Towards real-time medical diagnostics using hyperspectral imaging technology

Asgeir Bjorgan[a] and Lise Lyngsnes Randeberg[a]

[a]Norwegian University of Science and Technology, Department of Electronics and Telecommunications, O.S. Bragstads plass 2a, 7034 Trondheim, Norway

ABSTRACT

Hyperspectral imaging provides non-contact, high resolution spectral images which has a substantial diagnostic potential. This can be used for e.g. diagnosis and early detection of arthritis in finger joints. Processing speed is currently a limitation for clinical use of the technique. A real-time system for analysis and visualization using GPU processing and threaded CPU processing is presented. Images showing blood oxygenation, blood volume fraction and vessel enhanced images are among the data calculated in real-time. This study shows the potential of real-time processing in this context. A combination of the processing modules will be used in detection of arthritic finger joints from hyperspectral reflectance and transmittance data.

Keywords: tissue characterization, medical and biological imaging, tissue diagnostics, real-time diagnostics, skin segmentation, wavelets

1. INTRODUCTION

Hyperspectral imaging has recently been adopted for imaging of human tissue.[1] High spectral resolution in the technology enables non-contact, spatially resolved skin spectroscopy. The combination of statistical methods and physics-informed models can be used to derive objective, diagnostic information. Possible examples of clinical application include monitoring the progression of wound healing, monitoring tissue perfusion, detecting arthritic finger joints and diagnosing atherosclerosis.[2–6]

A hyperspectral imaging-based arthritis scanner for use in the clinic is currently under development in the Iacobus project.[7] Here, a combination of hyperspectral technology, optoacoustics and ultrasound will be used for early detection of arthritic finger joints. The imaging system will be able to collect hyperspectral transmittance and reflectance data from which inflamed finger joints can be identified. This requires processing algorithms, which should automatically detect and diagnose each finger joint according to the spectral and spatial characteristics of the hyperspectral images.

The high inherent data dimensionality of hyperspectral images can cause high processing times, which could extend beyond what would be usable in the clinic. Limited patient time requires tools aiding in diagnostic decisions to be fast. Processing times must be constrained to a bare minimum. Results, possibly down to a final diagnosis, should be available already within the end of image acquisition.

Processing can be divided into algorithms that require only a single pass on the data, and processing algorithms that require several passes through the data. Any algorithm that can be partially or wholly be brought into the former category is an algorithm that has the potential to be implemented in real-time, as the algorithm can make its single, required pass on the data simultaneously with the data acquisition (line-by-line processing). Real-time processing proper will then depend on whether feasible processing times can be constrained by the time required to obtain the data. This can be achieved using e.g. GPU processing,[8–14] multi-core, memory-optimized CPU processing or FPGA processing.[8] Such processing algorithms will contribute to an earlier diagnostic answer after image acquisition and reduce the total processing cost after the image has been fully obtained.

Further author information: (Send correspondence to A. B.)
A.B.: E-mail: asgeir.bjorgan@iet.ntnu.no

Several necessary pre-processing steps are required. Labeling hyperspectral pixels with spectroscopic information will require fast optical inverse modeling techniques due to the layered, scattering nature of human skin. Typically, such models are not developed with firm timing constraints in mind. Required processing times can be arbitrary, depending on the complexity of the models. Both a reduction in complexity and adaption of fast iteration techniques is required.

The stability and automatic interpretability of the results from inverse modeling methods can be subject to noise in the measurements. Noise removal techniques are necessary. Proper noise removal that can yield the data quality required from the inverse modeling algorithms can be challenging in a line-scanning situation, since preservation of both spatial and spectral resolution is required. However, since the inverse modeling processing step depends on such noise-removed data to be available, and subsequent processing algorithms require the results obtained from the inverse modeling step or noise-removed data, it is imperative to do the noise removal during image acquisition.

Some algorithms like noise removal can require acquisition of image statistics. In these cases, more precise answers are obtained if the statistical routines are allowed to run only on the parts of the image containing human skin. It can also be convenient to extract structural information from the image, like location of fingers and finger joints in human hands. This requires background segmentation. Both are solved by a general skin masking algorithm.

Lastly, the image must be calibrated to reflectance. Optical modeling requires that influences from a varying light source is removed from the image. Skin segmentation requires that the light source is removed from the spectra. Thus, reflectance calibration represents a fundamental step along the processing chain. Reflectance calibration is simple and fast given a reference spectrum, but due to unstable light sources or other influences, should be obtained during each image acquisition using a white reference standard.

Thus, necessary pre-processing steps are calibration, skin segmentation, noise removal and optical inverse modeling. In addition, wavelet processing has been identified as a possible processing technique for obtaining vessel structures.[15,16]

This paper presents a preliminary real-time processing chain using the defined pre-processing steps. Real-time performance is achieved using GPU and CPU parallelization techniques. The accumulated computing time is fitted the time required to collect one line of data to ensure de-facto real-time processing. The results of this study show the potential for real-time processing of clinical hyperspectral data. The presented approach allows for determination of tissue properties while collecting data using a line-scanning camera. The processing modules will be a part of a hyperspectral scanner built for detecting arthritic finger joints. However, the generic nature of the processing modules allows for a variety of applications.

The line-by-line noise removal method has previously been presented in Bjorgan et al.,[17] while the line-by-line inverse modeling method has been presented in Bjorgan et al.[18] Theory and performance of the wavelet techniques were presented in Denstedt et al.[15]

2. PROCESSING BLOCKS

An overview over processing blocks is shown in table 1 and Fig. 1. Modules exist both as stand-alone software libraries (see table 1), and encapsulated within processing stages in a real-time hyperspectral processing framework developed by the Norwegian Defence Establishment.[19] Description of software modules follow below.

2.1 Camera source and computer hardware

Hyperspectral data are obtained using a push-broom Hyspex VNIR-1600 camera (Norsk Elektro Optikk, Lillestrom, Norway).[20] Data was collected in the wavelength range 400-1000 nm, with a spectral resolution of 3.7 nm. The previously mentioned arthritis scanner will be based on a similar system. The technical implementation of this system will be published in a separate paper by the industrial partner.

The time needed to acquire one line of data was 30 ms using an integration time of 10 ms. This defines the real-time deadline limit for the processing. All processing of each line of data should be able to finish within the

Table 1: Processing modules and corresponding source codes.

Method	Implementation	GitHub repository	Commit
Skin masking	libspectralmask	`http://github.com/ntnu-bioopt/spectralmask`	83be811
Calibration	libcalibration		
Inverse model	libgpudm	`http://github.com/ntnu-bioopt/gpu-dm`	8cea95c
Noise removal	libmnf	`http://github.com/ntnu-bioopt/mnf`	0d27847
Vessel contr. enhc.	libhyperwavelet	`http://github.com/ntnu-bioopt/hyperwavelet`	3d91b5d
Vessel extraction	libfrangi	`http://github.com/ntnu-bioopt/libfrangi`	4c98545

arrival of the next. Software is designed to meet a soft real-time requirement, achieved through the use of an NVIDIA GPU (Geforce GTX 670) and an Intel CPU (Core i7). CUDA is used to develop the GPU algorithms, while BLAS routines are used to optimize matrix-based CPU processing whenever possible. Intel MKL was used in the timing tests.

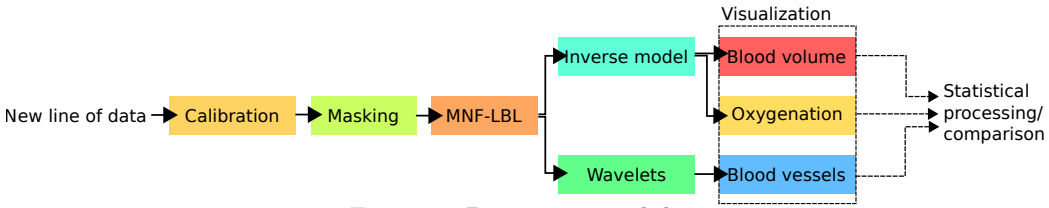

Figure 1: Processing modules.

2.2 Calibration

Input is raw radiance data, output is calibrated reflectance data.

The images are converted to reflectance and corrected for uneven illumination across the field of view using a Spectralon reflectance target (SRT-50-050 Reflectance Target, 12.7 x 12.7 cm, ACAL Bfi Nordic AB, Uppsala). The raw spectra are converted to reflectance by dividing the reflectance value at each wavelength by the corresponding value of the integrated reflectance standard. Given a reflectance standard placed in the start of the image, its position can be detected using simple image statistics. This ensures that the image can be calibrated in real-time using the current state of the light source.

2.3 Masking

Source code for the line-by-line masking algorithm is available on GitHub.[21] Input is calibrated reflectance data, output is human skin segmented from the background.

The spectral angle mapper (SAM)[22] is suitable for measuring differences between spectra. Segmentation can be made single-pass by assuming one or multiple reference spectra representing reflectance from human skin. The line reflectances are compared to the reference spectra in order to segment human skin from the background. However, due to individual variations, the library spectra must also be updated with new information as the image is scanned. This procedure is outlined in Fig. 2. Pixels below an angle threshold are assumed to be skin, while the rest is segmented out as non-skin. Edges are smoothed using a median filter.

2.4 Noise removal

The line-by-line denoising algorithm (MNF-LBL) is described in full in Bjorgan et al.[17] The source code is available on GitHub.[23] Input is noisy reflectance data and the number of bands to keep in the inverse MNF transform, output is denoised reflectance data. The code includes both MNF-LBL and the conventional MNF algorithm,[24] which can also output the MNF bands.

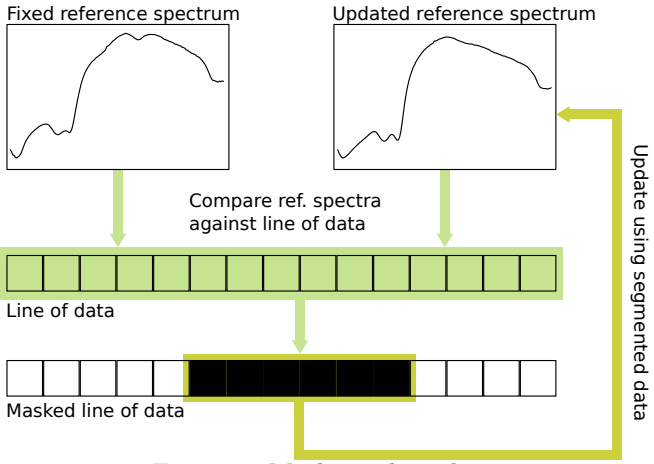

Figure 2: Masking algorithm.

Requirements for the noise removal technique include preservation of spatial and spectral resolution. In hyperspectral imaging, this can be achieved using the Minimum Noise Fraction transform (MNF).[24] The MNF transform is essentially a dimensionality reduction technique. The data are transformed into orthogonal eigenvectors based on spectral variance. A high amount of data and band-to-band correlations are used to extract the pure signal components. The signal space is reordered in terms of SNR (signal to noise-ratio). Inverse transforming a subset of the reordered signal space with an SNR above a defined threshold results in a noise-free dataset. The conventional MNF algorithm requires the full image to be available as one needs to know the spatial covariance. This is a problem when using a camera with a line-scanning geometry. However, a modification to the MNF approach has been found to permit denoising line by line in a manner suitable for real-time applications.

The MNF transform can be expressed as a linear matrix transform A. The columns in the matrix are the eigenvectors found by solving the eigenvalue problem

$$\mathbf{\Sigma}_{\mathcal{N}}\bar{a} = \lambda\mathbf{\Sigma}\bar{a}. \tag{1}$$

The matrices $\mathbf{\Sigma}_{\mathcal{N}}$ and $\mathbf{\Sigma}$ are the covariance matrices of the noise estimate and the image, respectively. Denoising of the image $\mathbf{Z}$ can be summed up as a large matrix operation $\mathbf{D}$ defined as

$$\begin{aligned} \mathbf{Z}^* &= (\mathbf{A}^{-1})^T\mathbf{R}\mathbf{A}^T\mathbf{Z} && (2)\\ &= \mathbf{D}\cdot\mathbf{Z} && (3)\\ &= \mathbf{D}\cdot[\bar{Z}_1,\ldots,\bar{Z}_p]. && (4) \end{aligned}$$

The matrix $\mathbf{R}$ is the identity matrix with the last m elements of the diagonal set to zero. Given $\mathbf{D}$, each $\bar{Z}_i$ can be denoised separately.

The proposed MNF line-by-line algorithm (MNF-LBL) denoises the image line by line by updating the statistics with each new line of data, deriving a new D_i and applying it to the new line of data for denoising. Care is taken to update the statistics in a numerically stable way, as this can be challenging when only a single pass is made on the data. BLAS and LAPACK is used for memory optimization in the matrix operations.

2.5 Inverse model

The inverse model (GPU-DM) is described in full in Bjorgan et al.[18,25] The source code is available on GitHub.[26] Input is reflectance data and inverse modeling parameters, output is extracted properties like epidermal melanin content, superficial and deep blood volume fraction and oxygenation. The analysis can be extended to e.g. water and fat content, and amounts of bilirubin, methemoglobin and beta-carotene.

A layered diffusion model with plane wave illumination is used to simulate the light transport in human tissue.[18,27,28] It is possible to obtain a closed-form, analytic expression for the diffuse reflectance from this model.[27]

The complexity of the expression and its analytic derivative is suitable for a self-contained GPU kernel implementation by limiting the model to a two-layered skin model (epidermis and dermis). Epidermal and dermal absorption coefficients are fitted to the reflectance and spectrally unmixed using a non-negative least squares algorithm. Variations in penetration depth across the input wavelength range is exploited to yield depth-resolved properties.

Assuming pixel-independency, the model is suitable for GPU implementation using simple steps, at the cost of some crosstalk or blurring between properties obtained from neighboring pixels. However, considering the complex light scattering interactions, complete separation would only be possible at great computational cost and with a higher uncertainty.

2.6 Vessel contrast enhancement (wavelets)

Source code for the hyperspectral wavelet transform is available on GitHub.[29] Input is denoised reflectance data, output is deomposed wavelet coefficients.

Wavelet methods can be used to decompose hyperspectral images into coefficients where image information is accumulated. Symlet[30] filters of length 8 were used to obtain coefficients where blood vessels had a high contrast to the rest of the tissue. It has been shown that this can be correlated to vessels or blood content at various depths by Denstedt et al.,[15] and preliminary vessel characterization algorithms were outlined in Bjorgan et al.[16] Vessels can be enhanced using the Frangi vesselness filter,[31–33] before subsequent vessel characterization.

A combination of the filterbank algorithm[34] and memory optimizations available in the BLAS library is used to enable fast line processing. A GPU version of the filterbank algorithm is also available, but does not provide a significant speed advantage due to the need to rearrange the datastream from BIL to BIP interleave.

3. RESULTS AND DISCUSSION

Clinical applicability of hyperspectral imaging puts constraints on the processing time for the data due to the limited time the medical personnel has for each patient. Each processing step towards the final result is required to be as fast as possible, preferably real-time. Availability of these real-time processing steps realizes the possibility for fast hyperspectral diagnostic systems. One such system is currently under development for the automatic detection and diagnosis of arthritic finger joints.

The time needed for each processing module was found to be within the real-time deadline limit of 30 ms per line of data. Proper scheduling of the processing operations enabled a lag of no more than 2 lines. Scheduling example is shown in Fig. 3. Times for line-by-line processing modules are shown in table 2.

GPU processing times include transfer to and from GPU device memory. Applying all memory transfers in parallel with the compute stream can reduce the time of e.g. the inverse model down to 3.5 ms, though this will also require some extra overhead in management of memory arrays. Computing times of the processing modules were measured separately. It is assumed that these will remain approximately the same when scheduled together in a multi-threaded environment. However, this can be subject to the availability of resources like memory, the memory bus, cache or processing resources. Thread overhead, mutex and semaphore access can also interfere with the time when all processing modules are combined. Additional overhead due to camera control, 3D corrections, continuous autofocus and coregistration of images is also to be expected.

Some processing steps are from the onset difficult to implement line-by-line. Examples include processing which would seem to require a priori knowledge of image data, like proper noise removal and skin masking. However, this study shows that both are possible to implement in real-time in such a way that only one pass is made on the data, or at worst, multiple passes through single, disjoint lines of data. Results from noise removal and masking are compared against the noisy results in Fig. 4, 5 and 6. Noise removal enables variations in derived properties to stand out against random variations due to noise.

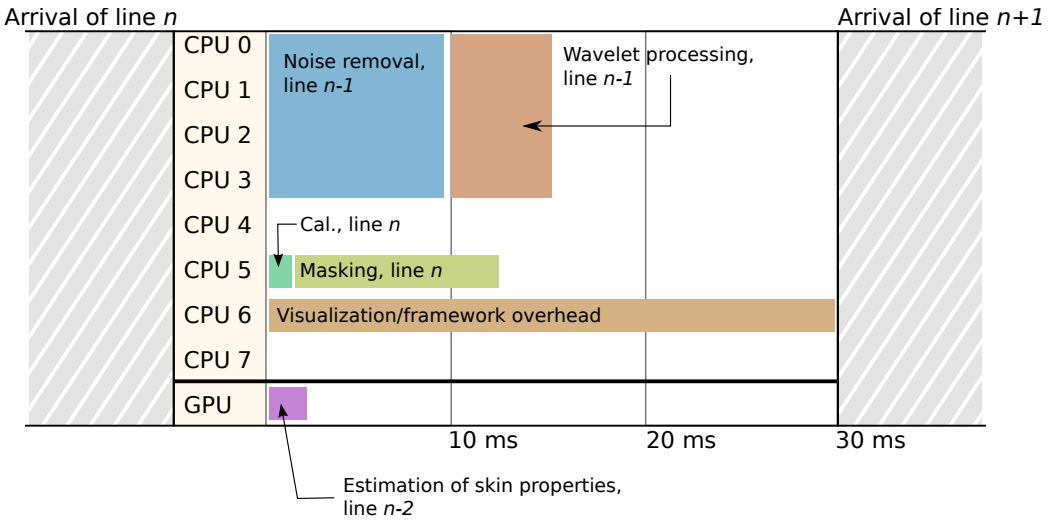

Figure 3: Scheduling of the processing operations for a single line of data.

Table 2: Mean computing time for processing modules as applied to a single line of data. The times were measured separately and sequentially after being applied to an image consisting of 3958 lines, 1600 samples and 160 bands. GPU times include serial memory transfer.

Method	#CPU cores	Mean processing time (ms)
Skin masking, one ref. spec.	1	7.39 ± 4.06
Calibration	1	1.48 ± 0.10
Inverse model (GPU-DM)	GPU	4.23 ± 0.15
Noise removal (MNF-LBL)	4	9.98 ± 1.05
Vessel contrast enhancement	4	3.62 ± 0.19
Vessel contrast enhancement	GPU	1.06 ± 0.04

3.1 Further work

Real-time processing modules have been developed, allowing for processing or preprocessing to be done at the time of acquisition in a line-scanning setup. The system allows for modular combination of the algorithms needed to extract the information relevant for the specific application in question. At the end of camera acquisition, processing results are ready to be input into diagnostic algorithms or statistical post-processing algorithms. The definition of post-processing, diagnostic algorithms will depend on the application in question. So far, this approach has been tested on arthritis and non-healing ulcers.

The system will be extended to processing of transmittance data through hands and finger joints.[5,35] Preprocessing modules like calibration, masking and noise removal can mostly be used unmodified. Wavelet processing techniques need to be fine-tuned to the spectral characteristics of transmittance data.

The combination of results obtained from reflectance and transmittance data will be used in the detection of arthritis in finger joints. Here, the statistical post-processing will involve a comparison between results obtained from each finger joint. Approximate locations of the finger joints will be obtained by extracting structural information from the masked hand. The diagnostic performance of the algorithms will be evaluated in a future medical trial. The real-time performance of the developed algorithms ensures real-time visualization of results during scan, which will be useful during testing and further development of the system. In production, the real-time performance ensures the necessary low response time in the system.

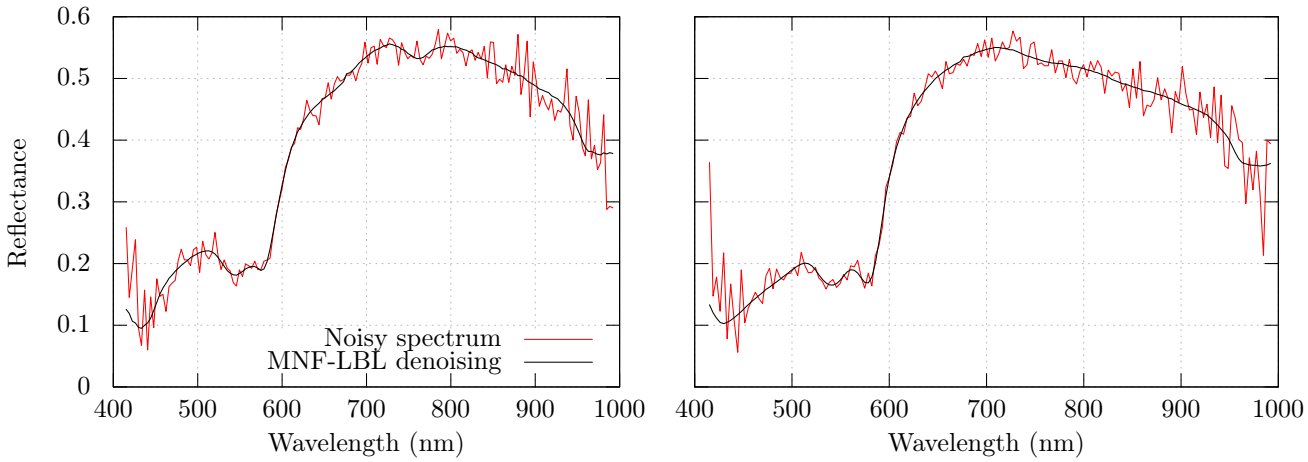

Figure 4: Comparison of spectra before and after noise removal using MNF-LBL.

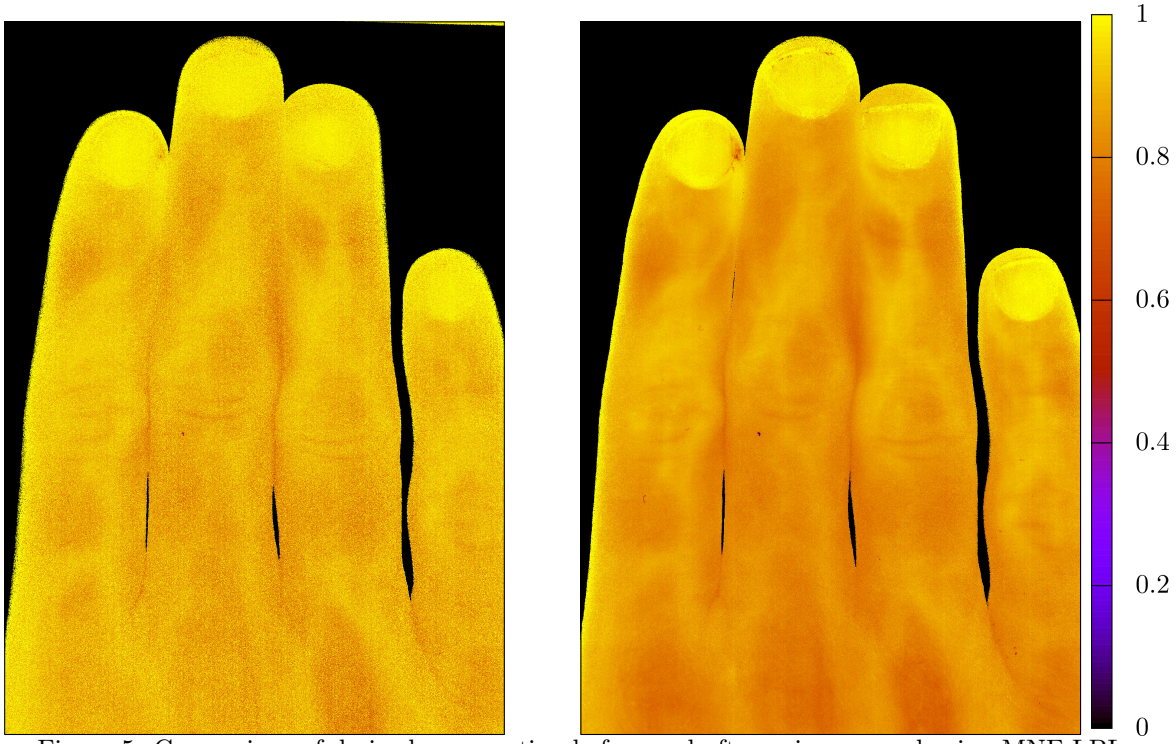

Figure 5: Comparison of derived oxygenation before and after noise removal using MNF-LBL.

4. CONCLUSION

The developed real-time processing modules provide a basic framework for a future, flexible real-time diagnostic system based on hyperspectral imaging. The algorithms will be put to test in a clinical hyperspectral device developed for early detection of arthritic finger joints.

Acknowledgments

Thanks to Norsk Elektro Optikk AS and the Norwegian Defence Research Establishment for fruitful discussions and exchange of code. This work is a part of the Iacobus project, `http://www.iacobus-fp7.eu`. Iacobus is supported by the European Commission's 7th RTD Framework Programme Collaborative Project No. 305760.

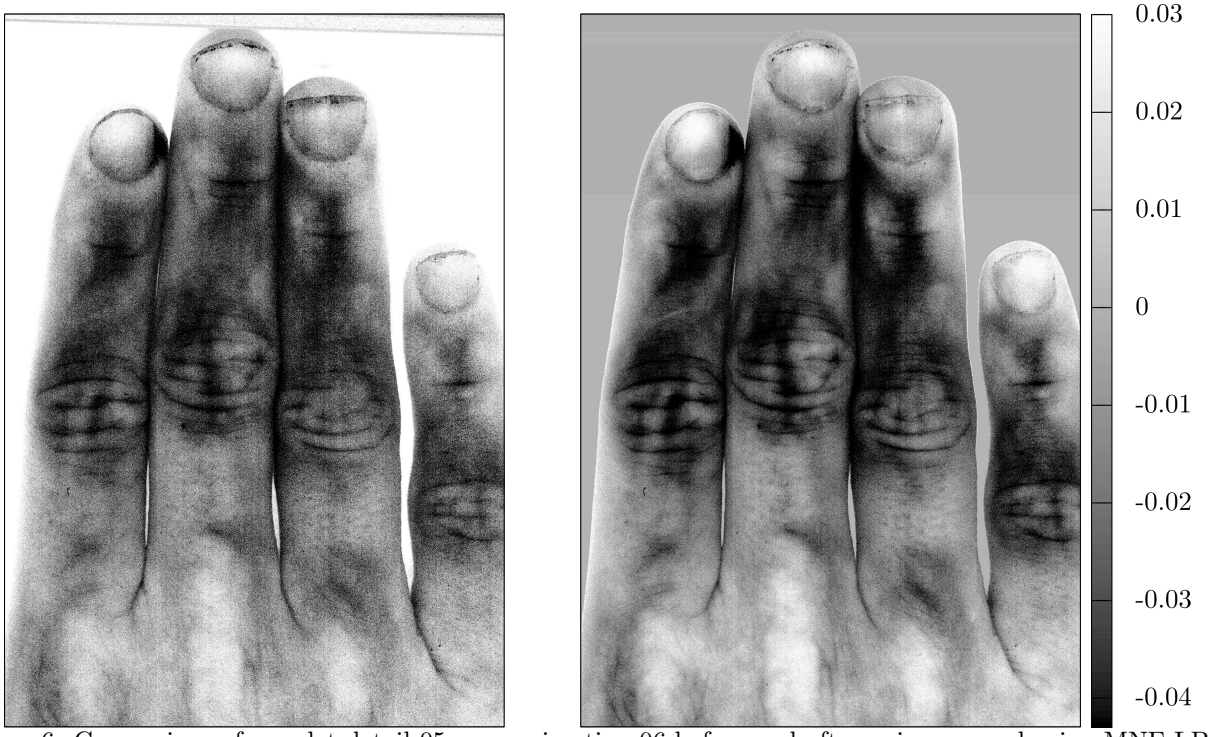

Figure 6: Comparison of wavelet detail 05, approximation 06 before and after noise removal using MNF-LBL.

REFERENCES

[1] Lu, G. and Fei, B., "Medical hyperspectral imaging: a review," *J. Biomed. Opt.* **19**(1), 010901 (2014).
[2] Denstedt, M., Pukstad, B. S., Paluchowski, L. A., Hernandez-Palacios, J. E., and Randeberg, L. L., "Hyperspectral imaging as a diagnostic tool for chronic skin ulcers," *Proc. SPIE* **8565**, 85650N–85650N–14 (2013).
[3] Randeberg, L. L., Larsen, E. L. P., and Svaasand, L. O., "Characterization of vascular structures and skin bruises using hyperspectral imaging, image analysis and diffusion theory," *J Biophotonics* **3**(1-2), 53–65 (2010).
[4] Sorg, B. S., Moeller, B. J., Donovan, O., Cao, Y., and Dewhirst, M. W., "Hyperspectral imaging of hemoglobin saturation in tumor microvasculature and tumor hypoxia development," *J. Biomed. Opt.* **10**(4) (2005).
[5] Milanic, M., Paluchowski, L. A., and Randeberg, L. L., "Simulation of light transport in arthritic-and non-arthritic human fingers," *Proc. SPIE* **8936** (2014).
[6] Larsen, E. L., Randeberg, L. L., Olstad, E., Haugen, O. A., Aksnes, A., and Svaasand, L. O., "Hyperspectral imaging of atherosclerotic plaques in vitro," *J. Biomed. Opt.* **16**(2) (2011).
[7] "Iacobus project web page." Available online on http://www.iacobus-fp7.eu. Last visited 2015-06-17.
[8] Gonzalez, C., Sanchez, S., Paz, A., Resano, J., Mozos, D., and Plaza, A., "Use of fpga or gpu-based architectures for remotely sensed hyperspectral image processing," *Integration* **46**, 89–103 (2013).
[9] González, D., Sánchez, C., Veguilla, R., Santiago, N. G., Rosario-Torres, S., and Vélez-Reyes, M., "Abundance estimation algorithms using nvidia (r) cuda (tm) technology," *Proc. SPIE* **6966** (2008).
[10] Setoain, J., Prieto, M., Tenllado, C., and Tirado, F., "Gpu for parallel on-board hyperspectral image processing," *Int J High Perform Comput Appl* **22**, 424–437 (Nov. 2008).
[11] Setoain, J., Tenllado, C., Prieto, M., Valencia, D., Plaza, A., and Plaza, J., "Parallel hyperspectral image processing on commodity graphics hardware," *Parallel Processing Workshops, 2006. ICPP 2006 Workshops. 2006 International Conference on* , 472 (2006).
[12] Tarabalka, Y., Haavardsholm, T. V., Kåsen, I., and Skauli, T., "Real-time anomaly detection in hyperspectral iimage using multivariate normal mixture models and gpu processing," *J Real-Time Image Proc* **4**(3), 287–300 (2009).
[13] Sanchez, S. and Plaza, A., "Real-time implementation of a full hyperspectral unmixing chain on graphics processing units," *Proc. SPIE* , 81570F–81570F–9 (2011).

[14] Sanchez, S., Ramalho, R., Sousa, L., and Plaza, A., "Real-time implementation of remotely sensed hyperspectral image unmixing on gpus," *J Real-Time Image Proc* (2012).
[15] Denstedt, M., Bjorgan, A., Milanic, M., and Randeberg, L. L., "Wavelet based feature extraction and visualization in hyperspectral tissue characterization," *Biomed. Opt. Express* **5**(12), 4260–4280 (2014).
[16] Bjorgan, A., Denstedt, M., Milanic, M., Paluchowski, L. A., and Randeberg, L. L., "Vessel contrast enhancement in hyperspectral images," *Proc. SPIE* **9318** (2015).
[17] Bjorgan, A. and Randeberg, L. L., "Real-time noise removal for line-scanning hyperspectral devices using a minimum noise fraction-based approach," *Sensors* **15**(2) (2015).
[18] Bjorgan, A., Milanic, M., and Randeberg, L. L., "Estimation of skin optical parameters for real-time hyperspectral imaging applications," *J. Biomed. Opt.* **19**(6) (2014).
[19] Skauli, T., Haavardsholm, T. V., I., K., Arisholm, G., Kavara, A., Opsahl, T. O., and A., S., "An airborne real-time hyperspectral target detection system," *Proc. SPIE* **7695** (2010).
[20] "Hyspex vnir-1600 main specifications." Available online on http://www.hyspex.no/products/hyspex/vnir1600.php. Last visited 2014-07-08.
[21] Bjorgan, A., "libspectralmask source code repository," (2015). Available online on http://github.com/ntnu-bioopt/spectralmask. Last visited 2015-06-17.
[22] Keshava, N. and Mustard, J., "Spectral unmixing," *IEEE Signal Proc Mag* **19**, 44 –57 (jan 2002).
[23] Bjorgan, A. and Randeberg, L. L., "libmnf software repository," (2015). Available online on http://github.com/ntnu-bioopt/mnf. Last visited 2015-06-17.
[24] Green, A. A., Berman, M., Switzer, P., and Craig, M. D., "A transform for ordering multispectral data in terms of image quality with implications for noise removal," *IEEE Trans. Geosci. Remote Sensing* **26**(1), 65–74 (1988).
[25] Bjorgan, A. and Randeberg, L. L., "Implementation of a real-time algorithm for estimating spatially resolved optical properties from hyperspectral images," tech. rep., NTNU (2015). Available online on http://ntnu-bioopt.github.io/publications/technical_reports/Bjorgan2015_gpudm_implementation.pdf. Last visited 2015-06-17.
[26] Bjorgan, A. and Randeberg, L. L., "libgpudm software repository," (2015). Available online on http://github.com/ntnu-bioopt/gpudm. Last visited 2015-06-17.
[27] Svaasand, L., Norvang, L., Fiskerstrand, E., Stopps, E., Berns, M., and Nelson, J., "Tissue parameters determining the visual appearance of normal skin and port-wine stains," *Laser Med Sci* **10**, 55–65 (1995).
[28] Haskell, R. C., Svaasand, L. O., Tsay, T., Feng, T., McAdams, M. S., and Tromberg, B. J., "Boundary conditions for the diffusion equation in radiative transfer," *J Opt Soc Am A* **11**, 2727–2741 (Oct 1994).
[29] Bjorgan, A. and Randeberg, L. L., "libhyperwavelet source code repository," (2015). Available online on http://github.com/ntnu-bioopt/hyperwavelet. Last visited 2015-06-17.
[30] Daubechies, I., [*Ten Lectures on Wavelets*], no. 61 in CBMS/NSF Series in Applied Math. (1992).
[31] Frangi, A. F., Niessen, W. J., Vincken, K. L., and Viergever, M. A., "Multiscale vessel enhancement filtering," *Proc. Med. Image. Comput. Assist. Interv.* **1496**, 130–137 (1998).
[32] Bjorgan, A., "libfrangi source code repository," (2015). Available online on http://github.com/ntnu-bioopt/libfrangi. Last visited 2015-06-17.
[33] Kroon, D.-J., "Hessian based frangi vesselness filter," (2010). Available online on http://www.mathworks.com/matlabcentral/fileexchange/24409-hessian-based-frangi-vesselness-filter. Last visited 2015-06-17.
[34] Mallat, S. G., "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE T. Pattern Anal.* **11**(7), 674–693 (1989).
[35] Milanic, M., Paluchowski, L. A., and Randeberg, L. L., "Simulation of light transport in arthritic- and non-arthritic human fingers," *In submission* .