimization. Due to the ability of the program to run fully multithreaded, computation time will almost linearly decrease with future generations of multi-core CPUs. In general the time needed for computing the output image is much shorter than it actually takes to acquire the stacks. For example the acquisition of 24 tiles (Table 1, third row) took roughly 2 hours compared to 23 minutes of computation time.

4.1 Other applications

The PCM has the advantage over gradient based alignment methods that it cannot be trapped in local minima as it simply computes all possibilities. It is, however, limited to the detection of translational shifts. (Note: there are approaches which try to extend Fourier based methods to rotational detection in 3D[6, 7]). Therefore we extended the PCM to generic 3D registration using gradient based rotation detection where we solve the rotations using an iterative scheme while the translations are computed using PCM.[8] In cases where the initial guess is not close to the solution and/or the correct alignment is a very small local minima we found that the combination of PCM and iterative rotation detection performs better then purely iterative approaches. We are currently developing a general plug-in for ImageJ using this technique which will be released soon.

5. SUMMARY & OUTLOOK

In this paper we present an algorithm and an implementation for ImageJ capable of fast and reliable stitching of huge sets of 3D image stacks. We applied it successfully to several different types of biological samples acquired with different microscopic systems. Although not shown in this paper this approach is also practically applicable

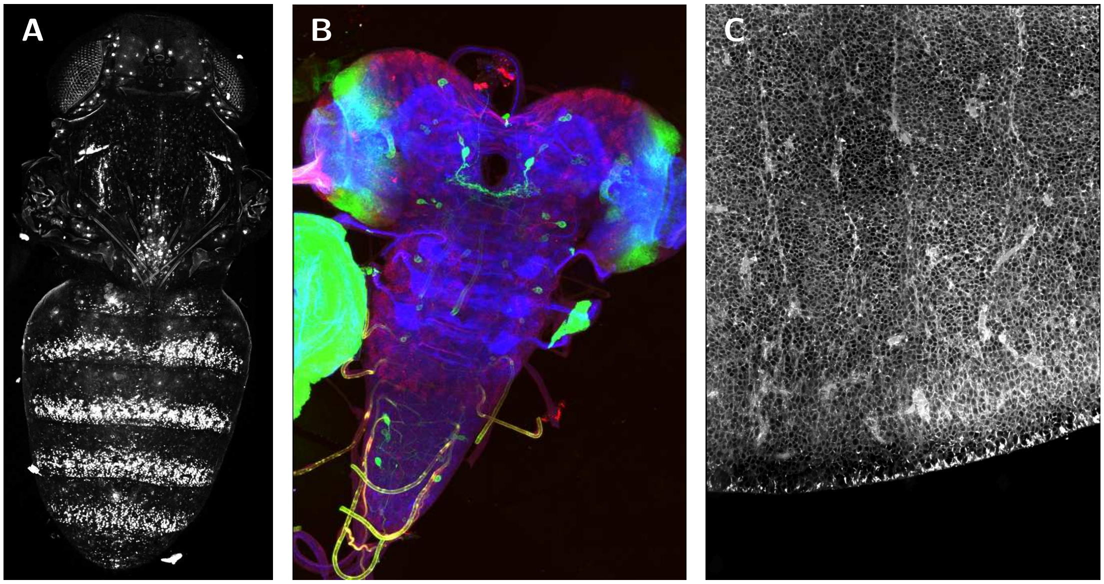

Figure 4. shows stitched images of 3D confocal tiles. (A) shows a *Drosophila melanogaster* pupae expressing a GFP reporter under the regulation of the yellow gene, imaged few hours before eclosion using a 4× dry lens on an Optiphot confocal microscope (Nikon). It was stitched from three image stacks arranged in a 1×3 grid (Table 1 first row). The maximum intensity projection is shown. (B) shows the Drosophila larval nervous system stained with three dies, stitched from a grid of 2×3 RGB images (see table 1 second row), the maximum intensity projection is shown. (C) shows a zone in the dorsal telencephalon of human embryonic tissue from week 17 post conception, incubated for 24 hours at 37°C in DiI. It was imaged using a 63×/1.4 objective on Zeiss LSM 510 equipped with a motorized stage. The final image was created from 24 image stacks arranged in a 4×6 grid (see table 1 third row), slice 18 is shown.

to 2-dimensional and theoretically even to n-dimensional datasets. Future work comprises further optimization of the source code as well as the development of a more user friendly gui for the input configuration of the images.

ACKNOWLEDGMENTS

We want to thank Arnim Jenett, Albert Cardona, Daniel White, Greg Jefferis and Nicolas Gompel for inspiring this work. We also thank Simone Fietz and Wieland B. Huttner, James W. Truman, Nicolas Gompel and Benjamin Prud'homme for providing unpublished images.

REFERENCES

[1] Rasband, W., "ImageJ: Image processing and analysis in Java [version 1.40g]," (1997–2008).

[2] Kuglin, C. D. and Hines, D. C., "The phase correlation image alignment method," in [*International Conference on Cybernetics and Society*], *Proc. IEEE*, 163–165 (1975).

[3] Good, I. J., "The interaction algorithm and practical fourier analysis," *J. R. Statist. Soc. B* **20 (2)**, 361–372 (1958).

[4] Hale, D., "Mines java toolkit: Java packages for scientific computing," (2006). `http://www.mines.edu/~dhale/jtk/index.html`.

[5] Saalfeld, S., *Automatic inter-slice registration of tiled Transmission Electron Microscopy (TEM) images*, Diploma thesis, Technische Universität Dresden (2008).

[6] Kovacs, J. A., Chacn, P., Cong, Y., Metwally, E., and Wriggers, W., "Fast rotational matching of rigid bodies by fast fourier transform acceleration of five degrees of freedom," *Acta Cryst.* **D59**, 1371–1376 (2003).

[7] Welling, J. S., Eddy, W. F., and Young, T. K., "Rotation of 3d volumes by fourier-interpolated shears," *Graphical Models* **68 (4)**, 1371–1376 (2006).

[8] Preibisch, S., Ejsmont, R., Rohlfing, T., and Tomancak, P., "Towards digital representation of drosophila embryogenesis," *Proceedings of 5th IEEE International Symposium on Biomedical Imaging* , 324–327 (2008).