A. Brömme, C. Busch, N. Damer, A. Dantcheva, M. Gomez-Barrero, K. Raja,
C. Rathgeb, A. Sequeira, and A. Uhl (Eds.): BIOSIG 2021,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2021

VeinPLUS+: A Publicly Available and Free Software Framework for Vein Recognition

Michael Linortner[1], Andreas Uhl[1]

Abstract: In this work a new and open source software framework for vein recognition is presented, which supports a complete tool chain to conduct biometric experiments. The software can be used out of the box on several publicly available databases and offers a convenient mechanism to configure the tool chain to fit the needs for individual experimental setups. It is implemented in C++ and works on Linux as well as on Windows and therefore offers an advantage over existing vein recognition libraries. The project's aim is to offer a valuable tool to conduct research in the field of vein recognition and importantly, supporting reproducible research.

Keywords: Biometrics, vein recognition, free software framework, VeinPLUS+, C++, software.

1 Introduction

Utilizing veins in fingers or hands as biometric trait, known under the term vein recognition, has become an emerging technology in both industry and science. To accomplish reproducible research outcome in the field of vein recognition or in biometrics in general, two important requirements need to be met: Firstly, experiments should be conducted on publicly available data sets and secondly, the software which produces the experimental results should be publicly accessible as well. Performing biometric experiments requires a software which provides several functionalities. It needs to manage the data set used in the experiment, to provide functions to extract features from biometric samples, and to compare them. To evaluate a recognition technique usually a set of comparison scores needs to be generated following a dedicated protocol which subsequently demands functionality to deduce common accepted performance parameters from these scores like receiver operating characteristic (ROC), detection error tradeoff (DET) or equal error rate (EER). There exist several publicly available data sets containing vein images of hands or fingers, but only two software frameworks consisting of a complete tool chain to run biometric experiments. In this work we propose a new platform independent, publicly available and free software framework suitable to conduct experiments and performance analysis of biometric vein recognition tasks. As it is meant to be a framework for research it meets several requirements crucial for such a type of software: it is platform independent and relies only on few external dependencies, which is important to set up the software quickly and easily; it can be used out of the box. The user configures a vein recognition system as desired, then calls the software, which loads the specified data set, processes each sample, generates a set of comparison scores following the specified protocol and delivers the results of a performance evaluation in the form of DET data.

[1] Department of Computer Sciences, University of Salzburg, Salzburg, Austria, {mlinortner, uhl}@cs.sbg.ac.at

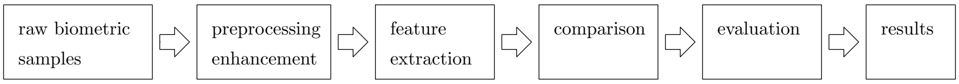


Fig. 1: Biometric tool chain

2 Prior work

Literature research revealed that there are only two software frameworks publicly available which cover a full vein recognition system with all necessary tools to conduct full research experiments out of the box.

One is “Bob”[2] a signal-processing and machine learning toolbox developed by the IDIAP research institute [An12, An17]. The framework is implemented in Python and its core functionality in C/C++. Bob runs on Linux and MacOS 64-bit, but not on Windows. The comprehensive toolbox includes a library for vein recognition providing the feature extraction methods repeated line tracking (RLT) [MNM04], maximum curvature (MC) [MNM07], principal curvature (PC) [Ch09], wide line detector (WLD) [Hu10] and several preprocessing methods, mostly for masking the finger region in a raw biometric sample.

A second framework is the “PLUS OpenVein Toolkit” (OVTK)[3] [KU19], which is specialized for vein recognition and also implements the full biometric extraction, comparison and evaluation tool chain. It is written in MATLAB code and therefore is platform independent, of course, relying on a MATLAB installation, which is a commercial non-free product. It implements the same feature extractions methods as Bob, which are based on a MATLAB implementation by Bram Ton, publicly available on MATLAB Central[4]. In comparison to Bob it provides several other and also more recent feature extraction and preprocessing methods. There is code available for certain vein recognition techniques, like the above mentioned MATLAB implementation for MC, or vein image enhancement like [BRG17] or for deep learning approaches[5], but they are no comprehensive framework supporting a complete biometric tool chain out of the box.

3 The Software

The software presented in this work is a comprehensive and flexible framework for conducting research experiments in the field of vein recognition. It is a platform independent implementation in C++, following the C++11 standard. The software is named VeinPLUS+ where PLUS stands for Paris Lodron University Salzburg and PLUS+ refers to the implementation language.

The software supports a full biometric tool chain as depicted in figure 1: loading and managing data sets, preprocessing/enhancing samples, extracting features, creating comparison scores consisting of genuine and impostor comparisons following a specified protocol, evaluating the computed scores and producing results as DET.

[2] https://www.idiap.ch/software/bob/

[3] http://www.wavelab.at/sources/OpenVein-Toolkit/

[4] https://de.mathworks.com/matlabcentral/profile/authors/1836574

[5] https://github.com/ridvansalihkuzu/vein-biometrics

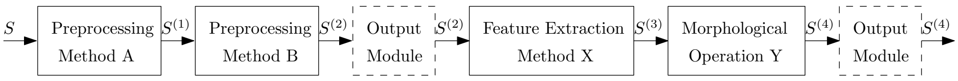
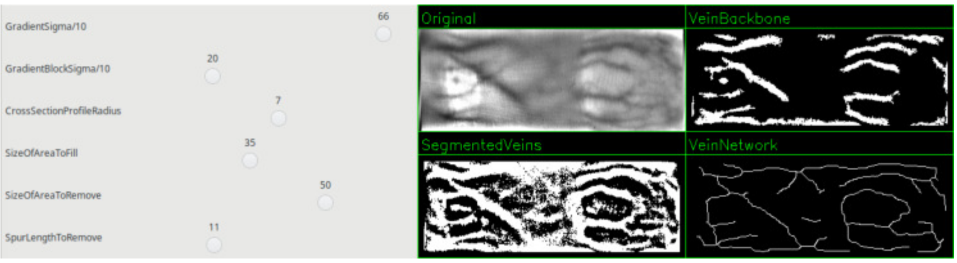


Fig. 2: Preprocessing and feature extraction pipeline example

The whole tool chain is configurable via settings specified in an XML structure and stored as file. Each supported preprocessing, feature extraction and feature comparison method is implemented as a separate module, the parameters of which are defined in the settings. Those modules can be arranged in any order, configurable in the settings file, to create an individual feature extraction pipeline and comparison process fitting the needs of the desired experimental setup without the need to change a line of code, therefore offering maximum flexibility. Figure 2 depicts an exemple configuration for a process pipeline consisting of two preprocessing methods A and B, a feature extraction method X and a morphological postprocessing method Y. Each biometric (vein) sample S from the data set is passed through the processing pipeline consisting of n modules, where $S^{(i)}$ is the processed sample and output of the i-th module. The final output $S^{(n)}$ is used by the comparison module to generate the comparison scores. As indicated in figure 2, at any position of the pipeline an output module can be inserted optionally which writes the samples' current processing state $S^{(i)}$ to files, providing the user a convenient mechanism to check the results after each processing step. Table 1 lists the implemented preprocessing and feature extraction methods. Further it is labeled whether the method is provided by Bob or the OVTK. Note that Bob and the OVTK provide additional preprocessing methods and the

	Method	VP+	OVTK	Bob
Preprocessing	Gaussian filter	✓	✓	✓
	Contrast limited adaptive histogram equalization (CLAHE)	✓	✓	✓
	Finger region masking [LLP09]	✓	✓	✓
	Background subtraction [KZ12]	✓	✗	✗
	Image normalization [Ki12]	✓	✗	✗
	High frequency emphasis filtering (HFE) [Zh09, Ha71]	✓	✓	✗
	Even-symmetric circular Gabor filters [ZY09]	✓	✓	✗
	Single scale retinex (SSR) with guided filter [Xi15]	✓	✗	✗
	Speeded-up adaptive contrast enhancement (SUACE) [BRG17]	✓	✓	✗
Feature extraction	Repeated line tracking (RLT) [MNM04]	✓	✓	✓
	Maximum curvature (MC) [MNM07]	✓	✓	✓
	Principal curvature (PC) [Ch09]	✓	✓	✓
	Wide line detector (WLD) [Hu10]	✓	✓	✓
	Gabor filter as in [KZ12] proposed	✓	✓	✗
	Anatomy structure analysis based vein extraction (ASAVE) [Ya17]	✓	✓	✗
	SIFT based vein recognition	✓	✓	✗
	Minutiae point extraction [LU21]	✓	✗	✗
	Spectral minutiae representation (SMR) [XV10]	✓	✗	✗

Tab. 1: Overview of the implemented methods in VeinPLUS+ (denoted as VP+). It is also marked whether a method is available in Bob or the OVTK.



GradientSigma/10 66

GradientBlockSigma/10 20

CrossSectionProfileRadius 7

SizeOfAreaToFill 35

SizeOfAreaToRemove 50

SpurLengthToRemove 11

Fig. 3: Helper GUI provided by VeinPLUS+. Exemplary the GUI for the ASAVE method is depicted. With the track bars each parameter can be changed and the effect is immediately visible. On the top left of the display the input image is shown, below the segmented veins and on the right side the two extracted feature components “vein backbone” and “vein network” of the ASAVE method.

OVTK also provides additional feature extraction approaches, other than the listed.
Besides the biometric tool chain two additional features are implemented in VeinPLUS+. When conducting experiments it is important to use optimal parameters for the configured tool chain in order to achieve good performance results. Often a grid search is executed to find such optimal parameters, meaning a combination of selected parameter values is evaluated. The software supports such an approach by automatically generating settings files. A settings file which configures the tool chain as desired is used as template. Each parameter, for which different values should be evaluated, is written into an instruction file and an individual list of values is assigned to them. Using the template settings file, the software automatically generates settings files for each value combination specified in the instruction file. The second feature is a helper tool, which provides a graphical user interface (GUI) for each preprocessing and feature extraction method where the method’s parameters can be changed and the effects on the input image are immediately visible on the screen. Figure 3 shows the GUI for the ASAVE method. This convenient tool allows to quickly check the outcome of a method and helps to understand a parameters impact on the outcome especially when several parameters influence each other.

VeinPLUS+ uses the OpenCV library[6] for image processing tasks and CLI11[7] for command line parsing. Therefore, the user needs to install OpenCV in order to use VeinPLUS+. CLI11 is included in the source code of the framework. VeinPLUS+ offers an option to use MatIO[8], a library witch enables reading and writing standard MATLAB files. The software has been developed using OpenCV 3.4.2 and has been tested on Linux with GCC 4.8 and GCC 8.3 as on Windows 10 with Microsoft (R) C/C++ Optimizing Compiler Version 19 (Visual Studio 2017).
The software can be downloaded from our webpage[9] and the source code is hosted on our gitlab server[10].

[6] https://opencv.org
[7] https://github.com/CLIUtils/CLI11
[8] https://github.com/tbeu/matio
[9] http://www.wavelab.at/sources/VeinPLUSplus/
[10] Link can be found on http://www.wavelab.at/sources/VeinPLUSplus/

4 Exemplary experiments

This section describes exemplary experiments for a basic workflow when using the VeinPLUS+ framework. The results are compared to the outcome produced by the OVTK.

4.1 Experimental setup

The experiments are conducted on two data sets from the publicly available "PLUSVein-FV3 Finger Vein Database"[11] [KPU18]. Two data sets are used, both captured under near-infrared (NIR) laser illumination, one showing the palmar, the other showing the dorsal view of a finger, denoted as Las-P and Las-D, respectively. The database has been chosen for the experiments because, additionally to the raw images, extracted region of interest (ROI) images are provided which are used throughout this work. The ROI extraction is a crucial step in vein recognition which can considerably influence the accuracy of the outcome. Using these ROI data helps to improve the reproducibility of the conducted experiments.

As the results produced by the VeinPLUS+ software are compared to the outcome of the OVTK two vein recognition methods are selected which are available in both frameworks. MC, which is a well established recognition method and ASAVE, a more recently published approach, have been selected. The same settings of the two methods have been used on both data sets. The tool chain has been configured as follows:

MC: In the first processing step CLAHE is applied on the input images, followed by applying even-symmetric circular Gabor filters [ZY09]. Afterwards the image is downsampled by a factor of two and the MC features are extracted. To generate the comparison scores the correlation based approach proposed in [MNM04] is used.

ASAVE: Gaussian smoothing is applied on the input image in the first step followed by CLAHE and downsampling by a factor of two. Subsequently, the ASAVE features are extracted and the comparison scores are computed by executing ASAVE's "integration comparison" algorithm [Ya17].

The comparison modules use the fingerprint verification contest (FVC)[12] protocol which results in 3600 genuine and 64620 impostor comparisons for the used data sets. The performance accuracy is evaluated in terms of EER. Both frameworks should produce the same EER values, but variations are expected because of the different implementations. Additionally, the time performance is measured and reported. The settings are available for download together with the VeinPLUS+ software and the experimental results can easily be reproduced, which is one of the important features of the software.

The experiments are executed on a PC with Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz and Debian GNU/Linux 10 (buster). The software has been compiled with GCC 8.3.0 on optimization level O3.

[11] http://wavelab.at/sources/PLUSVein-FV3/

[12] http://bias.csr.unibo.it/fvc2006/perfeval.asp

Data set	Method	VeinPLUS+				OVTK			
		EER(%)	t_e	t_c	t_t	EER(%)	t_e	t_c	t_t
Las-D	MC	0.25	46	185	232	0.28	529	135	709
Las-D	ASAVE	0.24	177	683	864	0.36	647	2461	3135
Las-P	MC	2.56	44	186	235	2.79	536	118	667
Las-P	ASAVE	3.39	175	681	858	3.31	606	2479	3143

Tab. 2: Accuracy (EER) and time performances (t_e: feature extraction time, t_c feature comparison time, t_t: total execution time in seconds) of the VeinPLUS+ framework in comparison to the OVTK.

4.2 Results

Table 2 lists both recognition as well as computational performance results produced by the experiments. As expected, both frameworks produce similar EER values with slight variations. Only for ASAVE on the Las-D data set the relative difference is larger than in the other cases. This can be explained with several divergences in the implementation. For example, OpenCV and MATLAB provide CLAHE, but is implemented differently, even the parameter range differs. For the ASAVE approach there is no original reference implementation available. OVTK implements the original proposed search method for the correlation-based part of the feature comparison procedure: first a horizontal shift is applied and subsequently a vertical shift, which in theory demands less computational effort than a full correlation. Exploiting optimized functionality, offered by OpenCV for example, a full correlation is executed faster in practice, which has been utilized by VeinPLUS+. Small differences in the output of each module in the processing pipeline may sum up and can make a noticeable difference in the final performance. In case of ASAVE on the Las-D data set the considerable difference can be explained that VeinPLUS+ performs a full correlation in the comparison stage and finds a more optimal solution than the implementation in the OVTK.

For the time performance evaluation the time t_e for extracting the features from all 1800 samples in the data set and the time t_c for computing 68220 comparison scores have been measured as well as the total execution time $t_t = t_e + t_c + t_\Delta$, where t_Δ includes the time for loading the data, executing the performance evaluation, etc. It can be observed that both feature extraction methods are executed faster in the VeinPLUS+ framework than in the OVTK, caused by more efficient data handling, function calls and loops in C++ compared to MATLAB. The correlation based feature comparison for MC is processed faster by the OVTK, exploiting the strength of MATLAB. For ASAVE VeinPLUS+ executes the comparisons faster than the OVTK, which can be reasoned by divergences in the implementation details of the algorithm as explained above. As the values for the total execution time show VeinPLUS+ completes the task for both recognition methods faster.

Note that the settings used for the experimental setup are not optimized to achieve the best possible accuracy. The utilized parameters are a compromise to manage that both frameworks process the data in a similar way in order to compare the performance values, again explained by differences in the implementation.

5 Conclusion

In this work a new platform independent software framework for vein recognition has been presented. It is the first free multi-platform vein recognition toolkit. The framework, named VeinPLUS+, comprises the complete tool chain to conduct biometric experiments and can be used out of the box on several publicly available databases for finger vein recognition. The software offers the user a flexible way to individually configure the tool chain to fitting the needs of a desired experimental setup. Sharing this configuration, in form of a small text-based (XML) file, allows perfect reproducibility of the given experiments. The framework is implemented in C++ and runs on Linux as well as on Windows. Therefore, it offers an advantage over the two existing frameworks Bob, which does not work on Windows, and the OpenVein Toolkit, which runs on MATLAB a commercial software package.

VeinPLUS+ is under continuous development with the goal of extending the functionality by adding new enhancement and vein recognition methods as well improving the performance. It is planned to add deep learning vein recognition methods and to parallelize the comparison process. As the software is open source the research community is encouraged to actively contribute to the project. The project's aim is to offer a tool for reproducible research.

6 Acknowledgment

This project was partly funded by the Austrian Science Fund FWF project "Advanced Methods and Applications for Fingervein Recognition" under grant No. P 32201-NBL.

References

[An12] Anjos, A.; Shafey, L. El; Wallace, R.; Günther, M.; McCool, C.; Marcel, S.: Bob: a free signal processing and machine learning toolbox for researchers. In: 20th ACM Conference on Multimedia Systems (ACMMM), Nara, Japan. October 2012.

[An17] Anjos, A.; Günther, M.; de Freitas Pereira, T.; Korshunov, P.; Mohammadi, A.; Marcel, S.: Continuously Reproducing Toolchains in Pattern Recognition and Machine Learning Experiments. In: International Conference on Machine Learning (ICML). August 2017.

[BRG17] Bandara, A. M. R. R.; Rajarata, K. A. S. H. K.; Giragama, P. W. G. R. M. P. B.: Super-efficient spatially adaptive contrast enhancement algorithm for superficial vein imaging. In: 2017 IEEE International Conference on Industrial and Information Systems (ICIIS). pp. 1–6, Dec 2017.

[Ch09] Choi, Joon Hwan; Song, Wonseok; Kim, Taejeong; Lee, Seung-Rae; Kim, Hee Chan: Finger vein extraction using gradient normalization and principal curvature. In: Image Processing: Machine Vision Applications II. volume 7251. International Society for Optics and Photonics, SPIE, pp. 359 – 367, 2009.

[Ha71] Hall, E.L.; Kruger, R.P.; Dwyer, S.J.; Hall, D.L.; Mclaren, R.W.; Lodwick, G.S.: A Survey of Preprocessing and Feature Extraction Techniques for Radiographic Images. IEEE Transactions on Computers, C-20(9):1032–1044, 1971.

[Hu10] Huang, B.; Dai, Y.; Li, R.; Tang, D.; Li, W.: Finger-Vein Authentication Based on Wide Line Detector and Pattern Normalization. In: 2010 20th International Conference on Pattern Recognition. pp. 1269–1272, Aug 2010.

[Ki12] Kim, Hwi-Gang; Lee, Eun Jung; Yoon, Gang-Joon; Yang, Sung-Dae; Lee, Eui Chul; Yoon, Sang Min: Illumination Normalization for SIFT Based Finger Vein Authentication. In: Advances in Visual Computing. Springer Berlin Heidelberg, pp. 21–30, 2012.

[KPU18] Kauba, Christof; Prommegger, Bernhard; Uhl, Andreas: The Two Sides of the Finger - An Evaluation on the Recognition Performance of Dorsal vs. Palmar Finger-Veins. In: Proceedings of the International Conference of the Biometrics Special Interest Group (BIOSIG'18). pp. 1–8, 2018.

[KU19] Kauba, Christof; Uhl, Andreas: An Available Open-Source Vein Recognition Framework. In: Handbook of Vascular Biometrics, chapter 4, pp. 113–142. Springer Nature Switzerland AG, 2019.

[KZ12] Kumar, A.; Zhou, Y.: Human Identification Using Finger Images. IEEE Transactions on Image Processing, 21(4):2228–2244, April 2012.

[LLP09] Lee, Eui Chul; Lee, Hyeon Chang; Park, Kang Ryoung: Finger vein recognition using minutia-based alignment and local binary pattern-based feature extraction. International Journal of Imaging Systems and Technology, 19(3):179–186, 2009.

[LU21] Linortner, Michael; Uhl, Andreas: Towards Match-on-Card Finger Vein Recognition. In: Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security. IH&MMSec '21, Association for Computing Machinery, New York, NY, USA, pp. 87–92, 2021.

[MNM04] Miura, Naoto; Nagasaka, Akio; Miyatake, Takafumi: Feature extraction of finger-vein patterns based on repeated line tracking and its application to personal identification. Machine Vision and Applications, 15(4):194–203, Oct 2004.

[MNM07] Miura, Naoto; Nagasaka, Akio; Miyatake, Takafumi: Extraction of finger-vein patterns using maximum curvature points in image profiles. IEICE TRANSACTIONS on Information and Systems, 90(8):1185–1194, 2007.

[Xi15] Xie, Shan Juan; Lu, Yu; Yoon, Sook; Yang, Jucheng; Park, Dong Sun: Intensity Variation Normalization for Finger Vein Recognition Using Guided Filter Based Singe Scale Retinex. Sensors, 15(7):17089–17105, 2015.

[XV10] Xu, H.; Veldhuis, R. N. J.: Complex spectral minutiae representation for fingerprint recognition. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops. pp. 1–8, June 2010.

[Ya17] Yang, L.; Yang, G.; Yin, Y.; Xi, X.: Finger Vein Recognition with Anatomy Structure Analysis. IEEE Transactions on Circuits and Systems for Video Technology, pp. 1–1, 2017.

[Zh09] Zhao, J.; Tian, H.; Xu, W.; Li, X.: A New Approach to Hand Vein Image Enhancement. In: 2009 Second International Conference on Intelligent Computation Technology and Automation. volume 1, pp. 499–501, Oct 2009.

[ZY09] Zhang, J.; Yang, J.: Finger-Vein Image Enhancement Based on Combination of Gray-Level Grouping and Circular Gabor Filter. In: 2009 International Conference on Information Engineering and Computer Science. pp. 1–4, Dec 2009.