Raw data extraction from electrocardiograms with Portable Document Format

Nuria Ortigosa[a], Vicente M. Giménez[a]

[a] *I.U. Matemática Pura y aplicada, Universidad Politécnica de Valencia*
Camino de Vera s/n, 46022 Valencia, Spain
nuorar@upvnet.upv.es

Abstract

During the last two decades there has been a thorough research and development of standards and protocols in order to cope with different electrocardiogram formats from heterogeneous acquisition systems. Despite the efforts of public and private consortiums on creating a standardized electrocardiogram (ECG) storage format, there is still not a single one. Indeed, there is also the necessity of access to raw data of the ECGs previously acquired. Most of these documents have been saved as Adobe PDF files, since for medical staff it is an easy format for later visualization. However, this format presents difficulties when trying to access original raw data for subsequent studies and signal analysis. In this manner, this paper presents an application that obtains plain numerical data from ECG files stored with PDF format. Data can also be exported to one of the most common file formats in existence, to

be easily accessed thereafter.

Keywords: ECG storage formats, Portable Document Format, Scalable Vector Graphics

1. Introduction

The cost of an electrocardiograph is low, specially when compared to other medical equipment (such as, for example, the radiological one). This is probably why different ECG recording companies have been working with such a large number of different data formats. During the last 20 years, the definition and adoption of a single standardized format for ECG recordings has been promoted for public and private consortiums [1, 2] due to the fact that interoperability between them could save around 77.8 billion of dollars per year [3], just in the United States.

Therefore, a European funded project called OpenECG [4] started in 2002 to promote the adoption of the SCP-ECG ("Standard Communications Protocol for computed assisted ECG") by implementing visualization tools and interoperability standards to help manufacturers on their implementations [5].

Similarly, in 2001 the United States Food and Drug Administration (FDA) requested a standard which eased the storage and exchange of ECGs informa-

tion, since it received a large number of annotated ECGs collected in a wide variety of formats. Thus, the HL7 (a not-for-profit international organization for sharing health information) developed the Health Level 7-annotated ECG (HL7-aECG) standard [6], a new XML-based format.

Another format to store medical data is DICOM [7], which initially was developed for medical image storage and, in 1995, finally became a European standard also used for cardiac and vascular information.

Surprisingly, efforts towards a single standardized ECG format are not only supported by the Standard Development Organizations. In 2003, Philips Medical Systems published the XML (extensible markup language) schema that they used for their entire line of ECG products [8], and began to deliver this information to its costumers and to European OpenECG project members [9, 10].

Nevertheless, although interoperability farther than at regional or national level is a desirable goal, it may still take another 20 years to be fully achieved [11]. In this sense, integrated applications for standardized data formats exchange have been developed to cope with the numerous different formats and to facilitate their visualization [12]. For example, van Ettinger et al. [13] have implemented a conversion library and an ECG viewer to

work with HL7-aECG, SCP-ECG and DICOM files. In order to facilitate interoperability for records obtained in Italy, Marcheschi et al. [14] designed a network infrastructure to manage with different standards. Similarly, [15] and [16] use XML language as a central platform to exchange data between these formats. Moreover, in [17] Trigo et al. have recently developed a modular application to exchange ECGs information of different data formats across healthcare information systems.

Unfortunately, ECGs previously stored often present data formats different from those commented above. In particular, many electrophysiologists and medical staff usually save only the plots of the ECG recordings as Adobe PDF ("Portable Document Format") due to the fact that, when ECG acquisition equipment is replaced in hospitals, ECG recordings stored with proprietary formats hamper the free access to previous ECGs in the medical history. Therefore, PDF format assures them an easy visualization in any computer and allows subsequent revision for clinical studies and patients evolution analysis. Thus, in this paper we present an application developed to recover raw data from ECG recordings saved in PDF format, in order to facilitate later signal processing of ECGs previously stored by medical staff in this format, and to integrate them with the existing information system

of the hospital.

The rest of the paper is organized as follows. Features of ECGs stored with PDF format, a description of their structure, and the application interface are described in Section 2. Section 3 reports the application computational requirements, some particularities and discussion of results. Finally, conclusions and future work are drawn in Section 4.

2. Materials and Methods

2.1. SVG files

The ECGs recorded with PDF format whose raw data we wanted to extract were acquired using the Philips PageWriter TC50 [18]. When revising the technical sheets of this electrocardiograph we noticed that, when Philips Medical Systems began to develop their ECG data format based on XML, they turned to the Scalable Vector Graphics (SVG) application language [9], which is ideal to display easily two-dimensional graphics.

SVG [19] is a markup language which is able to formalize a set of graphical elements such as rectangles, lines, polygons, etc. The most relevant feature of SVG documents is that they can be considered simultaneously as both text and images. This ambivalence points out to SVG as an ideal format to

connect textual and graphical information in just one file [20].

Therefore, we first tried to convert electrocardiograms to SVG data format, to check whether the Philips electrocardiograph stored PDF files as vector graphics. In order to do so, we used the free software tool Inkscape for conversion [21]. We confirmed that the converted SVG file was formed by a set of graphical elements located with their absolute coordinates, and not by a bitmap data set. As a result, we decided to extract raw data by implementing an application whose input were a PDF file and the generated output contained the corresponding true-value numerical data of each lead, in order to make easier the subsequent signal analysis.

2.2. Electrocardiogram leads

An electrocardiogram is the main instrument for the diagnosis of cardiovascular diseases. It can be defined as the graphical representation of the electrical activity of the heart.

In electrocardiography, the term "lead" refers to the measurement of voltage between two electrodes, which are placed on the patient's body. To perform a standard 12-lead ECG, it takes 10 electrodes: 6 electrodes for the chest leads (V1, V2, V3, V4, V5, V6), and 4 electrodes to acquire the limb leads (I, II, III) and the augmented limb leads (aVL, aVF, aVR).

Depending on the duration of the leads that the electrophysiologist prefers to visualize, ECGs can be stored using different printout formats. Ones of the most popular printout formats are: 3×4, $3 \times 4 + 1$, and 6×2. Details of these lead organization will be provided in Figure 2 and Section 3.

2.3. Electrocardiogram structure

As aforementioned, the SVG file obtained by Inkscape from ECG PDF conversion presents a structure similar to an XML file. The horizontal and vertical coordinates (in pixels) of each sample from the ECG lead is represented under a label called $< path >$. Coordinates can be defined as absolute or relative to the last coordinate, depending on whether after the label $< path >$ the token is $d =$ "M or $d =$ "m, respectively. In our case, we have configured Inkscape options to generate SVG files with absolute coordinates, being the abscissa and the ordinate coordinates separated by a comma.

We must remark that, under the label $< path >$ there appear not only the polylines which correspond to the ECG leads, but also the lines that define the background grid and the reference pulses. They can be differentiated by their corresponding line widths and colours, as well as by their order of appearance along the SVG file and the number of samples for which they are defined.

2.4. Application interface

The presented application has been programmed using GUIDE, the MATLAB graphical user interface development environment. Figure 1 shows the main window of the ECG data extraction application.

The interface is divided in three subsections. In the left top half the input ECG parameters are shown, such as the paper speed and the amplitude scale of the limb and the chest leads. It is standard to represent each microVolt (μV) of amplitude as 10 mm, and each second as 25 mm. However, as faster paper speeds and different scales can be used, these parameters can be modified by the user.

Then, the user must proceed to load the desired PDF file whose raw data is going to be extracted. Once it has been selected, the application calls the Inkscape program as a background task, in order to perform the PDF conversion to SVG format. All our ECGs acquired and recorded information of 12 leads, according to one of the following printout formats: 3×4, $3 \times 4 + 1$ or 6×2 (see Figure 2). So, next step consists of automatically detect which lead configuration was used when the loaded ECG was recorded. This automatic recognition takes place by detecting and counting the number of reference pulses.

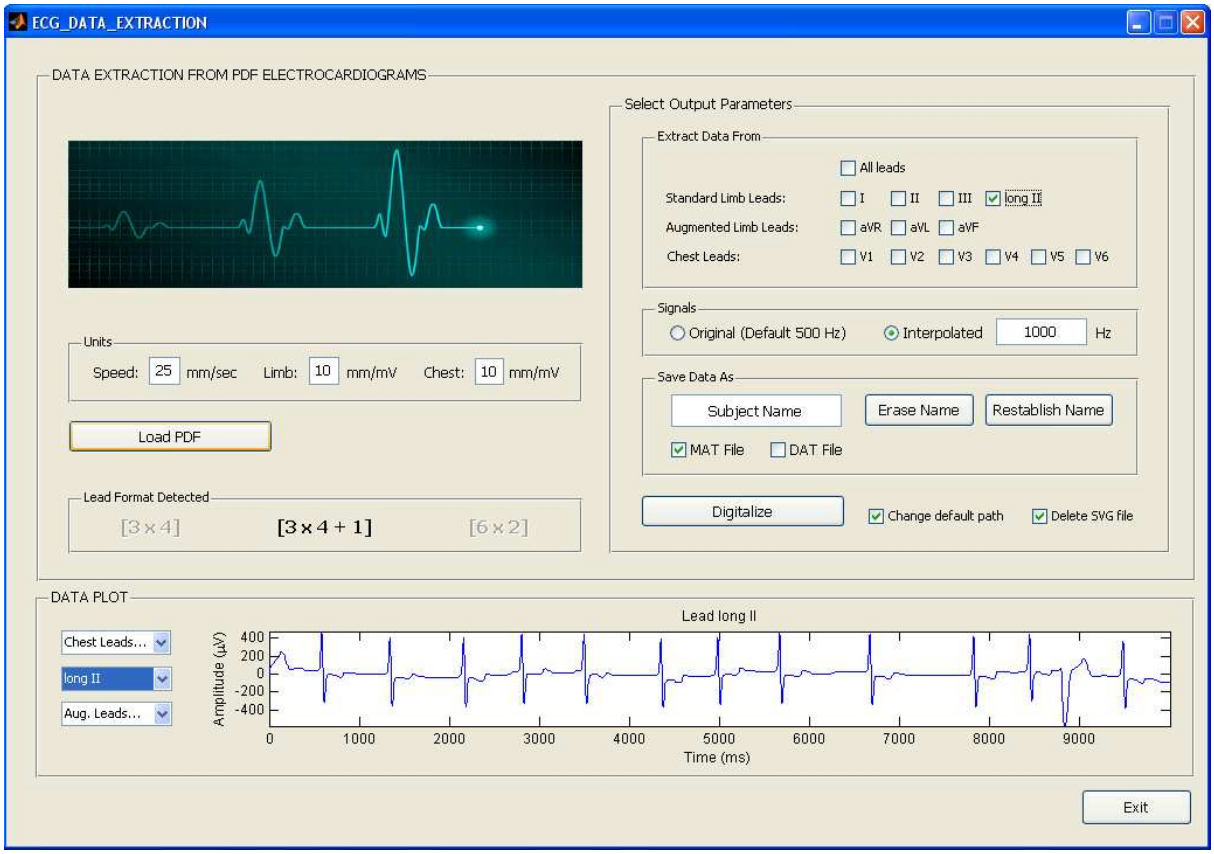
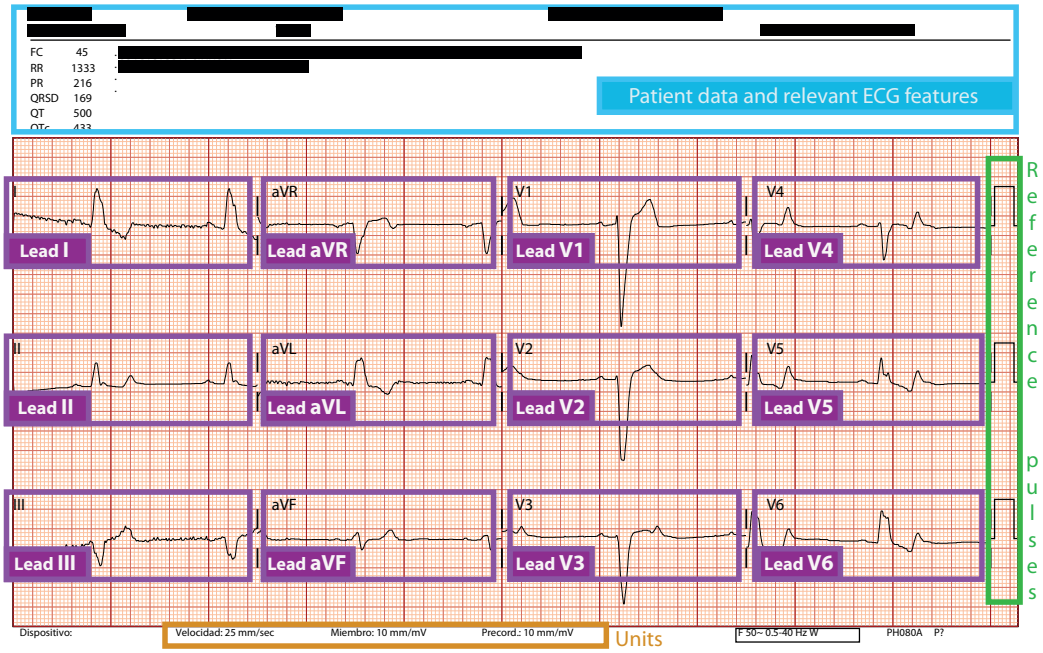
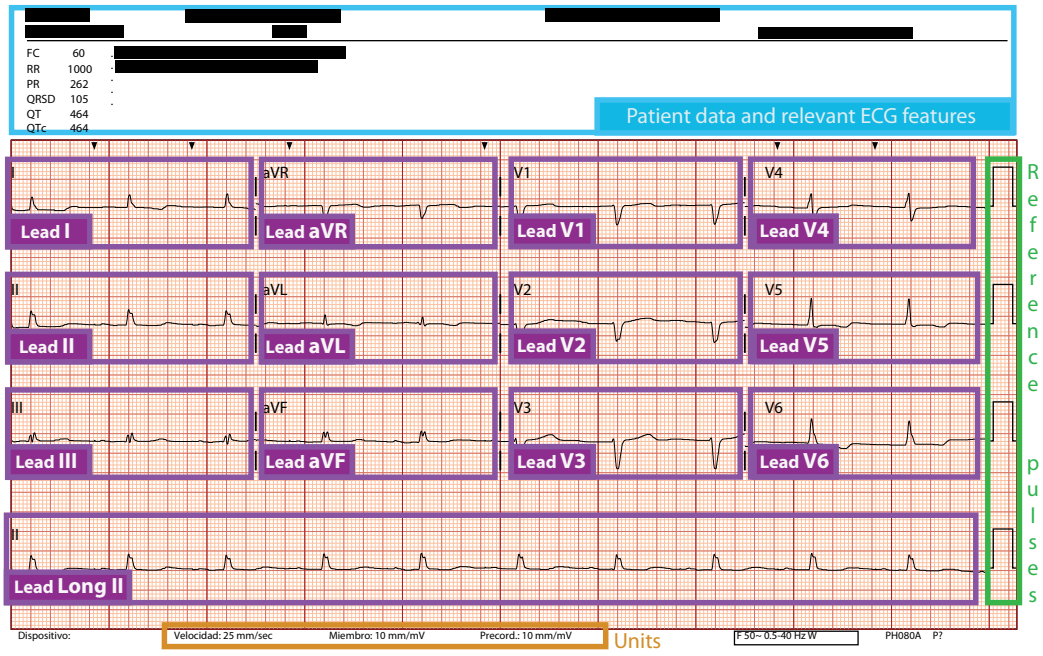


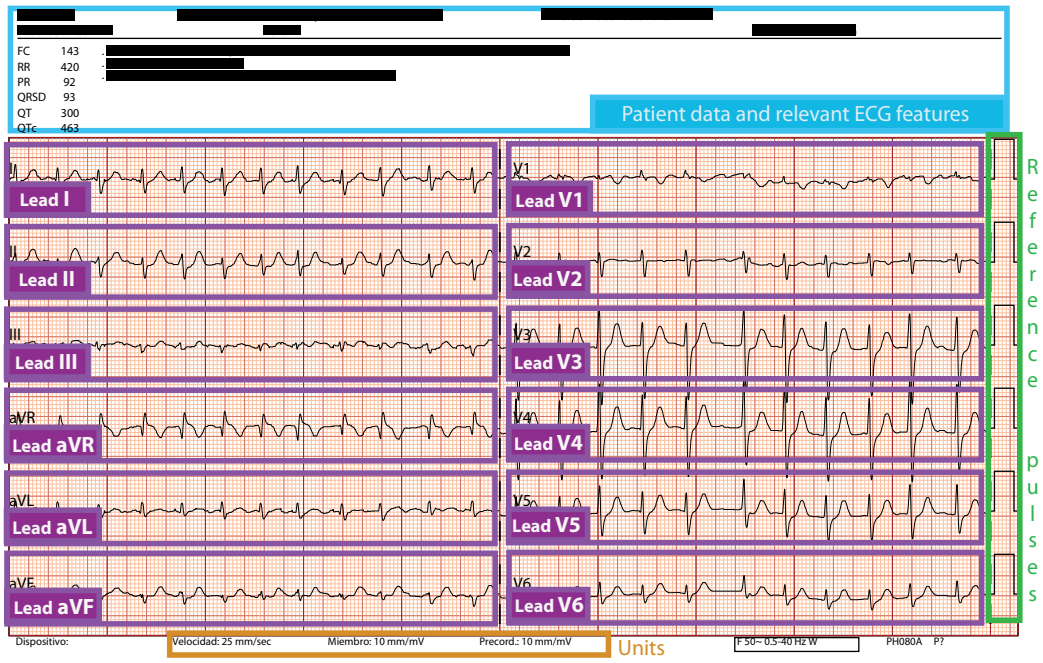
Figure 1: Graphical interface of the presented application for raw data extraction from ECGs in PDF format.


FC 45
RR 1333
PR 216
QRSD 169
QT 500
Patient data and relevant ECG features
Lead I
Lead aVR
Lead V1
Lead V4
Lead II
Lead aVL
Lead V2
Lead V5
Lead III
Lead aVF
Lead V3
Lead V6
Reference pulses
Velocidad: 25 mm/sec
Miembro: 10 mm/mV
Precord.: 10 mm/mV
Units

(a)


FC 60
RR 1000
PR 262
QRSD 105
QT 464
QTc 464
Patient data and relevant ECG features
Lead I
Lead aVR
Lead V1
Lead V4
Lead II
Lead aVL
Lead V2
Lead V5
Lead III
Lead aVF
Lead V3
Lead V6
Lead Long II
Reference pulses
Velocidad: 25 mm/sec
Miembro: 10 mm/mV
Precord.: 10 mm/mV
Units

(b)



(c)

Figure 2: Most common ECG printout formats. Subfigure 2(a) shows 3×4 leads configuration, whereas 2(b) and 2(c) depict $3 \times 4 + 1$ and 6×2, respectively.

Next, the SVG file generated by Inkscape is read to look for the numerical information of leads, depicted under the label $< path >$ and the token $d =$ "M. As commented in section 2.3, every graphical element of the SVG file is indicated using those tokens. Consequently, so as to ensure the differentiation of the grid from the reference pulses and the leads, the application takes into account not only the colour and width of the lines, but also the number of

samples which are contained under each label (8 in the case of the reference pulse, and 2 for each line of the grid). Afterwards, once the numerical data of each lead has been extracted, the application translates this coordinates to the origin by subtracting the abscissa and ordinate coordinates of the corresponding reference pulse for each lead. For example, in case of $3 \times 4 + 1$ lead configuration, content of leads I, aVR, V1 and V4 are relocated using the first reference pulse information (Figure 2(b)).

Thereafter, based on the information content of the paper speed and the leads scale textfields, the application converts the extracted coordinates of each sample (which are expressed in pixels) to true values in milliseconds and μV.

At the right top half of the interface, the user can indicate the output parameters of the extracted data file. In case the user needed the leads raw data with a certain frequency sampling, he/she could indicate the desired value in the textfield next to the button *Interpolated*. The user can also choose the leads he wants to extract, the name of the output file (which is the same as the PDF file by default), and the directory where it will be saved, as long as the format to save this data: with .MAT extension (the default binary format for data files in MATLAB) or with .DAT extension (an ASCII

data file whose content can be read just using any text editor application, such as Wordpad, for example, or be plotted with gnuplot).

Finally, at the bottom of the interface, the user can choose and visualize the data extracted from the desired leads. Figure 3 depicts a flowchart with the algorithm and the tasks developed by the application.

3. Discussion

The application presented in this paper has been developed with the purpose of extracting numerical information from ECGs that have been previously recorded as PDF-format files. Although most of our ECGs were acquired using the Philips PageWriter TC50, the presented application is able to obtain numerical data from ECGs stored with PDF format using any model of Philips electrocardiograph, and those from other brands which are based on XML. This is of great value, since this implies a large proportion of ECG machines, due to the fact that Philips (via its acquisition of Hewlett-Packard Medical Products Group) represents one of the largest suppliers of ECG machines in the world.

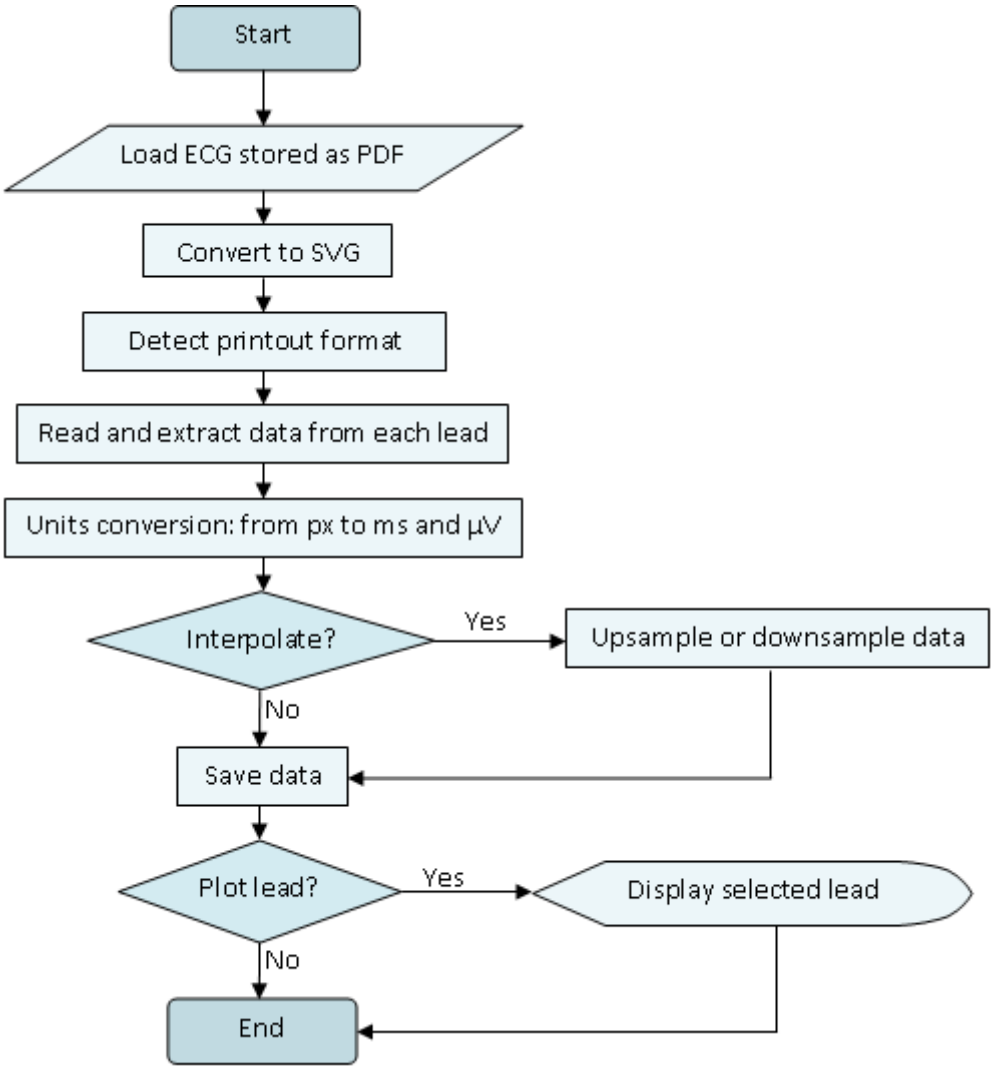


Figure 3: Flowchart of the tasks developed by the application.

The valid configurations for automatic recognition of lead organization are the most common ones: 3×4, $3 \times 4 + 1$ or 6×2, as commented in section 2.4. For our ECGs, the first configuration corresponds to all 12 leads of 2.5 seconds of duration. The second one is the same, but adding information for 10 seconds of lead II (long II). In the last configuration (6×2), all 12 leads have a duration of 5 seconds, displayed along 6 rows at the PDF. In case we wanted to include any other different configuration of leads, software modifications we should do consist of looking for reference pulses and relocate the origin of associated leads according to them.

Regarding the detection of relevant marks in the ECG (such as those produced by pacemakers or indicated by the cardiologist, as shown in Figure 4), it is important to remark that they also appear in the SVG file under the label $< path >$. Thus, in case they could be of interest for subsequent analysis, we could extract their temporal position, just looking for polylines defined by three coordinates with filled style (which define the triangle of the mark).

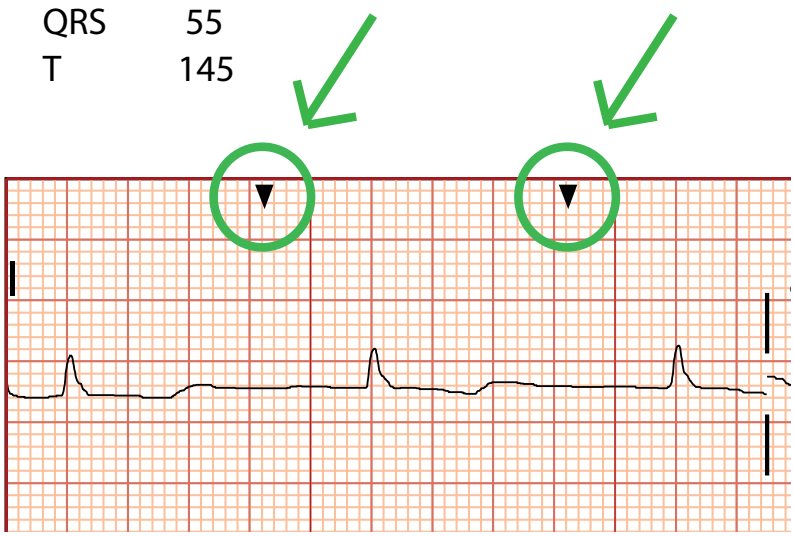


Figure 4: Example of ECG with additional marks.

Concerning the sampling frequency of the raw data, we should take into account that, from SVG files converted from PDF format, we can extract 500 samples per second. However, these data are not uniformly sampled due to the process of storage of the data. In order to to facilitate the subsequent analysis we provide an output datafile with uniformly interpolated data. In addition, in case we needed a different sampling frequency, we could indicate the desired one at the corresponding text box, and the generated output file will be saved by uniformly upsampling or downsampling the original data. Regarding uniformly sampled data reconstruction, we tested with different interpolation methods, such as nearest-neighbour, linear, cubic and splines.

We finally chose linear interpolation method since it presents good results and low computational cost.

Besides, it is important to remind that the presented application carries out the conversion to SVG format by means of the open source software Inkscape. In order to ensure the proper working, the application checks all the existing hard disk drives, so as to find the "Program Files\Inkscape" directory. Moreover, it also differentiates when using a 64 bits operating system or a 32 bits one, by looking for the directory "Program Files (x86)\Inkscape" in that case.

Finally, we are going to analyse the computational load of the whole data recovering for our application. As aforesaid, it has been implemented by using the MATLAB graphical user interface development environment. Detailed computation times spent under a 2.79GHz CPU with 3GB of RAM are shown in Table 1. It can be seen that for lead configuration 6×2, the program spends larger times when extracting data and saving the raw data output file. It is due to the larger number of samples of this configuration (30000 when sampling frequency is 500Hz) compared to other printout formats (15000 and 20000 samples for 3×4 and $3 \times 4 + 1$, respectively). Anyway, on average, computation times are lower than 10 seconds, and they could be

much more smaller (less than 4 seconds per ECG) in case of minimal interface interaction; indeed, the program can be run in batch mode for which the user must indicate just the directory where the ECG PDF-files are stored and the output directory where the transformed files will be saved.

Table 1: Computation times (in seconds) for different tasks of the presented application for a CPU with 3GB of RAM, a 2.79GHz processor and a Windows XP operating system. First column corresponds to time spent on calls to Inkscape (which represents a quarter of the time).

Printout format	Load PDF and SVG conversion	Lead configuration recognition	Data extraction and output data file creation
3×4	2.216	2.223	3.064
$3 \times 4 + 1$	1.737	2.221	5.860
6×2	2.080	2.427	8.586

4. Conclusions

In this paper we have presented an application that extracts raw data from ECG stored with PDF format. It is based on the conversion of PDF files to an XML based data format, the Scalable Vector Graphics (SVG), which describes two-dimensional graphical objects. Thus, the application is

able to obtain pixel coordinates of the polylines that conform the different leads of the ECG, so as to convert them to amplitudes of the ECG signal (in μV). The application can also automatically recognize the most common ECG leads' configuration and can provide an ASCII output data file with different sampling frequencies.

As aforementioned, the application is very useful to recover data from stored ECGs with PDF format. Cardiologists and medical staff usually save the ECG recordings with this format, so as to ensure the later visualization without problems of format compatibility when using proprietary data formats from different ECG devices. The presented application recovers data from these files, allowing to perform numerical analysis and subsequent studies of patient medical evolution along time.

Future work will include the conversion of recovered data from PDF ECGs to standard formats (such as SCP, for example), which would ease the patient history integration of recent ECG recordings with those acquired previously, favouring the adoption of standards and interoperability.

Appendix

The application has been developed using MATLAB software, and its executable standalone application for Microsoft Windows Operating Systems is available at http://personales.upv.es/nuorar/. It runs on any Microsoft Windows Operating System (from XP and later), when the MATLAB Compiler Runtime (MCR) library is installed on the computer.

The user should download and install the open source Inkscape vector graphics editor and the MCR library before running the .exe file. Instructions for installation are detailed at README.pdf file at the above-mentioned website.

We also plan to export this software to Unix and OSX operating systems. As soon as available, these standalone applications will be uploaded to the above-mentioned website to be fully accessible.

Acknowledgements

This work was supported by grants MTM2010-15200 (from the Spanish Ministry of Economy and Competitiveness) and UPV-IIS La Fe, 2012/0468.

References

[1] R.R. Bond, D.D. Finlay, C.D. Nugent, and G. Moore. A review of ECG storage formats. *International Journal of Medical Informatics*, 80(10):681–697, 2011.

[2] J.D. Trigo, A. Alesanco, I. Martínez, and J. García. A review on digital ECG formats and the relationships between them. *IEEE Transactions on Information Technology in Biomedicine*, 16(3):432–444, May 2012.

[3] J. Walker, E. Pan, D. Johnston, J. Adler-Milstein, D.W. Bates, and B. Middleton. The value of health care information exchange and interoperability. *Health Affaires*, pages W5–10–W5–18, January 2005.

[4] OpenECG project. http://www.openecg.net/, Last accessed on June 20, 2013.

[5] C.E. Chronaki, F. Chiarugi, P.J. Lees, M. Bruun-Rasmussen, F. Conforti, R. Ruiz-Fernández, and C. Zywiet. OpenECG: An european project to promote the SCP-ECG standard, a further step towards interoperability in electrocardiography. pages 285–288. Computers in Cardiology, 2002.

[6] B.D. Brown and F. Badilini. HL7 aECG implementation guide. http://www.hl7.org/implement/standards/product_brief.cfm?product_id=102, Last accessed on June 20, 2013.

[7] DICOM (digital imaging and communications in medicine). http://medical.nema.org/, Last accessed on June 20, 2013.

[8] N. Long. Open ECG data standard: Philips medical systems perspective. *Journal of Electrocardiology*, 36:167, 2003.

[9] E.D. Helfenbein, R. Gregg, and S. Zhou. Philips medical systems support for open ECG and standardization efforts. pages 393–396. Computers in Cardiology, 2004.

[10] S. Zhou and E. Helfenbein. OpenECG format: Philips' experience. pages 46–47. 2nd OpenECG Workshop, 2004.

[11] Semantic interoperability for better health and safer healthcare. http://www.empirica.com/publikationen/documents/2009/semantic-health-report.pdf, Last accessed on June 20, 2013.

[12] C.E. Chronaki, F. Chiarugi, A. Macerata, F. Conforti, H. Voss, I. Johansen, R. Ruiz-Fernández, and C. Zywietz. Interoperability in digital

electrocardiography after the openECG project. pages 49–52. Computers in Cardiology, 2004.

[13] M.J.B. van Ettinger, J.A. Lipton, M.C.J. de Wijs, N. van der Putten, and S.P. Nelwan. An open source ECG toolkit with DICOM. pages 441–444. Computers in Cardiology, 2008.

[14] P. Marcheschi, A. Mazzarisi, S. Dalmiani, and A. Benassi. ECG standards for the interoperability in patient electronic health records in italy. pages 549–52. Computers in Cardiology, 2006.

[15] X. Li, V. Vojisavljevic, and Q. Fang. An XML based middleware for ECG format conversion. pages 1691–1694. 31st Annual International Conference of the IEEE EMBS, 2009.

[16] J.D. Trigo, A. Kollmann, A. González, D. Hayn, A. Alesanco, G. Schreier, and J. García. Plataforma para la integración y la gestión homogénea de formatos de electrocardiografía. XXVIII Congreso Anual de la Sociedad Española de Ingeniería Biomédica CASEIB, 2010.

[17] J.D. Trigo, I. Martínez, A. Alesanco, A. Kollmann, J. Escayola, D. Hayn, G. Schreier, and J. García. An integrated healthcare information system for end-to-end standardized exchange and homogeneous management of

digital ECG formats. *IEEE Transactions on Information Technology in Biomedicine*, 16(4):518–529, July 2012.

[18] Philips pagewriter TC50 cardiograph. http://www.healthcare.philips.com/main/products/cardiography/products/cardiograph/pagewritertc50.wpd, Last accessed on June 20, 2013.

[19] Scalable vector graphics (SVG). http://www.w3.org/TR/SVG/, Last accessed on June 20, 2013.

[20] A. de la Rosa and J.A. Senso. La dualidad texto-imagen en SVG (scalable vector graphics): nuevas posibilidades para la descripción de información gráfica. *El Profesional de la Información*, 12(5):377–398, September 2003.

[21] Inkscape: Open source scalable vector graphics editor. http://inkscape.org/, Last accessed on June 20, 2013.