

# ROBUST GENERAL MOSAIC RECONSTRUCTION OF MICROSCOPIC IMAGES

Olafur Arthursson  
Can Gencer  
Johan Henriksson

March 15, 2004

# 1 Introduction

Image registration or matching is a process of aligning two or more images, which might be taken from different viewpoints. It has wide areas of applications, such as medical imaging, satellite photography, post-production video effects and video compression. In our case, we are interested in aligning images from a microscope.

Putting a digital camera on a microscope opens a lot of possibilities for efficient imaging. However, there are situations when the resolution of a single image is not enough, or when a large specimen cannot be captured in a single image even at low optical magnification. Therefore, there is a need for constructing a large image from a number of tiles, i.e. partial images. Commercial software for mosaic reconstruction typically require that partial images are taken in a known pattern, usually  $n \times m$  images arranged in a grid. In the absence of a motorized microscope, this is typically not what the partial images look like.



Figure 1: Assembled images

## 2 The Problem

We have a number of partial images acquired with a microscope. In our case, the number of such images varies from 2 to about 15. We can only make a limited number of assumptions about the images:

- They are not taken in a known order.
- They may not form a solid, filled area.

- They may have different amount of overlap.
- The image content may vary a lot.
- They are focused and have fair contrast.
- They contain some amount of at least manually identifiable structural features.
- The images have the same orientation, i.e. there is no need for rotation.
- The images have the same size.

Our goal is to develop and implement an algorithm that will robustly arrange a number of spatially unconstrained partial images into a whole image. The priority of the algorithm should be robustness rather than speed. We define robustness by the algorithm's ability to really find a correct solution and identifying it as such or clearly identifying the lack of a valid solution. The motivation for robustness is to make the program be able run unsupervised.

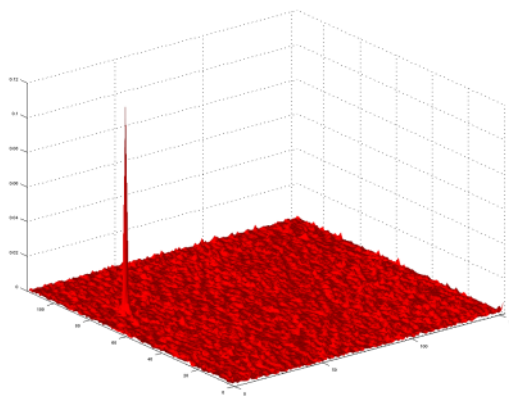


Figure 2: Correlation Surface

### 3 Method

We can divide the problem into two parts, the problem of aligning two images and the assembly of multiple aligned images. The core of the algorithm is aligning two overlapping images correctly. There are a variety of well-tested methods that can be used for this purpose such as cross-correlation, fourier methods, point mapping and elastic model-based matching[3].

For this task, we considered a well-known method called phase correlation for image alignment. It needs no apriori knowledge about the images and is stastically robust. In



Figure 3: Four interpretations of the vector

image registration, robustness implies correct registrations in presence of any effect that may cause a deviation from a perfect match[2]. Phase correlation(PC) is insensitive to changes in gain and to the presence of noise.

Given two images  $I_1$  and  $I_2$  which differ only by a displacement vector  $(d_x, d_y)$ :

$$I_2(x, y) = I_1(x - d_x, y - d_y)$$

their corresponding Fourier transforms  $F_1$  and  $F_2$  will be related by

$$F_2(w_x, w_y) = e^{-j(w_x d_x + w_y d_y)} F_1(w_x, w_y).$$

The two images have the same Fourier magnitude but a phase difference which is directly related to their displacement. The exponential form of  $F_i(\vec{w}) = |F_i|e^{j\phi_i(\vec{w})}$  for  $i = 1, 2$ , then the phase difference is given by  $e^{j(\phi_1 - \phi_2)}$ . Because of the shift theorem, this phase difference is equivalent to the phase of the cross-power spectrum,

$$\frac{F_1(w_x, w_y)F_2^*(w_x, w_y)}{|F_1(w_x, w_y)F_2^*(w_x, w_y)|} = e^{j(w_x d_x + w_y d_y)} \quad (1)$$

where  $*$  is the complex conjugate. The inverse Fourier transform of the phase difference is a delta function centered at the displacement, which is the point of registration. The correlation surface is thus given by,

$$Corr(x, y) = F^{-1} \left\{ e^{j(w_x d_x + w_y d_y)} \right\} = \delta(x - \Delta x, y - \Delta y). \quad (2)$$

Ideally the peak value of the correlation surface  $Corr(x, y)$  should be equal to 1.0, but is lower in reality because of the presence of random noise, but still the peak should be considerably higher than any other point in the surface. Therefore the translation values are estimated by:

$$(x, y) = \underset{\tilde{x}, \tilde{y}}{\operatorname{argmax}}(Corr(\tilde{x}, \tilde{y}))$$

If shifts greater than half the image size are to be expected, the four interpretations of this translation must be considered and the best selected (figure 3)[2]. To select the best interpretation we used the correlation between the aligned parts and chose the part with the best correlation.

The second problem was to assemble all the image fragments into one large image. There are numerous ways to do this. One way is to start with aligning images making groups of two images and then merge the groups of two into groups of four. This is done until only one group is left, the final image. Another way is to start with one image and find all images that align to that image. Then repeat this process for all images, adding them one by one to the alignment, thus gradually building the final image. This is the approach we used. One reason for this choice is that our alignment algorithm does not handle images of unequal sizes. Also the space required for aligning large images becomes very large quickly.

Our assembly algorithm uses two queues, one called aligned and another one called unaligned. During initialization aligned is empty and unaligned contains all the images. Then the first image is popped from unaligned and pushed into aligned. Then while aligned is not empty the first image in the queue is compared to all images in unaligned. For every hit (good alignment) the image in unaligned is moved to aligned. When all images in unaligned have been considered the first image in aligned is removed. The algorithm stops when aligned is empty. There may still be images in unaligned but they do not belong to the final alignment since they don't have good alignments with any of the pictures in the final alignment. To state this algorithm formally:

```

Assume that we have an image set  $\{I_1, \dots, I_N\}$ .
Aligned =  $\{I_1\}$ ;
Unaligned =  $\{I_2, \dots, I_N\}$ ;
while Aligned is not empty
    i = pop(Aligned);
    for all  $j \in$  Unaligned
        score = align(i,j);
        if score > threshold
            push(j,Aligned);
            update(Final image);
            remove(j,Unaligned);
        end if
    end for
end while

```

## 4 Results

For the image set we had, the method we used produced visually perfectly acceptable results. If there was no visual overlap between the images, the method usually resulted in finding the displacement vector as (1,1) with a high correlation value or a low correlation between the supposedly overlapping parts. We set the threshold for the correlation between overlapping parts as 0.5, which seemed to give very good results. The algorithm interprets results of 1,1 as the images are not overlapping at all.

To test how sensitive to noise and gain changes the algorithm really was, we did a series of experiments. First we applied various levels of gaussian noise to one of the images and made an alignment. The resulting alignment was identical to the alignment without noise upto very high levels of noise (image visually unrecognizable). Then we tested various gain changes up to extreme changes in gain such as 30% gain in one picture and 200% gain the other one. The resulting alignment was in the most extreme changes was off only by 1 pixel.

When trying out the assembly algorithm we used nine images that formed almost a circle. We preprocessed the images by subtracting the background(image taken from the empty microscope) from all the images. If we skipped this preprocessing step the alignment algorithm would not find all the alignments. After some experiments with different threshold values the algorithm finally assembled the 9 images correctly. We did this also for another

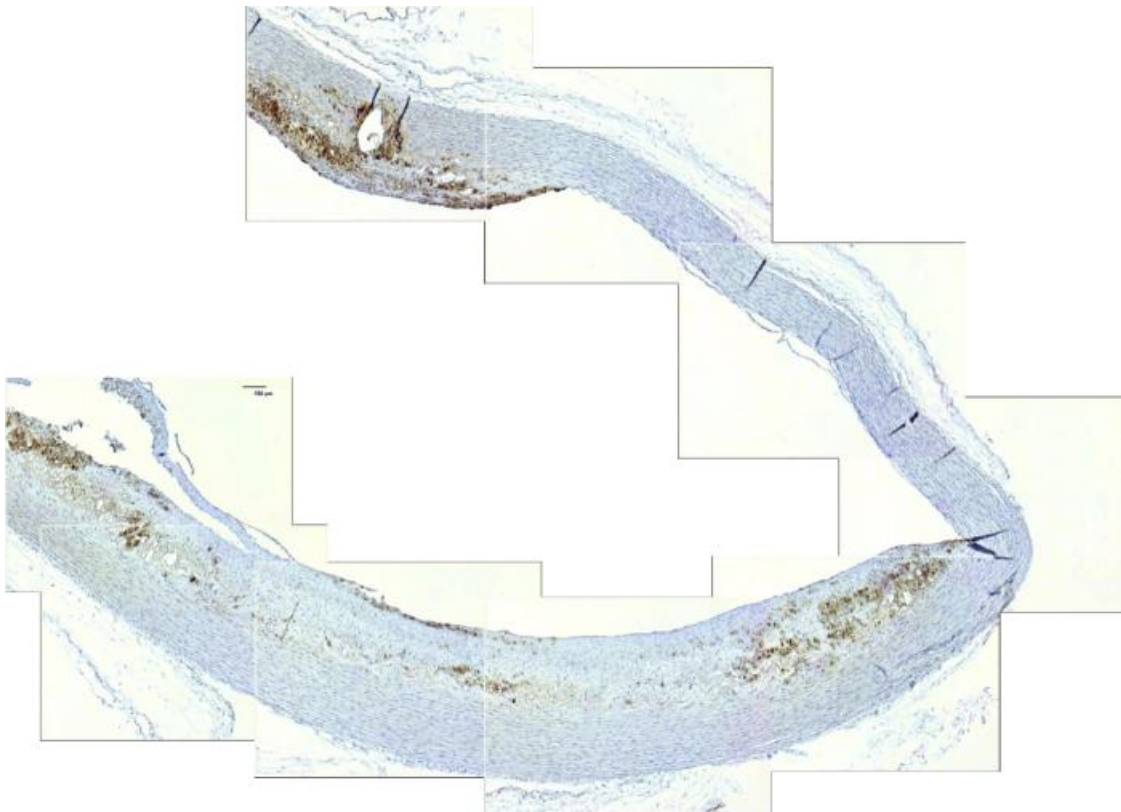


Figure 4: Assembly of 9 images

set of three images.

Performance wise, it took 4.5 seconds of CPU time on a Sunblade 2000 workstation for aligning two images seen in Figure 4. For images of considerably higher resolution, the time for aligning two images is much slower. When aligning multiple images, the upper bound on the number of alignments we have to test is  $\frac{n(n-1)}{2}$ , however the actual

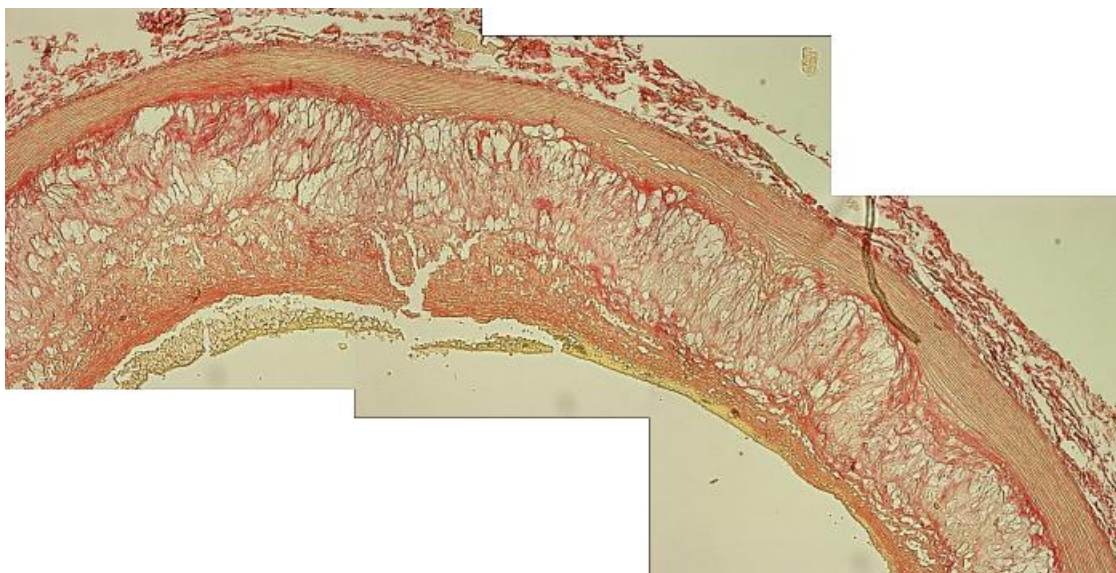


Figure 5: Assembly of 3 images

number of alignments tested is usually less because once we place an image with respect to another one, we do not need to test it versus other already placed images. The number of alignments made is also dependent on the initial order of images, so it might be a good idea to randomize the initial order of images.

## 5 Conclusion

The algorithm and method we have implemented works correctly for all the sample images that were provided. It might have problems with some other samples that it has not been tested with, but with some preprocessing it should be able to align any set of images with reasonable overlap and structural features.

We believe that one aspect it can be improved upon is the presentation of the final image. By changing how we present the overlapping parts, it is possible to get a more visually satisfying image as a result. For example, if images 1 and 2 are overlapping, we can use a weighted average to draw the overlapping pixels, where the weights increase in favor of image 2 as we get closer to image 2 and increase in favor of image 1 as we get closer to image 1. Another fine adjustment might be done to equalize the brightness of the images to make them look more even.

Another problem might be if we have a set of images that will form a ring. If the alignment is off by 1 pixel in a ring of 20 images, when we are adding the last image to the ring, the errors will accumulate and we might be off as much as 20 pixels. However we did not have the chance to test this as we did not have access to a set of images that would form a ring. Overall, we believe that phase correlation in assembling digital microscopy

images is a good and robust method.

## References

- [1] A. Averbuch, Y. Keller, “A Unified Approach To FFT Based Image Registration”.
- [2] A.J. Fitch, A. Kadyrov, W.J. Christmas and J. Kittler, “Orientation Correlation”, British Machine Vision Conference, Vol. 1, pp.133-142, 2002.
- [3] L. G. Brown, “A Survey of Image Registration Techniques”, 1992.