Mobile Fingerprint Identification

1. Demo

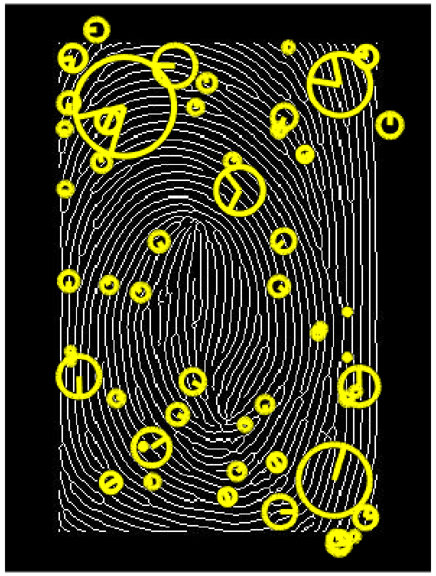
1. Demo

https://github.com/noureldien/FingerprintRecognition

2. Methods

2.1 Feature Extraction

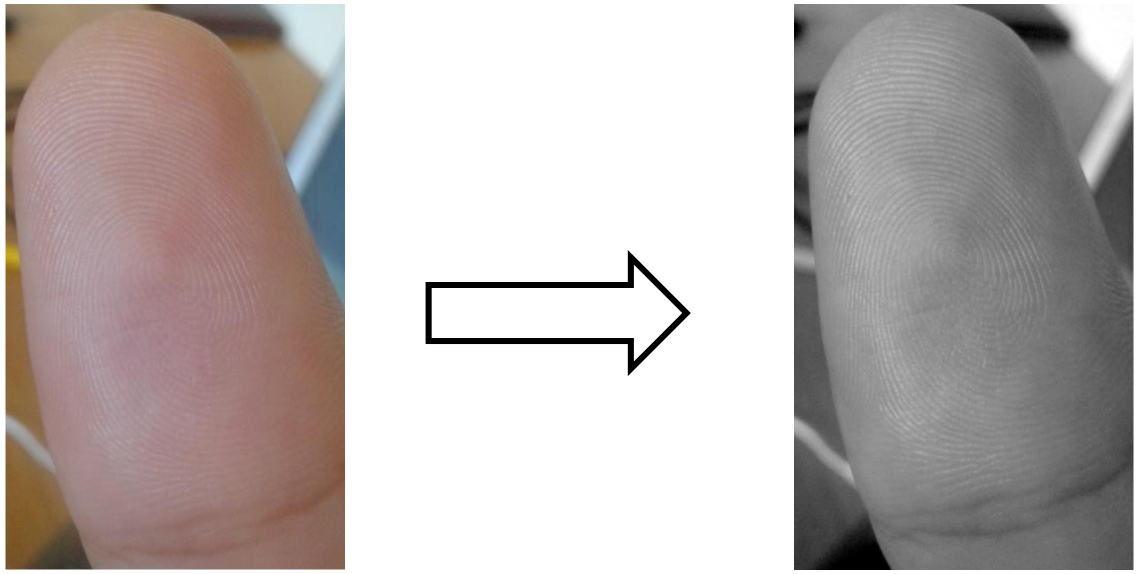
Raw Image



Extracted Features

2.1 Feature Extraction

- Grayscale

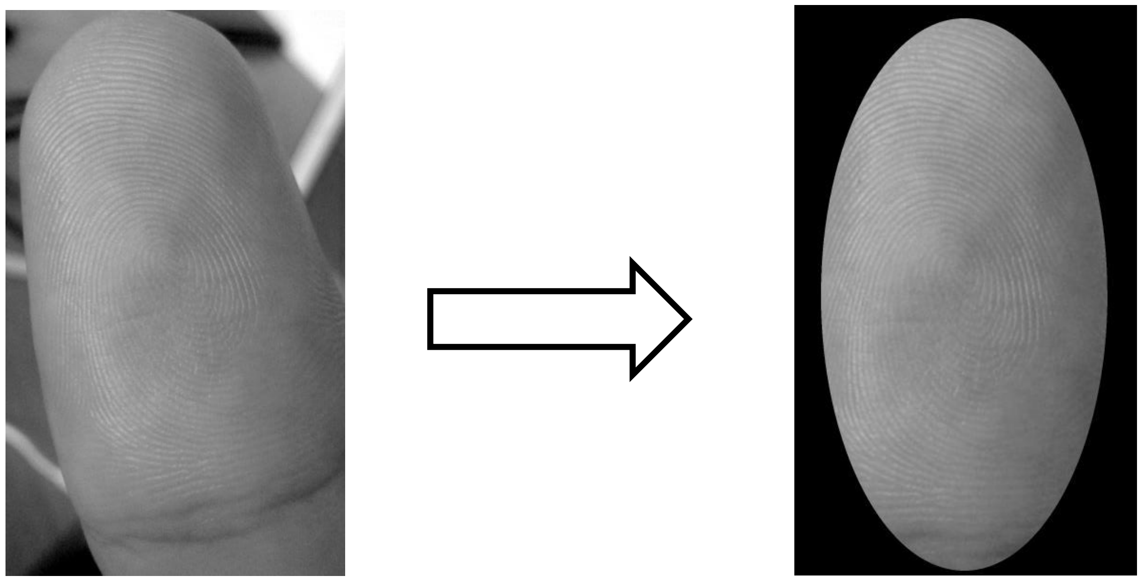


Raw Image

Grayscale Image

2.1 Feature Extraction

- Masking

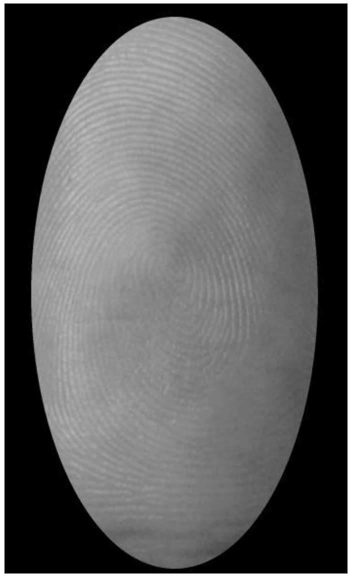


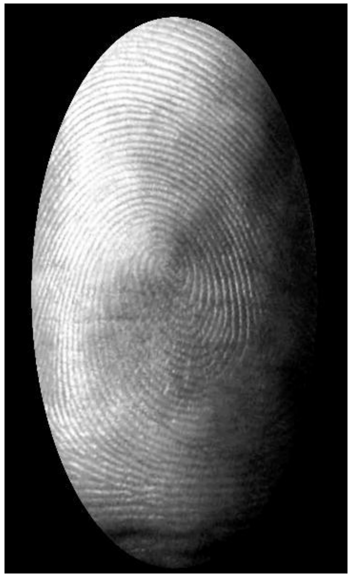
Grayscale Image

Masked Image

2.1 Feature Extraction

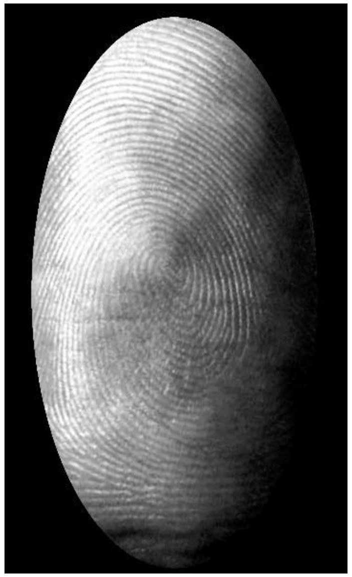
- Histogram Equalisation


Masked Image

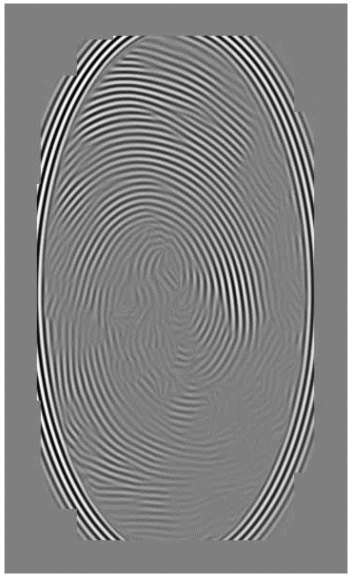

Histogram-equalised Image

2.1 Feature Extraction

- Ridge Orientation Filter



Histogram-equalised Image



Ridge-filtered Image

2.1 Feature Extraction

- Gabor Filter

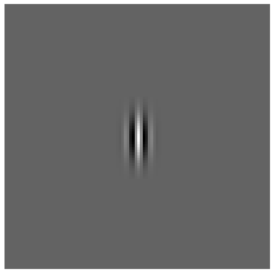
$$g_{\lambda,\theta,\varphi,\sigma,\gamma}(x,y) = \exp\left(\frac{x^2+\gamma^2+y'^2}{2\sigma^2}\right)\cos(2\pi\frac{x'}{\lambda}+\varphi)$$

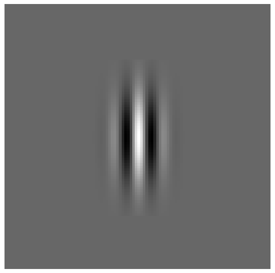
$$x'^2 = x\ cos\theta + y\ sin\theta$$

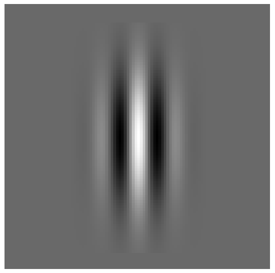
$$y'^2 = -x\ sin\theta + y\ cos\theta$$

2.1 Feature Extraction

- Wavelength (λ)

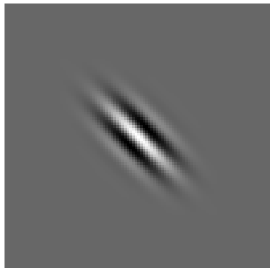

$\lambda = 5$


$\lambda = 10$

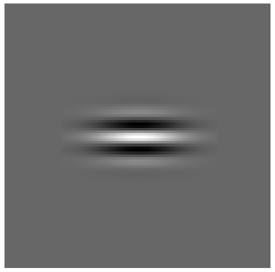

$\lambda = 15$

2.1 Feature Extraction

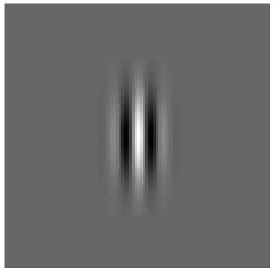
- Orientation (θ)



$\Theta = 45$



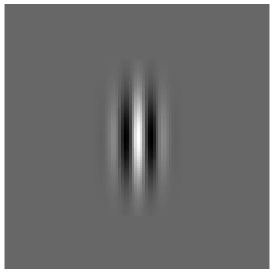
$\Theta = 90$



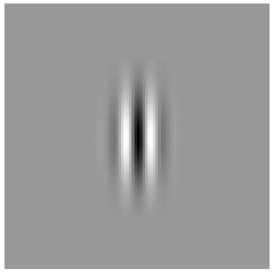
$\Theta = 0$

2.1 Feature Extraction

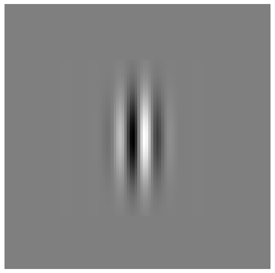
- Phase offset (ϕ)



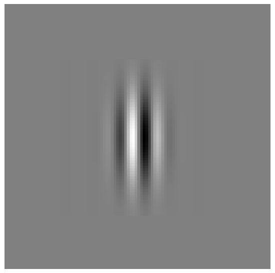
Φ = 0



Φ = 180



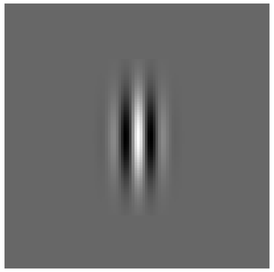
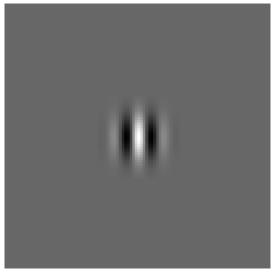
Φ = -90



Φ = 90

2.1 Feature Extraction

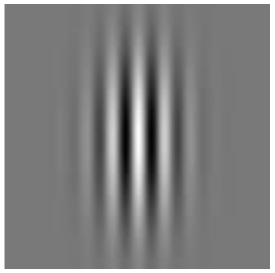
- Aspect ratio (γ)

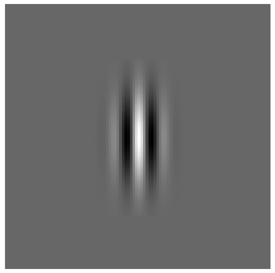



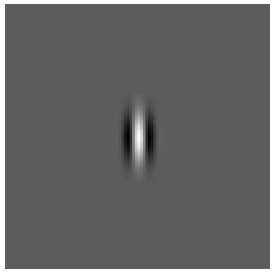
$\gamma = 1/2$ $\gamma = 1/1$

2.1 Feature Extraction

- Bandwidth (b)


b = 0.5


b = 1


b = 2

2.1 Feature Extraction

- Parameter Conditions

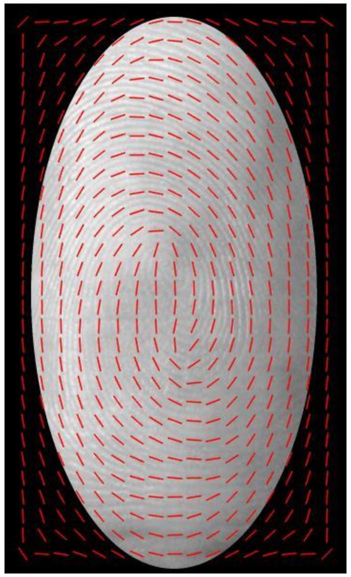
$$b = \ log_2 \frac{\frac{\sigma}{\lambda}\pi + \sqrt{\frac{ln}{2}}}{\frac{\sigma}{\lambda}\pi - \sqrt{\frac{ln}{2}}}$$

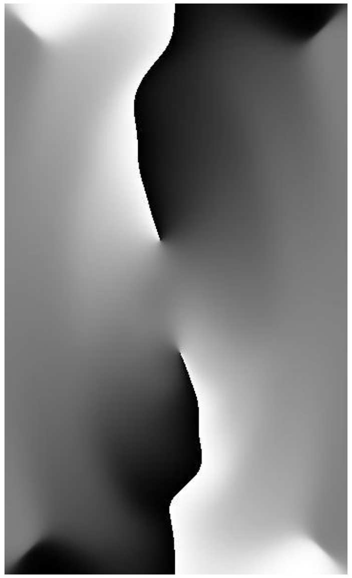
$$\frac{\sigma}{\lambda} = \frac{1}{\pi}\sqrt{\frac{ln2}{2}} * \frac{2^b + 1}{2^b - 1}$$

2.1 Feature Extraction

- Orientations

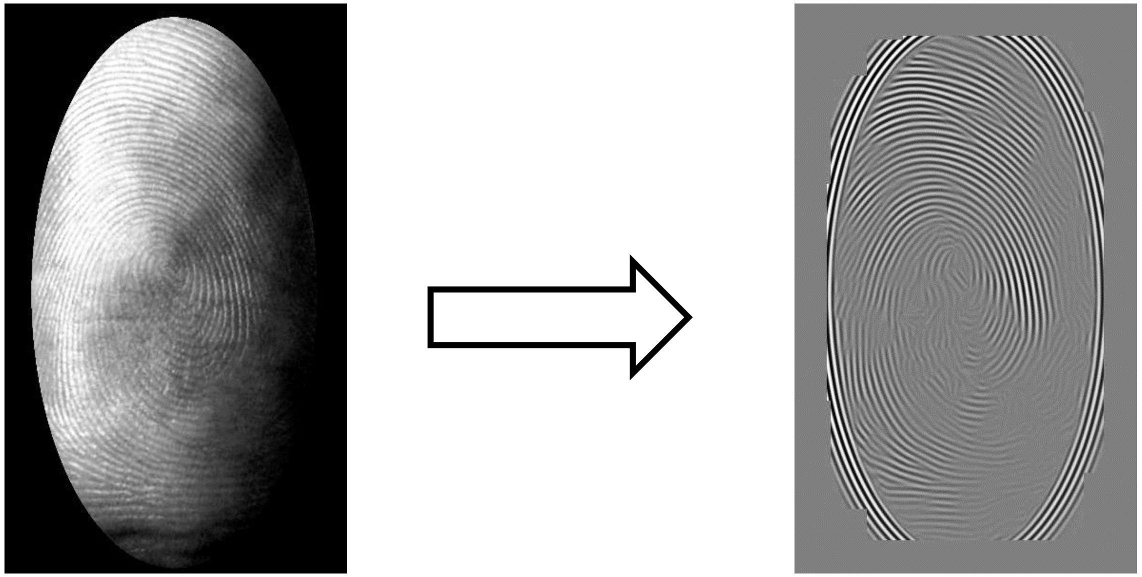
Histogram-equalised Image


Orientation Map


Orientation Visualisation

2.1 Feature Extraction

- Ridge Orientation Filter

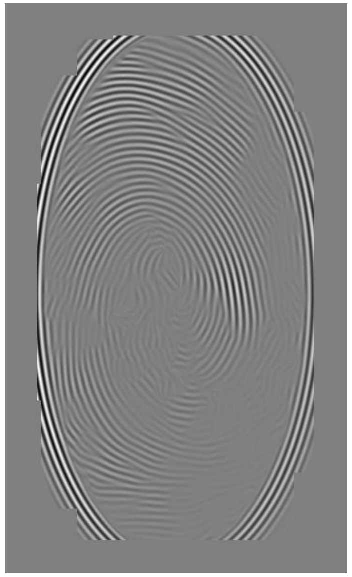


Histogram-equalised Image

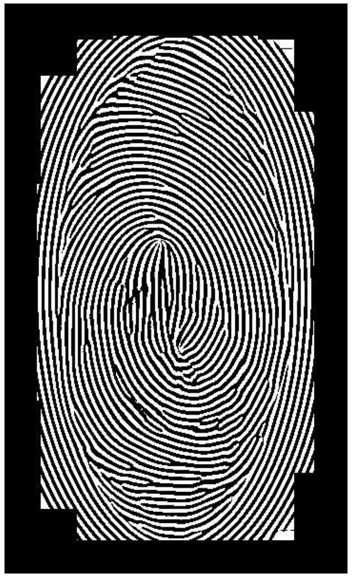
Ridge-filtered Image

2.1 Feature Extraction

- Thresholding



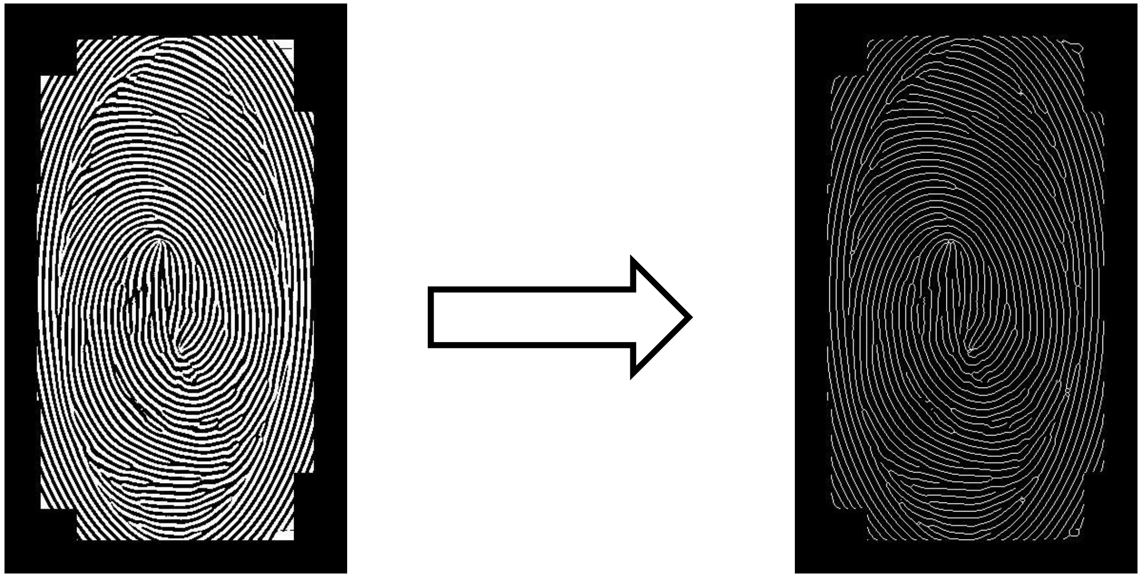
Ridge-filtered Image



Binary Image

2.1 Feature Extraction

- Thinning



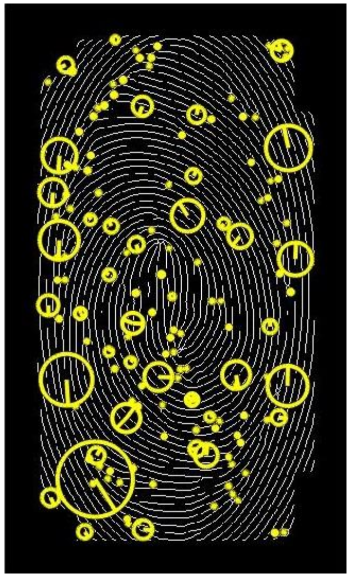
Binary Image

Skeleton Image

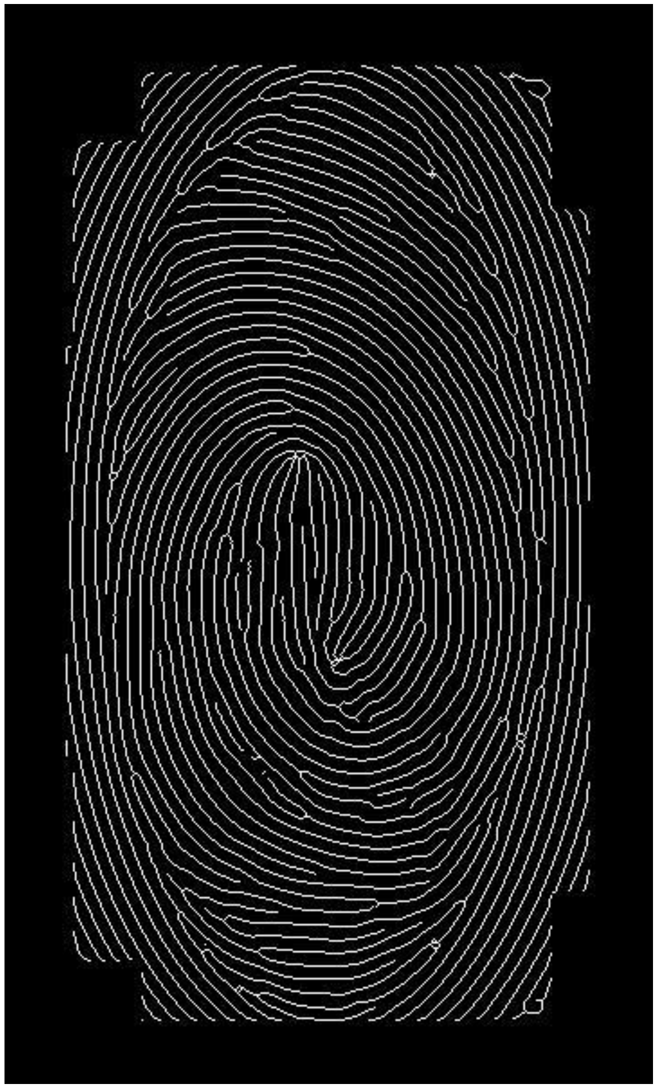
2.1 Feature Extraction

- SIFT Features

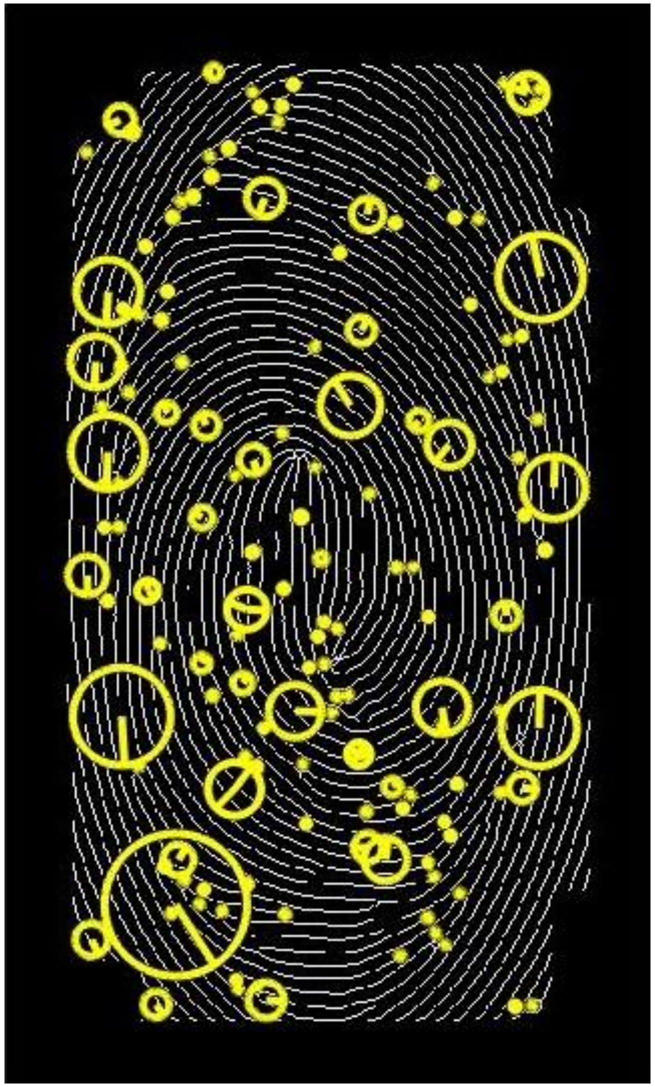
Skeleton Image


Extracted SIFT Features

2.1 Feature Extraction

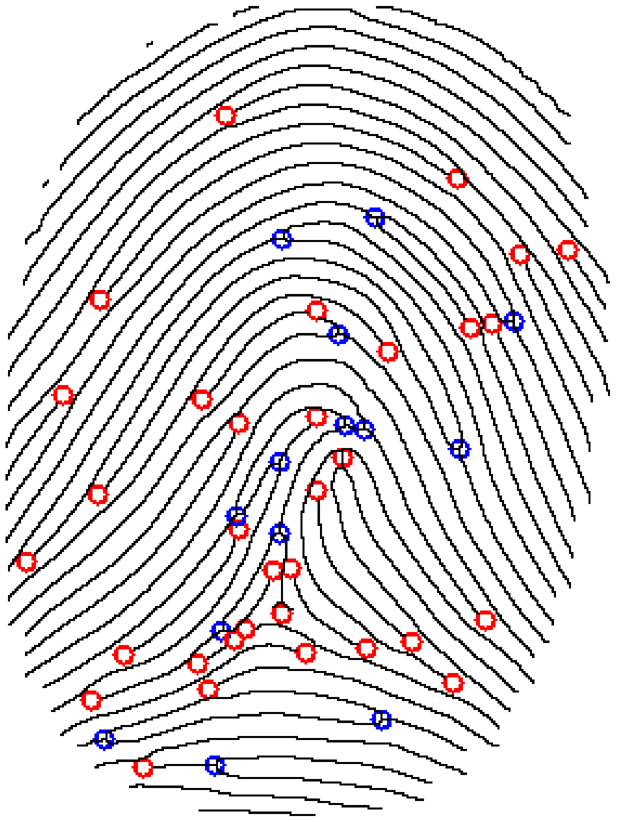


2.1 Feature Extraction



2.1 Feature Extraction

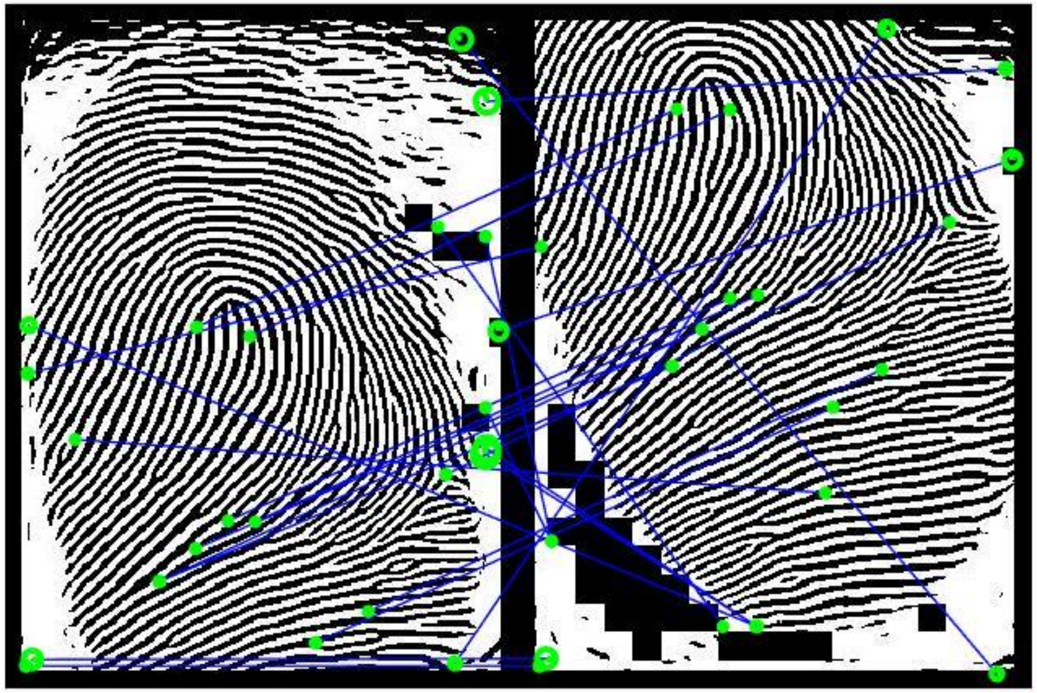
- Minutia Features



http://answers.opencv.org/question/6364/fingerprint-matching-in-mobile-devices-android-platform/

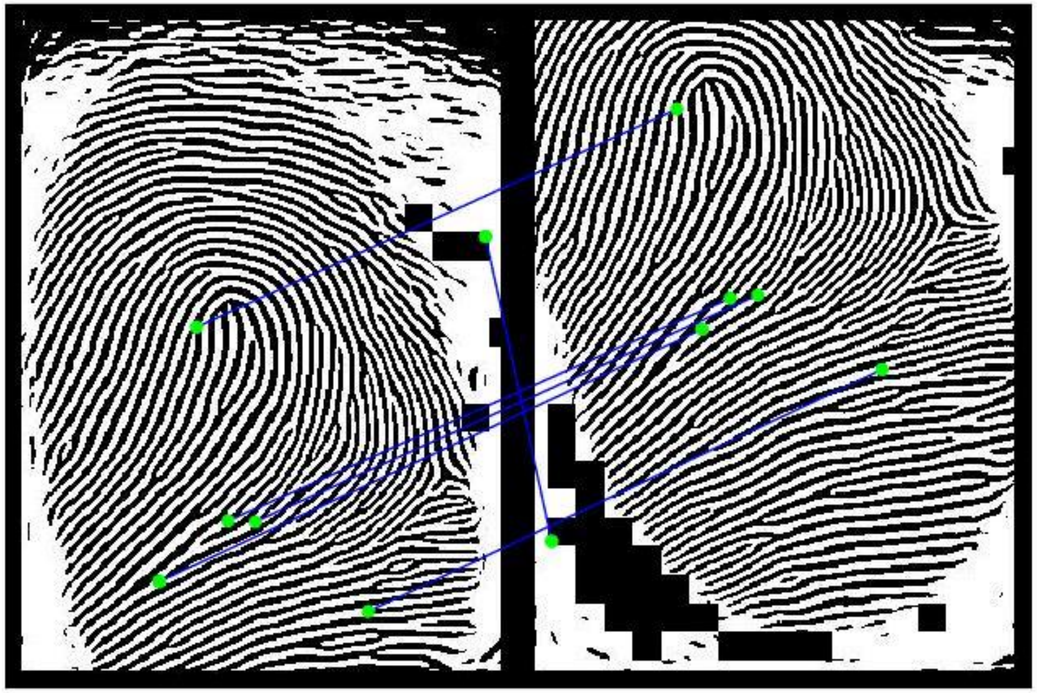
2.2 Feature Matching

- SIFT, RANSAC



2.2 Feature Matching

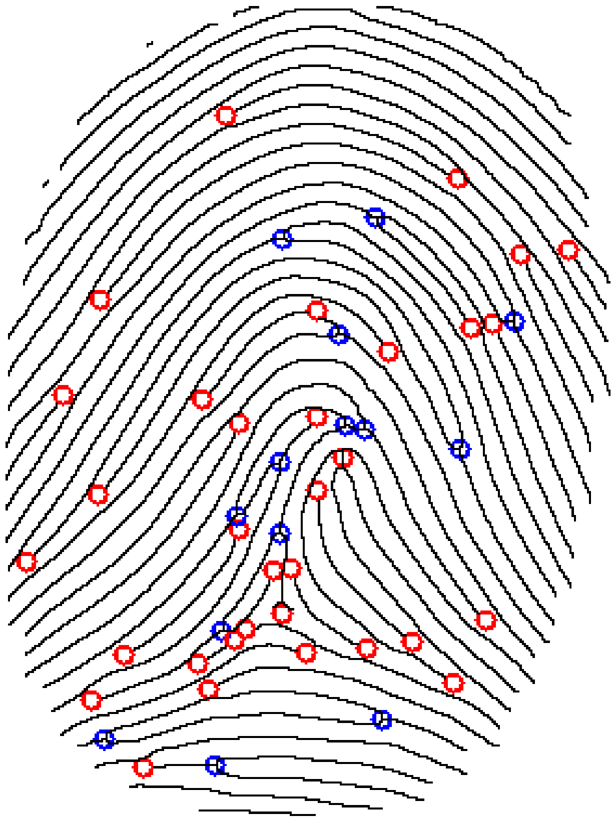
- Good match



3. Conclusion

3.1 Future Work

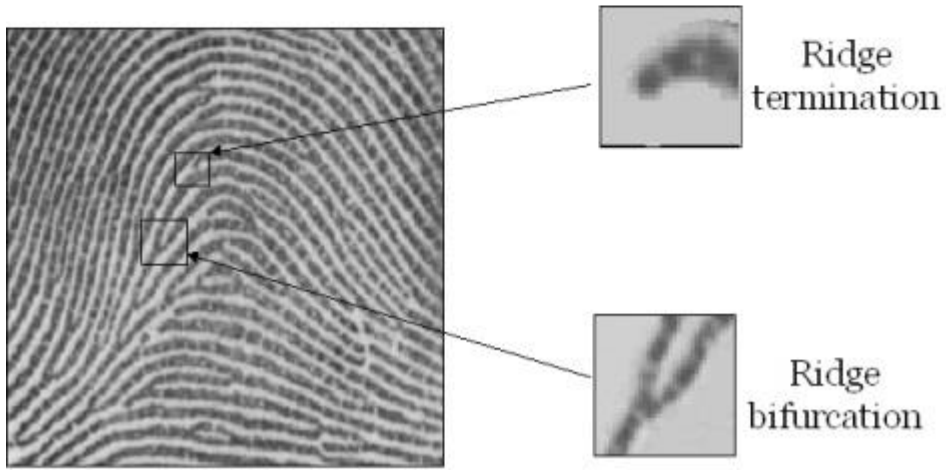
- Minutia Features



http://answers.opencv.org/question/6364/fingerprint-matching-in-mobile-devices-android-platform/

3.1 Future Work

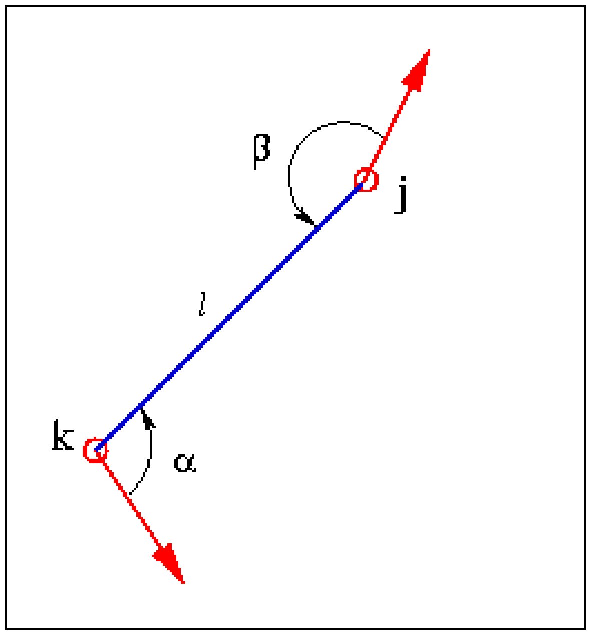
- Minutia Features

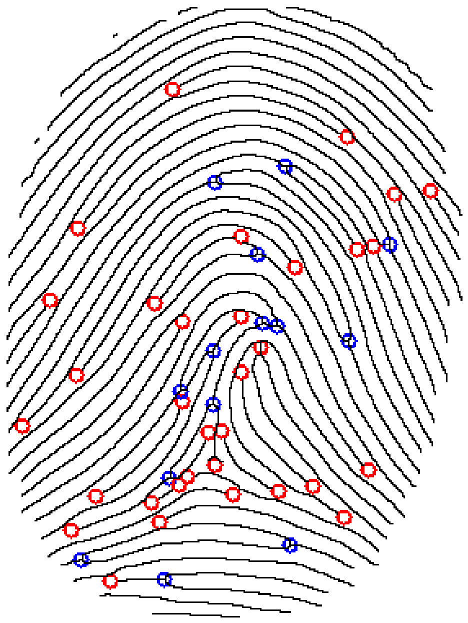


http://answers.opencv.org/question/6364/fingerprint-matching-in-mobile-devices-android-platform/

3.1 Future Work

- Angle Map





http://answers.opencv.org/question/6364/fingerprint-matching-in-mobile-devices-android-platform/

3.2 OpenCV Android

```
reffilter = exp(-(((x.^2)/(sigmax^2))+((y.^2)/(sigmay^2)))/2).*cos(2*pi*medianFreq*x);
```

```
Mat xSquared = new Mat(length, length, CvType.CV_32FC1);
Mat ySquared = new Mat(length, length, CvType.CV_32FC1);
Core.multiply(x, x, xSquared);
Core.multiply(y, y, ySquared);
Core.divide(xSquared, Scalar.all(sigmaX * sigmaX), xSquared);
Core.divide(ySquared, Scalar.all(sigmaY * sigmaY), ySquared);

Mat refFilterPart1 = new Mat(length, length, CvType.CV_32FC1);
Core.add(xSquared, ySquared, refFilterPart1);
Core.divide(refFilterPart1, Scalar.all(-2.0), refFilterPart1);
Core.exp(refFilterPart1, refFilterPart1);

Mat refFilterPart2 = new Mat(length, length, CvType.CV_32FC1);
Core.multiply(x, Scalar.all(2 * Math.PI * medianFreq), refFilterPart2);
refFilterPart2 = matCos(refFilterPart2);

Mat refFilter = new Mat(length, length, CvType.CV_32FC1);
Core.multiply(refFilterPart1, refFilterPart2, refFilter);
```

References

1. Parra, Philippe. "Fingerprint minutiae extraction and matching for identification procedure." *Department of Computer Science and Engineering University of California, San Diego La Jolla, CA* 92093: 0443.
2. Bhowmik, Pankaj, et al. "Fingerprint Image Enhancement And It‟ s Feature Extraction For Recognition." (2012).
3. WIĘCŁAW, Łukasz. "A minutiae-based matching algorithms in fingerprint recognition systems."*Journal of Medical Informatics & Technologies* 13 (2009).
4. Raja, K. B. "Fingerprint recognition using minutia score matching." arXiv preprint arXiv:1001.4186 (2010).
5. Answers.opencv.org, 'Fingerprint matching in mobile devices (Android platform) - OpenCV Q&A Forum', 2015. [Online]. Available: http://answers.opencv.org/question/6364/fingerprint-matching-in-mobile-devices-android-platform/. [Accessed: 02- Apr- 2015].
6. Matlabserver.cs.rug.nl, 'Gabor filter for image processing and computer vision - Model parameters', 2015. [Online]. Available: http://matlabserver.cs.rug.nl/edgedetectionweb/web/edgedetection_params.html. [Accessed: 08- Apr- 2015].

Thank you!

Questions?