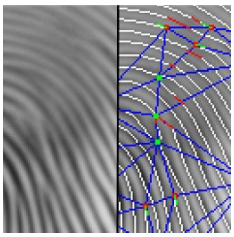
Fingerprint matching algorithm



Description	Automated fingerprint matching
Period	Spring 2010 – Fall 2010
Languages & Libs	Matlab, C++, SDL
Tags	Computer Vision, Data Structures

For my Bachelor of Science degree I developed a novel fingerprint matching algorithm, which ended up beating many alternative methods which were developed by research groups around the world. The used dataset the same which was used for FVC 2000 (Fingerprint Verification Competition).

If the false positive is set to a relatively low level (1% and less), my algorithm produced more true positive matches than 3 of the 11 participated methods, and I was quite happy with that result. This granted me the highest possible grade. Naturally I've learned a lot more since, and could easily improve the accuracy even further by using local texture features. My original method was purely based on detail's relative locations.

The basic steps can be seen in Figure 1. Once the image is smoothed along local direction of the ridges, the image is binarized using locally adaptive 2^{nd} order parametrization. The resulting image is thinned, and two types of interest points can be found (ridge endings and splittings). There can be some spurious ridge endings, if the quality of the print wasn't ideal.

Once spurious interest points have been discarded, a Delaynay triangulation is produced, and from this local groups of 4 interest points are indexed into a 3-dimensional K-d tree data structure, using rotation, scale and translation invariant metrics. Fingerprint matching is based on finding similar groups of interest points, and checking that several independent groups are related to reach other by approximately same *affine transformation*.

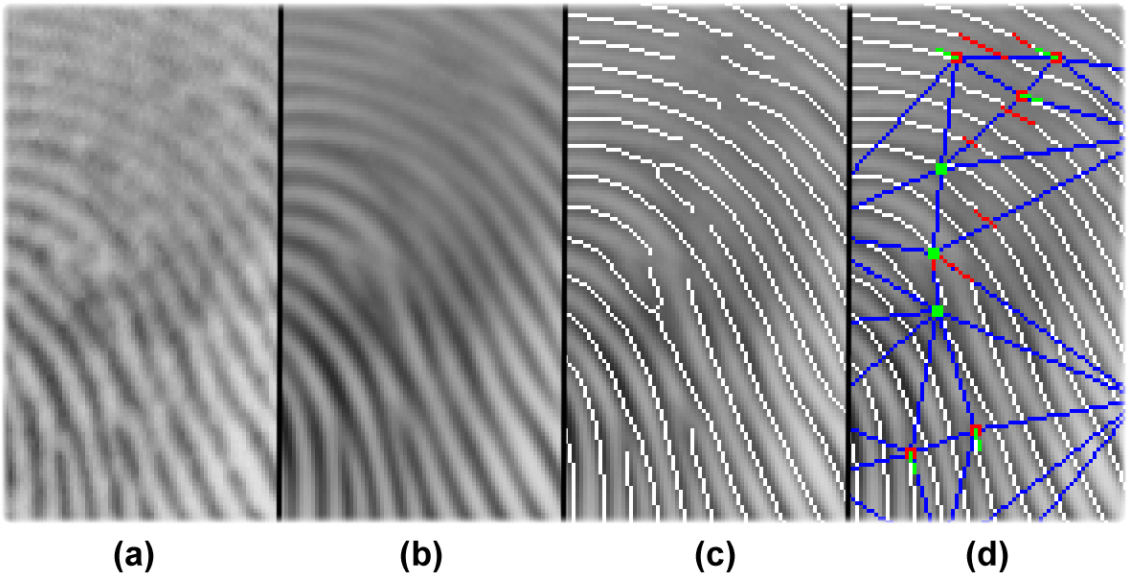


Figure 1: (a) Original image. (b) Smoothed result. (c) Detected ridges. (d) Fixed broken ridges (red) and generated Delaunay-triangulation (blue).