Using the Gamera framework for the recognition of cultural heritage materials

Conference Paper · January 2002

DOI: 10.1145/544220.544223 · Source: DBLP

7 authors, including:

Ichiro Fujinaga
McGill University
165 PUBLICATIONS **2,622** CITATIONS
SEE PROFILE

Sayeed Choudhury
Johns Hopkins University
61 PUBLICATIONS **436** CITATIONS
SEE PROFILE

Tim DiLauro
Johns Hopkins University
23 PUBLICATIONS **207** CITATIONS
SEE PROFILE

Using the Gamera Framework for the Recognition of Cultural Heritage Materials

Michael Droettboom
mdboom@peabody.jhu.edu

Ichiro Fujinaga
ich@peabody.jhu.edu

Karl MacMillan
karlmac@peabody.jhu.edu

G. Sayeed Chouhury
sayeed@jhu.edu

Tim DiLauro
timmo@jhu.edu

Mark Patton
mpatton@jhu.edu

Teal Anderson
emailteal@jhu.edu

Milton S. Eisenhower Library
Johns Hopkins University
3400 N. Charles St.
Baltimore, MD 21218

ABSTRACT

This paper presents a new toolkit for the creation of customized structured document recognition applications by domain experts. This open-source system, called Gamera, allows a user, with particular knowledge of the documents to be recognized, to combine image processing and recognition tools in an easy-to-use, interactive, graphical scripting environment. Gamera is one of the key technology components in a proposed international project for the digitization of diverse types of humanities documents.

Categories and Subject Descriptors

I.4 [**Computing Methodologies**]: Image processing and computer vision

General Terms

Algorithms, Human Factors, Languages

Keywords

document recognition, development tools

1. INTRODUCTION

Given the potential of digital libraries, it is understandable that a great deal of attention and resources has been allocated toward digital resources. However, it is important to realize that vast amounts of knowledge, particularly in the humanities, remains in print format. With increasing reliance on digital resources, efficient ingestion of print materials into digital libraries becomes vitally important. Without the presence of digitized and processed print materials, digital libraries might not represent the full range of human knowledge. Additionally, to unleash their full potential, digital libraries should include materials in multiple rich formats, including image, text, and sound. This transition from a "data poor" to a "data rich" environment could lead to new modes of inquiry, research, and instruction, especially for the humanities.

Currently, usable technology primarily focuses on modern business documents and is not ideally suited for recognition of cultural heritage materials. Rather than allowing commercial needs and technology available to constrain the choices of materials for ingestion, the technology for ingesting these materials should be improved.

As a part of the solution, we propose a new toolkit for the creation of domain-specific structured document analysis applications by domain experts. The goal is to leverage the user's knowledge of the target documents to create custom applications rather than attempting to meet the needs of diverse users with a monolithic application. This system, called Gamera, allows a knowledgeable user to combine image processing and recognition components in an easy to use, interactive, graphical scripting environment. The applications created by the user are suitable for use in a large-scale digitization project because they can be run in a batch processing mode and easily integrated into a digitization framework.

This paper will give an overview of Gamera, its applications, describe the user environment, and briefly discuss some of its technical features.

2. BACKGROUND AND GOALS

Gamera is being developed as part of an international, multidisciplinary project to build a data capture framework and testbed for cultural heritage materials. The primary goal of Gamera is to provide enough general components so that applications for the analysis and recognition of new types of documents can be developed by domain experts with a minimum of effort.

JCDL'02, July 13–17, 2002, Portland, Oregon, USA

Gamera grew out of a flexible optical music recognition (OMR) system [6] developed as part of the Lester S. Levy Collection digitization project. The Collection, part of the Special Collections of the Milton S. Eisenhowever Library at the Johns Hopkins University, comprises nearly 30,000 pieces of music which correspond to nearly 130,000 sheets of music and associated cover art. The original impetus to generalize the OMR system came from the realization that recognizing music also requires recognition of the text on the page. It was hoped that an existing optical character recognition (OCR) system would be suitable for this task. Unfortunately, early trials of existing systems revealed that there are many problems with the current generation of OCR software, which are primarily designed for modern business documents. However, by using the same learning-based technology used in our OMR system, we were able to achieve good results on text, and at the same time have an integrated and flexible system.

Once embarking on the Gamera project, it was soon clear that there were many other types of documents, particularly in the humanities, that would benefit from a generalized and extensible recognition framework. Document analysis tools are generally not available for documents in ancient languages that contain non-standard printing, or for a variety of reasons differ from common documents. The lack of commercial tools specifically targeted at these types of documents is not likely to change in the future because of the limited market for them.

There are other academic projects that explore the connection between cultural heritage materials and technology. For example, the Digital Atheneum Project [3] focuses on the development of algorithms and techniques to digitize damaged manuscripts in order to restore content that may no longer be visible. Ultimately, the project team intends to develop a toolkit that will allow humanists to create customized digital editions of manuscripts. In addition, The Cervantes Project at Texas A&M [9] is building an Electronic Variorum Edition of a well-known Don Quixote, ultimately providing a range of electronic resources for the study of Cervantes. Though with somewhat different foci, projects at CalTech [14] and UMass-Amherst [16] examine cursive OCR technology for recognition and indexing of handwritten documents. Gamera differs from these projects in that it provides a framework for a wide range of document analysis problems, rather than an end user interface or a specific set of algorithms. Additionally, Gamera is part of an overall effort to develop a workflow management system for ingestion of cultural heritage materials. Ideally, Gamera would be able to integrate the results of these other research efforts.

We intend to use Gamera to develop applications for the recognition of documents in a number of diverse collections, including medieval manuscripts, lute tablature, Greek text, handwritten text, and architectural diagrams. There are many collections that would benefit from this technology and would represent a diverse range of document analysis challenges. The following collections, including those from partner institutions (University of Oxford, Edinburgh University, Tufts University and the British Library), were chosen specifically to consider a wide range of document analysis issues.

- Roman de la Rose (`http://rose.mse.jhu.edu/`): Medieval French manuscripts
- Electronic Corpus of Lute Music (`http://www.kcl.ac.uk/kis/schools/hums/music/ttc/IR_projects/ECOLM/`): Lute tablature
- Perseus Digital Library (`http://www.perseus.tufts.edu`): Collections from antiquity through the twentieth century, including Greek, Latin, pre-modern, and modern with non-standard typesetting
- Statistical Accounts of Scotland (`http://edina.ac.uk/StatAcc/`): printed in 18^{th} and early 19^{th} century typefaces
- Celtic and Medieval Manuscripts Project (`http://www.image.ox.ac.uk/`): Manuscripts of Celtic origin
- Bodleian Library Broadside Ballads (`http://www.bodley.ox.ac.uk/ballads/`): Ballads from the 16^{th} to 20^{th} centuries
- John Johnson Collection of Printed Ephemera (`http://www.bodley.ox.ac.uk/toyota/`, `http://www.ilrt.bris.ac.uk/jidi/col_john.html` and `http://www.bodley.ox.ac.uk/johnson/`): Includes political prints, cartoons and advertising

3. ARCHITECTURE OVERVIEW

Gamera is primarily a toolkit for the creation of recognition applications by domain experts. It is composed of modules (similar to components or plug-ins), written in a combination of C++ and Python, that are combined in a very high-level scripting environment to form an application. The overall design is inspired by systems like MathWorks Matlab and spreadsheet macros, and similar to Feature Center [19], CVIPtools [22], and HUE [8]. However, there are some important differences between Gamera and these systems that make it more suitable for cultural heritage materials. While Feature Center is intended for document analysis, it is designed for technical documents. CVIPtools, while an important inspiration for Gamera, is primarily a teaching tool. HUE is perhaps most similar to Gamera. Both are structured-document analysis application development environments. Gamera, however, is targeted at a potentially less technically sophisticated audience, and it includes many important features that allow it to be more extensible and easy-to-use. Specifically, Gamera has a modern object-oriented design and a more flexible component model. Additionally, Gamera has an extensive GUI that includes a flexible training and ground-truth creation interface suitable for use by non-programmers. Many of the advantages of Gamera are realized through the use of the more modern object-oriented languages C++ and Python, in contrast to HUE's use of C and Tcl/Tk.

In Gamera, modules perform one of five document recognition tasks:

1. Pre-processing
2. Document segmentation and analysis
3. Symbol segmentation and classification
4. Syntactical or structural analysis

5. Output

Each of these tasks can be arbitrarily complex, involve multiple strategies or modules, or be removed entirely depending on the specific recognition problem. Additionally, keeping with the toolbox philosophy of Gamera, the user of the system has access to a range of tools that fall within the general category of these tasks. The actual steps that make up the recognition process are completely controlled by the user.

In addition to flexibility, Gamera also has several other advantages that are important to the Levy project and to large-scale digitization projects in general. These are:

1. A batch processing mode to allow many documents to be recognized without user intervention
2. Open source code and standards-compliance so that the software can interact well with other parts of a digitization framework
3. Platform independence, running on a variety of operating systems including Unix, Microsoft Windows and Apple MacOS
4. Rich user interface components for development and training
5. Recognition confidence output so that collection managers can easily target documents that need correction or different recognition strategies

3.1 Pre-processing

Pre-processing involves standard image-processing operations including noise removal, blurring, de-skewing [5], contrast adjustment, sharpening, binarization [17], and morphology. Any number of these operations may be necessary to take a raw image and prepare it for recognition.

Many documents, particularly cultural heritage materials, will depend on this part of the recognition process to ensure good overall performance of the system.

Discoloration of the documents, created by aging or inferior reproduction, makes binarization difficult and often requires locally-adaptive algorithms [21]. Additionally, broken lines often cause problems in the segmentation of symbols. Experiments suggest that simple blurring or morphology [18] may help with these difficulties.

3.2 Document segmentation and analysis

Before the symbols of a document can be classified, an analysis of the overall structure of the document may be necessary. The document segmentation and analysis process is designed to analyze the overall structure of the document, segment it into sections, and perhaps identify and remove elements [11, 24]. For example, in the case of music recognition, it is necessary to identify and remove the staff lines in order to be able to properly separate the individual symbols. Similarly, text documents may require the identification of columns, paragraphs, lines, or tables.

3.3 Symbol segmentation and classification

The segmentation [4], feature extraction [20], and classification of symbols is the core of the Gamera system. The current implementation provides tools for the creation of simple heuristic classifiers, template-based image matching, and a learning classifier using the k-nearest neighbor (k-NN) algorithm [7] enhanced with a genetic algorithm (GA) [12]. Other possible classification algorithms include neural-nets, decision trees, or hidden Markov models. The use of both learning and heuristic classifiers allows for the balance of flexibility, training time, and recognition speed.

3.4 Syntactical or structural analysis

This process reconstructs a document into a semantic representation of the individual symbols. Examples include combining stems, flags, and noteheads into musical notes, or grouping words and numbers into a table [6]. Obviously, this process is entirely dependent on the type of document being processed and is a likely place for large customizations by knowledgeable users. Fortunately, generalized tools are provided by Gamera, including tools for structural and syntactic analysis, and theorem and constraint solvers.

3.5 Output

The output functions convert either the raw symbols or the post-structural interpretation data into a suitable format for storage.

For the former, Gamera takes advantage of the rich set of eXtensible Markup Language (XML) tools provided by Python to store the symbol data. We are developing both an XML Schema and an XML DTD for this format. The design is extensible so that as other aspects of document analysis are developed (e.g. zoning of the image) new elements can be added in a straightforward way.

Output of the latter kind of data, i.e. post-structural interpretation data, is deliberately left open-ended, as different domains will have different requirements for an ultimately useful file format. For example, the GUIDO [13] file format is used for symbolic music representation by our Gamera-based music interpretation application.

4. USER ENVIRONMENT

Another important goal of Gamera is that it should be as usable as possible. It is not enough to assist experienced programmers: Gamera puts the ability to develop recognition processes in the hands of those who understand the documents best, the domain experts. At present, the usability is achieved by the combination of an easy-to-learn scripting language coupled with a graphical user interface (Section 4).

4.1 Scripting environment

4.1.1 Ease of use

Perhaps the most important aspect of the Gamera scripting environment is ease of use by users with limited computer programming experience. As previously stated, the targeted user is a person with expert knowledge of the source documents, who may or may not have computer programming experience. In order to meet this goal, Python was chosen as the foundation and extensions were written to be as easy to use as possible.

Python is a popular, general-purpose scripting language [10]. It has been used as a first language with considerable success [1]. For this reason, we believe that Python is a good choice for the basis of the scripting environment. The existence of books and tutorials [23] about the language also means that there is more help available to users than there would be for a custom scripting language.

In order to transform Python from a general-purpose scripting language to a scripting environment specifically tailored to the needs of document analysis, a set of extensions were written in a combination of Python and C++. The following example demonstrates the high-level and transparent nature of Gamera scripts.

```
# Load an image
image = load_image('example.tiff')

# Convert to binary using the Otsu thresholding
# algorithm
image.otsu_threshold()

# Remove staves and store information about them
staves = image.find_and_remove_staves()

# Perform recognition on the image - this is a
# two step process.  First, the image is
# segmented and then the k-NN classifier is
# used on the individual symbols.
symbols = image.cc_analysis()
classifier = Classifier(symbols, 'database.knn')
glyphs = classifier.auto_classify_all()

# Interpret the symbols with the Optical Music
# Interpretation object
import omi
omi.run(glyphs, image)

# Output to GUIDO and MIDI
omi.save('example.gmn', 'guido')
omi.save('example.mid', 'midi')
```

4.1.2 Flexibility

The flexibility of the scripting environment is facilitated by the choice of Python. Because Python is a general-purpose programming language, a large portion of the system can be implemented directly in Python itself. In general, only those algorithms that need direct access to image pixels and improved efficiency are written in C++. These modules can be added to Gamera without recompiling the entire system. This means modules can be easily shared over the internet with other Gamera users.

4.1.3 Extensibility

Despite the flexibility of the scripting environment, not all algorithms can be suitably or adequately implemented in Python. For this reason, a C++ module system for use by experienced programmers has been developed. Some of the features of this system are:

1. Automatic binding of C++ module code to Python
2. Runtime integration of C++ modules to Python matrix classes
3. Abstraction of the data storage format of image data using C++ templates to allow convenient access to compressed images
4. A flexible programming interface that allows the easy conversion of existing C and C++ code, which may use a variety of syntax to access the image data

In many cases, these plug-in modules do not need to be written specifically for the Gamera system, but can be inherited from other image processing libraries. For example, many of the thresholding plugins in Gamera use minimally modified source code from Xite [2]. Additionally, full support of the algorithms in the VIGRA computer vision library is provided [15]. Finally, support for the standard multi-dimensional array package for Python, Numeric Python, will be provided in the next release. This will allow algorithms from Numeric Python to operate on Gamera images and algorithms from Gamera to use Numeric Python arrays. By carefully designing the Gamera image classes, this borrowing of code is possible with a minimum of effort.

4.2 Graphical interface

Since document recognition is an inherently visual problem, a graphical user interface is included to allow the application developer to experiment with different recognition strategies. At the core of the interface is the console window (Figure 1) which allows the programmer to run code interactively and control the system either by typing commands or using menus. All commands are recorded in a history, which can later be used for building automatic batch scripts. The interface also includes a simple image viewer, image analysis tools (Figure 2), and a training interface for the learning classifiers (Figures 3 and 4). The interface can be easily extended to include new elements as modules are added to Gamera. Again, like the scripting environment, the graphical interface is created with standard tools entirely in Python, allowing users to extend and modify the system.

We plan to use our own experience with the system and the assistance of our on-team usability expert to further refine the system. Gamera aims to be usable—designed according to the principles of usability which include effectiveness, efficiency and satisfaction. Usability testing and research throughout the development of Gamera will foster the design of usable interfaces to accommodate a diversity of needs, skill levels, and disciplines.

5. CONCLUSION

Increasing the body of digitized and interpreted content will open up new worlds of scholarly research and a better understanding of our cultural heritage. Gamera is set to play an important role since it decreases development time and costs and places power in the hands of those with the most expertise.

6. ACKNOWLEDGEMENTS

The second phase of the Levy Project is funded through the National Science Foundation's Digital Library Initiative 2 (Award #9817430), an Institute of Museum and Library Services National Leadership Grant, and support from the Levy family.

We also wish to thank our partners in the overall effort to build a data capture framework and testbed for cultural heritage materials: Greg Crane (Tufts University), David Price and Norbert Lossau (University of Oxford), Richard Ovenden (Edinburgh University), and Cynthia Requardt and Brian Harrington (Johns Hopkins University).

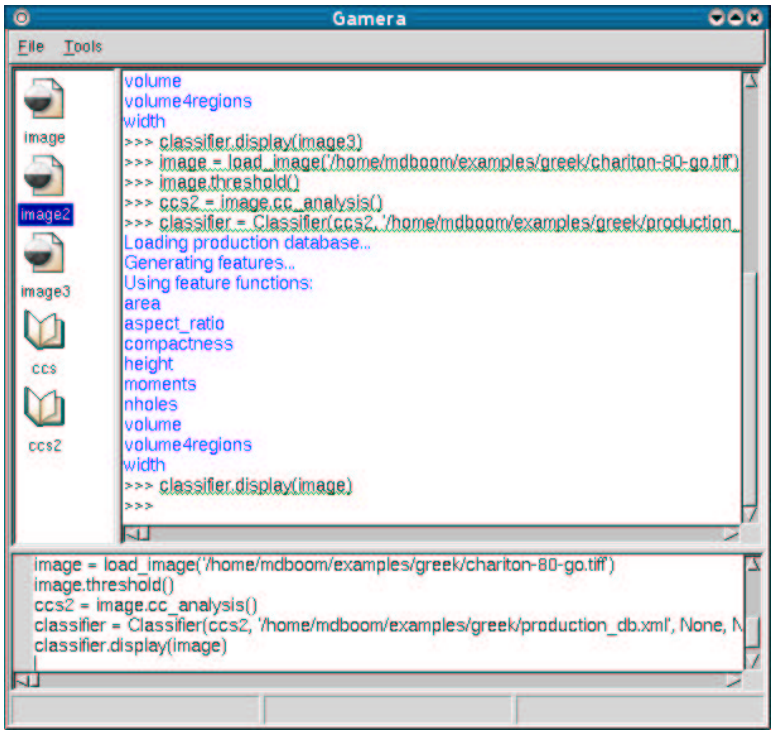

Figure 1: The Gamera console window.

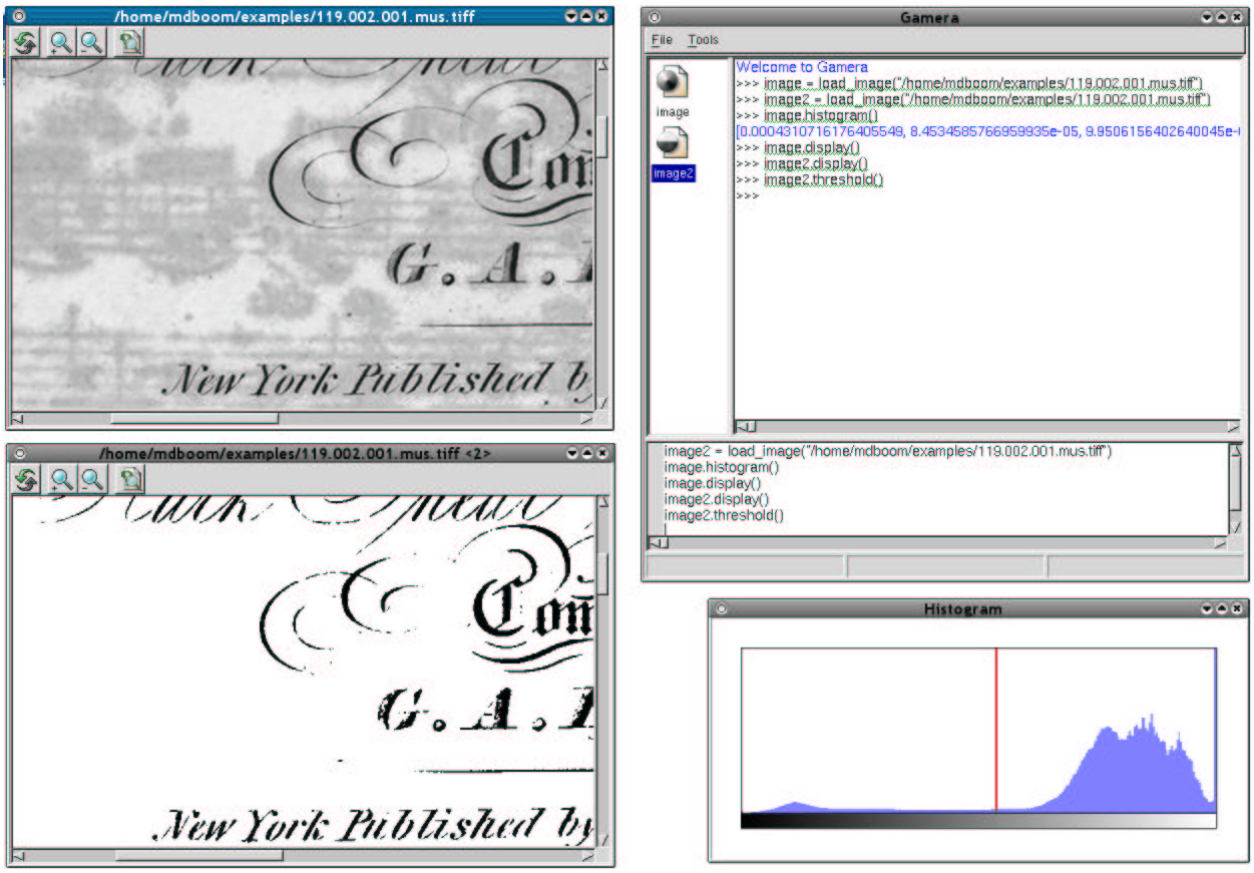

Figure 2: An example Gamera session, showing the image viewer, image analysis tools, and the Gamera console.

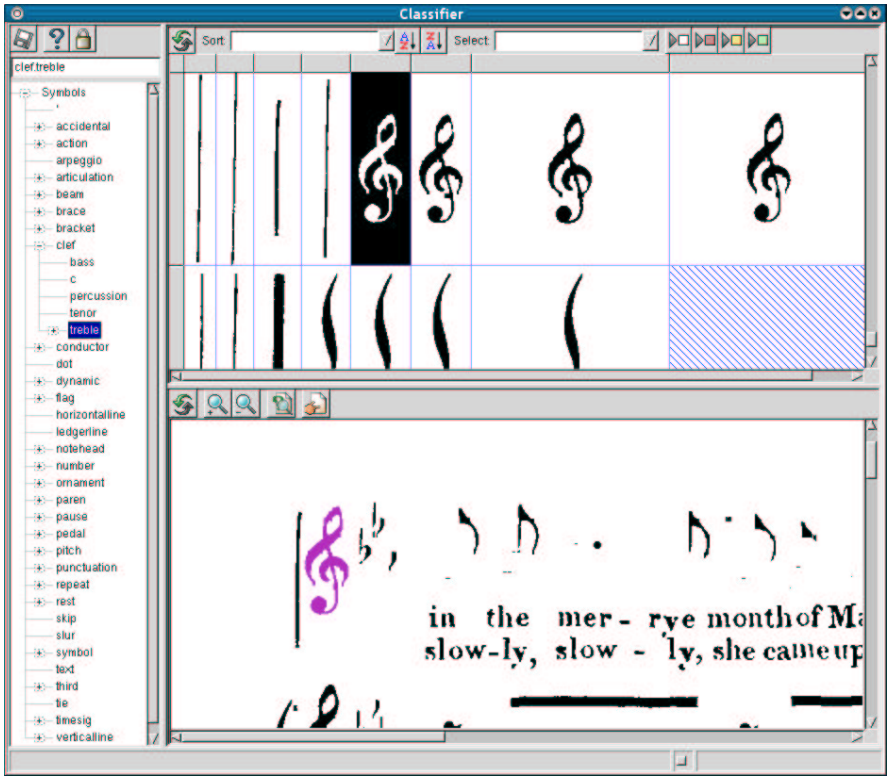

Figure 3: An interactive training session, classifying musical symbols.

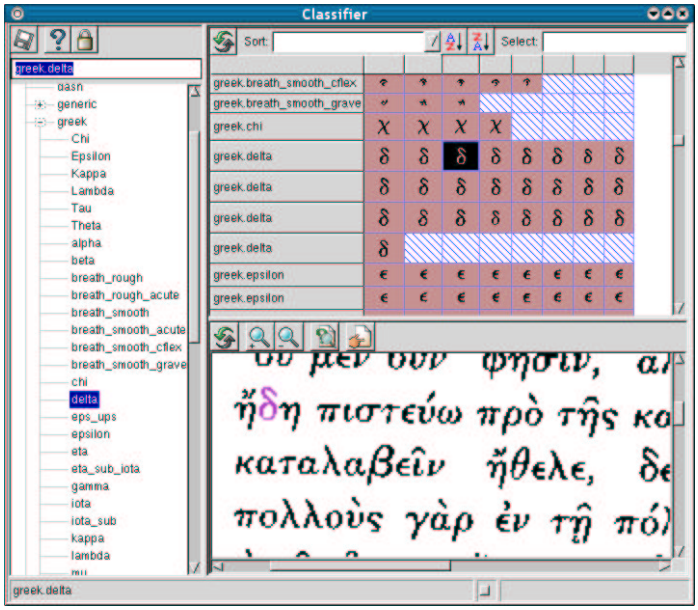

Figure 4: An interactive training session, classifying Greek characters.

7. REFERENCES

[1] L. Berehzny, J. Elkner, and J. Straw. Using Python in a high school computer science program: Year 2. In *Proceedings of the Ninth International Python Conference*, pages 217–223, 2001.

[2] S. Bøe. *XITE: Programmer's manual*. Image Processing Laboratory, Department of Informatics, University of Oslo, 1998.

[3] M. S. Brown and W. B. Seales. The digital atheneum: New approaches for preserving, restoring and analyzing damaged manuscripts. In *Proceedings of the First ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 437–443, New York, 2001. ACM Press.

[4] R. G. Casey and E. Lecolinet. A survey of methods and strategies in character segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7):690–706, 1996.

[5] R. Cattoni, T. Coianiz, S. Messelodi, and C. M. Modena. Geometric layout analysis techniques for document image understanding: A review. Technical Report 9703-09, ITC-IRST, 1998.

[6] G. S. Choudhury, T. DiLauro, M. Droettboom, I. Fujinaga, and K. MacMillan. Strike up the score: Deriving searchable and playable digital formats from sheet music. *D-Lib Magazine*, 7(2), 2001.

[7] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[8] C. Cracknell and A. C. Downton. A handwriting understanding environment (HUE) for rapid prototyping in handwriting and document analysis research. In *Proceedings of the Fifth International Conference on Document Analysis and Recognition (ICDAR)*, pages 362–365. IEEE Computer Society, September 1999.

[9] R. Furuta, S. Hu, S. Kalasapur, R. Kochumman, E. Urbina and R. Vivancos. Towards an electronic variorum dition of Don Quixote In *Proceedings of the first ACM/IEEE-CS joint conference on Digital Libraries*, pages 444–445. ACM Press, 2001.

[10] A. Gauld. *Learn to program using Python*. Addison-Wesley, Boston, 2000.

[11] R. M. Haralick. Document image understanding: Geometric and logical layout. In *Proceedings of IEEE Computer Society Computer Vision and Pattern Recognition*, pages 385–390, 1994.

[12] J. H. Holland. *Adaptation in natural and artificial systems*. University of Michigan Press, Ann Arbor, 1975.

[13] H. H. Hoos and K. Hamel. GUIDO music notation version 1.0: Specification part i, basic GUIDO. Technical Report 20, Technische Universität Darmstadt, 1997.

[14] P. Keaton, H. Greenspan, and R. Goodman. Keyword spotting for cursive document retrieval. In *Proceedings of the IEEE Workshop on Document Image Analysis, in conjunction with the Conference on Computer Vision and Pattern Recognition*, San Juan, Puerto Rico, 1997.

[15] U. Köthe. Reusable software in computer vision. In B. Jähne, H. Haußecker, and P. Geißler, editors, *Handbook on Computer Vision and Applications*, volume 3, pages 149–178. Academic Press, 1999.

[16] R. Manmatha, C. Han, E. M. Riseman, and W. B. Croft. Indexing handwriting using word matching. In *Proceedings of the First ACM International Conference on Digital Libraries*, pages 151–159, 1996.

[17] B. Sankur and M. Sezgin. Image thresholding techniques: A survey over categories. *Pattern Recognition*, 2001. Under review.

[18] J. P. Serra. *Image analysis and mathematical morphology*. Academic Press, London, 1982.

[19] R. H. Thibadeau, R. Romero, and D. S. Touretzky. Feature center: Getting the picture from documents and drawings. Technical report, Robotics Institute, Carnegie Mellon University, 1995.

[20] D. Trier, A. K. Jain, and T. Taxt. Feature extraction methods for character recognition: A survey. *Pattern Recognition*, 29(4):641–662, 1996.

[21] O. D. Trier and T. Torfinn. Evaluation of binarization methods for document images. In *Proceedings of IEEE Computer Society Pattern Analysis and Machine Intelligence*, pages 312–315, 1995.

[22] S. E. Umbaugh. *Computer vision and image processing: A practical approach using CVIPtools*. Prentice Hall, Upper Saddle River, NJ, 1998.

[23] G. van Rossum and F. L. Drake. *Python tutorial*. iUniverse, Campbell, CA, 2000.

[24] B. A. Yanikoglu and L. Vincent. Pink panther: A complete environment for ground-truthing and benchmarking document page segmentation. *Pattern Recognition*, 31(9):1191–1204, 1998.