7th International Conference on Document Analysis and Recognition (ICDAR), Edinburgh (Scotland), 2003

The Callpaper Concept: Turning Paper into Computer Terminals

Stefan Jaeger, Masaki Nakagawa
Tokyo University of Agri. & Tech., Department of Computer Science
2-24-16 Naka-cho, Koganei-shi, Tokyo 184-8588, Japan
stefan@hands.ei.tuat.ac.jp, nakagawa@cc.tuat.ac.jp

Abstract

Despite all the attempts of modern information technology to supplant paper by sophisticated electronic devices, paper is still omnipresent in virtually any office environment. We have learned over the last decade that paper is an indispensable part of our daily work and that even the latest computers are not able to mimic its functionality. Today, paper and computer are coexisting side by side without interacting with each other, both having their specific advantages. This paper presents a concept for bridging the gap between computer and paper by combining the advantages of both electronic and paper world. The implementation described in this paper augments standard paper with features of modern information technology. Moreover, it allows interfacing computers using pen and paper.

1. Introduction

The ease of use provided by the classic pen-and-paper "interface" is unsurpassed. Except for writing long text passages, most of us regard it as the most convenient way of expressing, conveying, and storing one's thoughts and ideas. However, pen and paper are hardly supported by current information technology. The integration of paper-based devices and modern computers has been hampered by two problems: the difficulty of recording handwriting without affecting the natural look and feel of pen and paper, and the insufficient recognition rates provided by handwriting recognizers. However, considerable progress has been made over the recent years. While new and highly sophisticated pen-based hardware is announced almost monthly nowadays, handwriting recognition rates are steadily improving and will soon reach a level of common acceptance. These developments pave the way for a better integration of pen and paper into the daily workflow.

The main idea of the concept described in the following sections is to generate a so-called model file for all paper documents expecting handwritten input. A model file describes the structure of a document and provides the context knowledge necessary for handwriting recognition. Moreover, it contains information about how the recognized data should be processed, including its final destination. A unique ID printed on each document specifies the corresponding model file. Writers can access the model file of a document via the Internet, under the number specified on the document. In practice, this means calling the (phone) number of a server dispatching model files, which motivates the name of the concept "Callpaper" — a very transparent concept for the writer. The Callpaper concept nicely supports the simultaneous generation of both paper copies and electronic copies of the same document, without imposing any additional load on a writer. It thus introduces the benefits of information technology into paper-based processes without affecting the traditional workflow. Needless to say, that handwriting recognition plays a crucial role here [5, 3]. However, the concept is independent of the hardware employed for capturing handwriting. The latest hardware devices are able to capture handwriting directly on paper, while retaining the look and feel of the common pen-and-paper interface. In fact, the latest hardware features enable a multi-dimensional paper space spanning many different features, such as writer position and writer identification.

This paper is structured as follows: Section 2 discusses the advantages and disadvantages of paper. Section 3 presents different hardware solutions for capturing handwriting on paper. A client-server implementation of the proposed concept is described in Section 4. Section 5 illustrates a typical application flow. The logical space spanned by modern hardware components is described in Section 6. Section 7 gives an overview of our current implementation status. Finally, the last section concludes this paper with an outlook on future plans.

2. Pen and Paper

What are the advantages and disadvantages of paper when compared to the devices of modern information technology? Though this seems to be an easy question, we must now admit that we have not answered it properly during the recent years, considering the fact that many of us were predicting an ever decreasing paper consumption caused by the technological progress. In fact, the advantages of paper have been grossly underestimated, in particular when it is utilized in combination with pens. Of course, the disadvantages of paper are obvious: limited feedback on user input, limited access and search functions, slow transfer et cetera. And yet, paper combines features that no other hardware device can offer. Paper has a high resolution and is thus pleasant to read. Paper can be easily folded and crushed. It is very cheap, a fact that provides us with a strikingly simple way of deleting data, namely disposing old paper and replacing it by new one. Paper supports fast input of both textual as well as graphical data. Moreover, it does not require tedious eye-hand coordination since the cursor is directly under the pen tip. Hence it puts less cognitive load on the user than the common mouse interface.

The limited feedback capability is probably the most important drawback of paper compared to hardware devices trying to mimic its features, such as high-resolution flat screens. This limited feedback capability requires different interface designs and calls for hybrid solutions with at least rudimentary visual or aural feedback.

3. Recording handwriting

It has long been possible to capture handwriting on paper by utilizing a standard digitizer lying beneath the paper. However, that is not quite what we want. We need a hardware device that allows us to retain the look and feel of paper we are so accustomed to. And, of course, we need on-line handwriting and not just a simple scan of an off-line document. On-line information describes the dynamics of the writing process, providing important information for recognition and segmentation. The ideal device would be a pen capable of capturing handwriting, recognizing handwriting, and sending recognition results to the proper destination via a wireless Internet connection. While there is little doubt that this is technically possible, it will still take a while before we will see such functionality integrated in a small and handy device like a pen. Nevertheless, the current hardware realizations are already very impressive, though they are merely approximations of this ideal. What we will definitely see in the near future are hybrid systems that distribute their functions among several components. A mobile phone in combination with a pen attached to the phone via a wireless connection is such a hybrid solution for instance. While the pen captures the handwriting, the mobile phone transmits the (recognized) data over the Internet.

The following subsections describe the current technical approaches for capturing on-line handwriting on paper. Note that in order to capture on-line handwriting it is necessary to capture not only relative coordinates but the absolute position of the pen tip during writing. This is mainly because the system would otherwise lose the position of the pen once it is lifted from the paper (pen-up movement). This difficulty excludes some of the existing standard tracking techniques, such as most of the techniques utilized for tracking mouse movements. Moreover, we must consider that paper itself is not a fixed medium but may also move while the pen writes or hovers over its surface.

3.1. Patterned paper

At the moment, the most flamboyant but also most controversial approach for capturing handwriting is patterned paper [1]. This approach is based on a barely perceptible pattern printed on all documents allowing handwritten input. The pattern serves two main purposes: First, it assigns a unique code to each pen position on the paper. Second, it allows an automatic identification of the paper, which in turn enables document specific handwriting processing. Technically, the pattern is a composition of combinatorial combinations of standard sub-patterns, which are constantly read by a small integrated pen-camera while writing on the paper. While this technique offers many of the desired features that a modern pen-based device should possess, the requirement of non-standard paper is a major drawback. In fact, many people think that this restriction will eventually prevent those devices from becoming mainstream, omnipresent tools. Anyway, the concept presented in this paper (see Section 4 and 5) is an appealing alternative to this technique, offering the same features without the requirement of special paper.

3.2. Sensor clamp

This method is based on an array of sensors attached directly to the paper [2]. The sensors receive signals emitted by the pen, while a processor utilizes those signals to calculate the pen's exact global position. Currently, a combination of infrared and ultrasonic signals are typically used for this purpose. A sensor array attached firmly to a sheet of paper avoids the problem of paper moving under the pen tip. Also, it should be technologically possible to integrate the sensor array directly into the cap of the pen. The cap could then be easily removed from the pen and attached to the paper — a self-explanatory and handy interface. The concept presented in Section 4 and 5 fits in well here.

3.3. Paper detectors

This is a relatively new method whose technical feasibility still needs to be proved, in particular in combination with handwriting recognition [4]. The idea is to record handwriting without any external sensors outside the pen. Of course, this poses a very challenging technological problem. Up to the present day, no hardware device has been able to detect the absolute position of a pen on paper, with the accuracy needed by handwriting recognition, without the help of information received from outside the pen. The technical realization described in [4] utilizes lasers to capture the position of the pen and detect the boundaries of the paper. Its technical details are out of the scope of this paper. Nevertheless, the Callpaper concept presented in the following sections also harmonizes with this sophisticated approach.

4. Client-server architecture

The Callpaper concept is based on a client-server architecture that manages requests for context information essential for the proper recognition and processing of handwritten documents, independent of the writer's location. Roughly speaking, the Callpaper framework provides a means to combine syntax and semantics over a long distance. A writer can enhance handwritten data with semantic information by establishing a connection over the Internet or local area network to the Callpaper server. This basic principle allows automatic processing of handwritten documents without modifying the standard pen-and-paper interface. Figure 1 shows the architecture of the Callpaper concept, which comprises the following physical and logical components: Callpaper, Callpaper server, Callpaper client, and Callpaper model.

4.1. Callpaper

Callpaper is the name of the paper that, in combination with a pen, can act as a computer terminal allowing us to interface computers with just pen and paper. Callpaper is by no means special paper. It is best described by the following handy formula:

$$Callpaper = Paper + ID + IT$$

In words, this formula states that every standard paper is a Callpaper as long as it has a unique **ID** assigned to it and a corresponding model file accessible over a network (**I**nformation **T**echnology). In particular, this definition implies that Callpaper is a physical object possessing all the advantages of paper.

4.2. Callpaper server

The Callpaper server is basically a large database containing model files corresponding to Callpapers. Writers can access those model files stored on the server under their unique IDs. Upon receiving a request for a specific model, the server dispatches the model file to the client site, where it provides context information essential for handwriting recognition. Depending on the server's implementation and the type of models, handwriting recognition may also be performed on the server side. In that case, the server does not only receive requests for model files but also handwritten data on which it performs complex handwriting recognition tasks according to the model file. For the writer, a request for a paper model is straightforward. He just needs to call a Callpaper's unique ID to access the corresponding model file, which is then received by the Callpaper client over the Internet or LAN.

4.3. Callpaper client

The Callpaper client is the software that manages the underlying communication with the Callpaper server on the client site. Accordingly, one important task is to download and interpret model files. Furthermore, the client recognizes handwriting on a document according to the specifications given in the document's model file. It also processes the final recognition result and/or transmits it to the final destination, as specified in the model file.

The Callpaper client is implemented either directly in the pen, or in a mobile phone connected to the pen via a wireless connection, or in a desktop computer. However, given the current state-of-the-art, the phone/pen-combination is much smaller and cheaper to realize than a single pen device featuring the same functionality. Depending on the hardware realization, the client may offer a rudimentary user interface and support simple feedback with an LED or a speaker.

4.4. Callpaper model

Callpaper models describe the meaning and structure of a document and provide instructions for recognizing the handwritten information thereon. The data contained in the model file is especially needed for mapping captured points generated by a pen device from the physical space into the logical space of a document. It is this mapping that allows us to use standard paper for interfacing electronic devices. In addition to this, a model file contains context information about how a document's recognized handwriting should be processed. Some informations a model file can provide are, for example:

- **Paper type:** form, drawing paper, blank paper, ...
- **Paper access:** paper validity, expirations, privacy, ...
- **Document fields:** names, numbers, signatures, ...
- **Consistency checks:** dates, sums, order numbers, ...

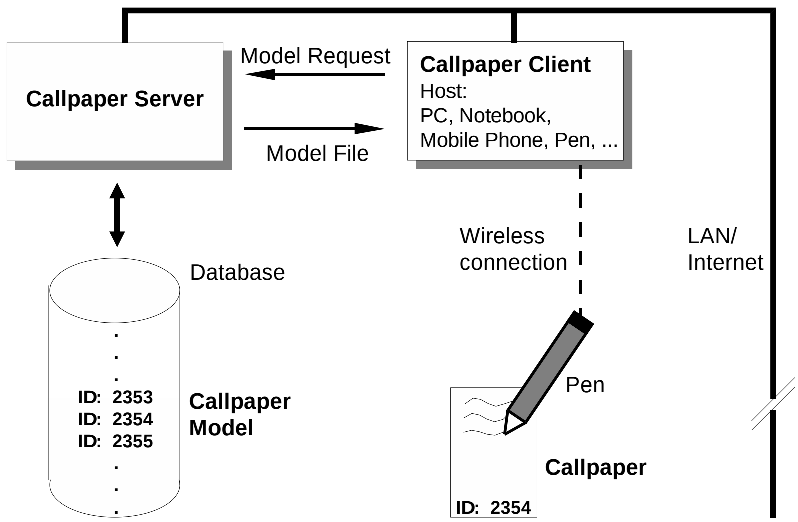

Figure 1. Callpaper architecture.

- **Recipient address:** email address, fax number, . . .
- **Transmission type:** email, fax, ftp, . . .

5. Application flow

The Callpaper concept offers a strikingly simple way for a writer to enter handwritten, paper-based information into an electronic device, and thus to bridge the gap between paper world and electronic world. Let us assume that the writer uses a pen/phone hardware combination. The pen captures the handwriting utilizing one of the above mentioned techniques and is attached to the phone via a wireless connection, such as Bluetooth or infrared. Furthermore, let us assume that the client, i.e. the phone, is capable of performing all necessary handwriting recognition tasks. Then, for the writer, accessing Callpaper functionality is hardly more than a simple phone call. For instance, sending a handwritten Callpaper via email to a recipient consists of three simple steps:

1. **Calling the server:** First, the writer dials the paper ID printed on the paper to call the Callpaper server. The server searches the corresponding model file in its database and returns it promptly to the writer.
2. **Writing the paper:** In the second step, the writer proceeds as normal: He writes the document having the look and feel of normal paper, while the pen is simultaneously capturing his handwriting.
3. **Sending the data:** Finally, the handwriting is recognized and the recognition result send to the recipient according to the specifications given in the model file. The writer can trigger the recognition and transmission of handwriting by either writing a special gesture on a position determined in the model file, such as a signature, or just by pressing a button of the mobile phone.

6. Multi-dimensional Callpaper space

A nice feature of the Callpaper concept is that paper is no longer confined to two dimensions. In combination with modern information technology, paper becomes a multi-dimensional space. For instance, a combination of pen and state-of-the-art cellular phone with integrated GPS receiver can support the following features:

	Writer position (2-dimensions)
+	Pen Position (2-dimensions)
+	Callpaper ID (1-dimension)
+	Writer ID (1-dimension)
+	Pen ID (1-dimension)
+	Time (1-dimension)
$\sum$	8-dimensional Callpaper space

These features expand the two-dimensional hyperspace of the writing plane into an eight-dimensional space. This expanded space extends the possibilities of document analysis and offers a wide range of new applications. It leads to a better customization of document processing, considering in more detail the specific requirements of customers in different locations and environments.

```
<PaperModel Type="MultipleChoice" Multiple="No" Transaction="Email"
Recipient="webmaster@callpaper.com" Width="105" Height="88"
P0X="0" P0Y="0" P1X="105" P1Y="0" P2X="105" P2Y="88" P3X="0" P3Y="88">
<TickField FieldName="Option A" ULX="87" ULY="1" LRX="101" LRY="18" ReturnString="- A -">
<TickField FieldName="Option B" ULX="87" ULY="18" LRX="101" LRY="36" ReturnString="- B -">
<TickField FieldName="Option C" ULX="87" ULY="36" LRX="101" LRY="53" ReturnString="- C -">
<ActionField FieldName="Send" ULX="0" ULY="0" LRX="15" LRY="88" Action="email">
</PaperModel>
```

Figure 2. Simple Callpaper model file.

7. Implementation

Our strategy is to start with simple documents requiring manageable handwriting recognition or none at all and then extend to more complex papers. Oddly enough, blank paper has the highest complexity in this respect since it provides the least context information for handwriting recognition. Accordingly, the first implemented Callpapers have been simple multiple choice tests. Figure 2 shows a simple model file corresponding to a multiple choice document whose ticked answers are send by email to a collector. The syntax of this model file is akin to standard HTML or SGML syntax. The first line specifies the paper type and other essential attributes, such as the address of the recipient and the type of transaction. The attribute “Multiple” specifies whether multiple ticks are allowed for this multiple choice test or not. This allows consistency checks by the Callpaper client later on. Moreover, the first line contains the coordinates of four calibration points necessary for calibrating the pen’s coordinate space so that it matches the logical paper space. A calibration mode, which is automatically started directly after downloading the model file, requires the writer to tap four points with the pen tip, one in each corner of the paper. The captured points are then matched with the model points, utilizing a linear normalization of translation, rotation, and scaling. Mathematically, only three points are needed for this purpose and the fourth point serves just as a control point reducing the effect of imprecise user input. The main body of the model file in Figure 2 is a list of tick field specifications. Each specification describes the upper left and lower right point of a tick field’s bounding box in coordinates of the logical paper space. A return string at the end of each specification contains the text that is send to the recipient via e-mail in case the field is ticked by the writer. The last line of the model file specifies the so-called action field. By simply ticking this action field, the writer can trigger the processing of handwriting and the transmission of its recognition result to the recipient.

In the present client-server setup, the client performs all processing and handwriting recognition, while the server acts mainly as a file server for model files. At present, the client runs on a desktop computer. However, it offers merely basic interface functions, such as simple acoustic feedback, to emulate its application in a small hand-held device like a pen or a phone.

8. Outlook

The complexity of Callpapers ranges from mere handwritten ink transmission to blank paper with sophisticated handwriting recognition and document analysis. Accordingly, the next steps are to increase the complexity, and support the full spectrum of Callpapers. More complex documents allow highly sophisticated transactions triggered by just writing on paper. Integrating more powerful handwriting recognition will increase the level of automatization possible, similar to the development in postal automation, which started with printed character recognition and later extended to cursive handwriting recognition [5, 3].

The calibration method in our current implementation can probably be made more user-friendly. For instance, instead of tapping calibration points, the writer could copy a single calibration mark printed on the paper. Another very appealing solution of the calibration problem is a camera integrated in the sensor clamp attached to the paper, though this requires additional hardware. The camera automatically matches a calibration mark with the specification in the model file whenever the writer attaches the clamp to a designated point on the paper.

A special focus of the future work lies on the generation of model files. Software tools assisting in generating model files for existing documents will allow us to easily enhance today’s documents with Callpaper functionality.

References

[1] http://www.anoto.com.

[2] http://www.siibusinessproducts.com.

[3] S. Jaeger, S. Manke, J. Reichert, and A. Waibel. Online Handwriting Recognition: The Npen++ Recognizer. *International Journal on Document Analysis and Recognition*, 3(3):169–180, 2001.

[4] http://www.otmtech.com.

[5] R. Plamondon and S. Srihari. On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 22(1):63–84, 2000.