INK COLOR RENDERING FOR ELECTRONIC ANNOTATIONS

(57) Abstract: A system and method for rendering ink annotations (1302, 1303) for a displayed image (1301) is disclosed. The invention renders annotations (1302, 1303) to mimic how physical ink would appear if similarly applied to physical paper. In one embodiment, the invention uses subtractive rendering to produce combinations of colors. Here, the system subtracts a complement of a color from an existing color to produce the rendered color. The rendered color is the color that is common to the annotation color and the existing color. This operation may be simplified as a binary ADD of the numeric representation of the two constituent colors. Using this invention, the resulting color shown at a given pixel P shows the history of all color layers at the pixel P, rather than merely the last applied color.
Ink Color Rendering for Electronic Annotations

1. Related Applications

This application is related to the following applications filed herewith:

U.S. Serial No. 09/455,806, filed December 7, 1999, entitled “Method and Apparatus For Capturing And Rendering Annotations For Non-Modifiable Electronic Content.”

2. Background

Technical Field

The disclosure generally relates to the electronic display of documents. More particularly, the disclosure relates to the rendering of color of annotations in electronically displayed documents.

Related Art

Many factors today drive the development of computers and computer software. One of these factors is the desire to provide accessibility to information virtually anytime and anywhere. The proliferation of notebook computers, personal digital assistants (PDAs), and other personal electronic devices reflect the fact that users want to be able to access information wherever they may be, whenever they want. In order to facilitate greater levels of information accessibility, the presentation of information must be made as familiar and comfortable as possible.

In this vein, one way to foster success of electronic presentations of information will be to allow users to handle information in a familiar manner. Stated another way, the use and manipulation of electronically-presented information may mimic those paradigms that users are most familiar with, e.g., printed documents, as an initial invitation to their use. As a result, greater familiarity between users and their “machines” will be engendered, thereby
fostering greater accessibility, even if the machines have greater capabilities and provide
more content to the user beyond the user’s expectations. Once users feel comfortable with
new electronic presentations, they will be more likely to take advantage of an entire spectrum
of available functionality.

One manner of encouraging familiarity is to present information in an electronic book
format in which a computer displays information in a manner that closely resembles printed
books. In order to more completely mimic a printed book, users will need to have the ability
to make textual notes to themselves, akin to writing in the margins of paper books. Users will
also want to highlight selected portions, as these are active-reading activities of which a user
would expect to see in an electronic book. Users will want to add drawings, arrows,
underlining, strike-throughs, and the like, also akin to writing in paper books. Finally, users
will want to add bookmarks.

The above-identified so-called “active-reading” activities are available. However, all
of these active-reading activities require modification of the underlying document. For
example, as is known in the art, if one adds a comment or annotation in an electronic editor,
the comment or annotation is inserted into the document. This insertion corrupts the
underlying document from its pre-insertion, pristine state. While this may not be an issue in
an editable document, the modification of a copyrighted document may run afoul of various
copyright provisions. The violations may be compounded with the forwarding of the
document to another in its modified state. Further, irrespective of any copyright
transgressions, publishing houses responsible for the distribution of the underlying text may
not be pleased with any ability to modify their distributed and copyrighted works.

Thus, the users’ desire to actively read and annotate works clashes with the goals of
publishing houses to keep copyrighted works in their unmodified state. Without solution of
this dilemma, the growth of the electronic publishing industry may be hampered, on one
hand, by readers who refuse to purchase electronic books because of the inability to annotate read-only documents and, on the other hand, by the publishing industry that refuses to publish titles that allow for annotations that destroy the pristine compilation of the electronic works.

Further, techniques of rendering annotations may not preserve the information needed by a user. Applications that allow users to make ink marks on top of a document’s displayed text typically treat the added ink as opaque - meaning that the added mark obscures and hides any information that might have previously existed "under" the added mark. Thus if the user adds ink over some displayed electronic text, the displayed electronic text is obscured by the display system’s rendition of the added ink. If the user adds several consecutive ink marks, even using several different colors in the process, the last added mark in the last used color is the one displayed foremost, thus causing a visible-loss of previously occurring information at that point, including previously added ink marks as well as the underlying electronic text.

This is a sub-optimal user experience, especially compared to similar ink marks that a user can make on physical paper. In the paper case, the user just picks up one or more (colored) pens and marks up the paper as desired. In the most typical cases, the effect of the colored pen on the text follows simple rules of nature - light colors do not obstruct darker colors (a yellow pen doesn’t blot out black text on a white background) and superimposed different colored ink marks can still retain their original significance.

In the case of any computer application with GUI that displays non-modifiable text, it can be assumed that the displayed text is of primary significance to the user. If the process of adding ink-marks to the primary text negatively impacts the presentation of the primary text, then the user is unlikely to frequently draw on or annotate the displayed electronic information and/or is unlikely to have a satisfactory ink-annotation experience. What is needed is a mechanism to enhance the ink-annotation features for electronically displayed information to match or surpass a similar experience with paper books.
3. Summary

The present invention provides a technique for annotating an electronic document without corruption of the document itself. In the context of the present invention, a "document" encompasses all forms of electronically displayable information including but not limited to books, manuals, reference materials, picture books, etc. To create an annotation, a user selects an object in the document to locate where the annotation is to be placed. The computer system determines which object has been selected and determines a file position associated with the selected object. The user adds the annotation and, eventually, returns to reading the document. The annotations may be filtered, navigated, sorted, and indexed per user input. Annotations may include text annotations, drawings, highlights, bookmarks, and the like as is related to the general field of active reading.

In the context of the present invention, a displayed "object" may include text, graphics, equations, and other related elements as contained in the displayed document. Annotations may include highlighting, adding textual notes, adding drawings (as one would expect to do with a pencil or pen to a paper book), and adding bookmarks.

To associate an annotation with a selected object, the annotations are linked to a file position in the non-modifiable document. The invention calculates the file position of, for example, the first character of the word (or other displayed element) and stores the file position with the annotation in a separate, linked local file. Alternatively, the non-modifiable document may represent a non-modifiable portion of a file, with the annotations being added to a write-enabled portion of the file.

The determined file position may be used for direct random access into the non-modifiable document despite the document being compressed or decompressed. In one embodiment, the file position is specified in a UTF-8 (a known textual storage format) document derived from an original Unicode (another known textual storage format).
However, in order to conserve space, the non-modifiable document may be compressed using a general-purpose binary compression algorithm, decompressed, and translated to Unicode for viewing. Accordingly, the file position as stored for an annotation is consistent through various storage schemes and compression techniques.

This invention further relates to making ink annotations for a displayed image mimic the appearance of physical ink annotations made on paper. Electronic annotations as are known in the art commonly work off the last-in-time principle of showing only the last applied annotation, with the annotation being rendered as opaque, obscuring the underlying text or other underlying annotations. This invention analyzes the ink for each annotated pixel and renders the color and brightness of each pixel to based on the original pixel color and the added annotation color so as to appear as physical ink would typically appear if similarly applied to physical paper.

The invention uses subtractive rendering to produce combinations of colors. In short, the invention subtracts a complement of a color from an existing color to produce the rendered color. The rendered color is the color that is common to the annotation color and the existing color. This operation may be simplified as a binary ADD of the numeric representation of the two constituent colors. Using this invention, the resulting color shown at a given pixel P shows the history of all color layers at the pixel P, rather than merely the last applied color.

The invention allows users to add "ink"-marks (or "ink-annotations") to a non-modifiable displayed document. Various embodiments provide one or combinations of the following advantages:

1. The ink "behavior" mimics the appearance of physical ink annotations on a paper book page;
2. The ink annotation does not obliterate the original displayed electronic text; and

3. The user is provided with more enhanced ink-annotation capabilities than may be otherwise available.

These and other novel advantages, details, embodiments, features and objects of the present invention will be apparent to those skilled in the art from following the detailed description of the invention, the attached claims and accompanying drawings, listed herein, which are useful in explaining the invention.

4. Brief Description of Drawings

Figure 1 shows a general-purpose computer supporting the display and annotation of an electronic document in accordance with embodiments of the present invention.

Figure 2 shows a displayed document on a computer screen in accordance with embodiments of the present invention.

Figures 3A and 3B show different document formats available for storing a document in accordance with embodiments of the present invention.

Figure 4 shows different bytes for storing characters in UTF8 and Unicode in accordance with embodiments of the present invention.

Figure 5 shows a process for determining the file position of an object in accordance with embodiments of the present invention.

Figure 6 shows another process for determining the file position of an object in accordance with embodiments of the present invention.

Figure 7 shows a process for displaying annotations in accordance with embodiments of the present invention.

Figures 8A and 8B show various storage techniques for storing annotations in accordance with embodiments of the present invention.
Figure 9 shows a screen for manipulating annotations in accordance with embodiments of the present invention.

Figure 10 shows before and after images of annotations on a page in accordance with embodiments of the present invention.

Figure 11 shows a close-up view of a portion of Figure 10 in accordance with embodiments of the present invention.

Figure 12 shows a method for determining a resultant value from subtractive rendering of two colors in accordance with embodiments of the present invention.

Figure 13 shows multiple annotations overlying displayed content in accordance with embodiments of the present invention.

5. Detailed Description

The present invention relates to a system and method for rendering capturing and associating annotations associated with a non-modifiable document.

Although not required, the invention will be described in the general context of computer-executable instructions, such as program modules. Generally, program modules include routines, programs, objects, scripts, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Moreover, those skilled in the art will appreciate that the invention may be practiced with any number of computer system configurations including, but not limited to, distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in both local and remote memory storage devices. The present invention may also be practiced in personal computers (PCs), hand-held devices, multiprocessor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, and the like.
Figure 1 is a schematic diagram of a computing environment in which the present invention may be implemented. The present invention may be implemented within a general purpose computing device in the form of a conventional personal computer 200, including a processing unit 210, a system memory 220, and a system bus 230 that couples various system components including the system memory to the processing unit 210. The system bus 230 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. The system memory includes read only memory (ROM) 240 and random access memory (RAM) 250.

A basic input/output system 260 (BIOS), containing the basic routines that help to transfer information between elements within the personal computer 200, such as during start-up, is stored in ROM 240. The personal computer 200 further includes a hard disk drive 270 for reading from and writing to a hard disk, not shown, a magnetic disk drive 280 for reading from or writing to a removable magnetic disk 290, and an optical disk drive 291 for reading from or writing to a removable optical disk 292 such as a CD ROM or other optical media. The hard disk drive 270, magnetic disk drive 280, and optical disk drive 291 are connected to the system bus 230 by a hard disk drive interface 292, a magnetic disk drive interface 293, and an optical disk drive interface 294, respectively. The drives and their associated computer-readable media provide nonvolatile storage of computer readable instructions, data structures, program modules and other data for the personal computer 200.

Although the exemplary environment described herein employs a hard disk, a removable magnetic disk 290 and a removable optical disk 292, it should be appreciated by those skilled in the art that other types of computer readable media which can store data that is accessible by a computer, such as magnetic cassettes, flash memory cards, digital video disks, Bernoulli cartridges, random access memories (RAMs), read only memories (ROMs), and the like, may also be used in the exemplary operating environment.
A number of program modules may be stored on the hard disk, magnetic disk 290, optical disk 292, ROM 240 or RAM 250, including an operating system 295, one or more application programs 296, other program modules 297, and program data 298. A user may enter commands and information into the personal computer 200 through input devices such as a keyboard 201 and pointing device 202. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 210 through a serial port interface 206 that is coupled to the system bus, but may be connected by other interfaces, such as a parallel port, game port or a universal serial bus (USB). A monitor 207 or other type of display device is also connected to the system bus 230 via an interface, such as a video adapter 208. In addition to the monitor, personal computers typically include other peripheral output devices (not shown), such as speakers and printers.

The personal computer 200 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 209. The remote computer 209 may be another personal computer, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the personal computer 200, although only a memory storage device 211 has been illustrated in Figure 1. The logical connections depicted in Figure 1 include a local area network (LAN) 212 and a wide area network (WAN) 213. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet.

When used in a LAN networking environment, the personal computer 200 is connected to the local network 212 through a network interface or adapter 214. When used in a WAN networking environment, the personal computer 200 typically includes a modem 215 or other means for establishing a communications over the wide area network 213, such as
the Internet. The modem 215, which may be internal or external, is connected to the system bus 230 via the serial port interface 206. In a networked environment, program modules depicted relative to the personal computer 200, or portions thereof, may be stored in the remote memory storage device. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

In addition to the system described in relation to Figure 1, the invention may be practiced on a handheld computer. Further, purpose-built devices may support the invention as well. In short, handheld computers and purpose-built devices are similar in structure to the system of Figure 1 but may be limited to a display (which may be touch-sensitive to a human finger or stylus), memory (including RAM and ROM), and a synchronization/modem port for connecting the handheld computer and purpose-built devices to another computer or a network (including the Internet) to download and/or upload documents or download and/or upload annotations. The description of handheld computers and purpose-built devices is known in the art and is omitted for simplicity. The invention may be practiced using C. Also, it is appreciated that other languages may be used including C++, assembly language, and the like.

Figure 2 shows a displayed document on a computer screen in accordance with embodiments of the present invention. As preferred, the document is displayed in a form that closely resembles the appearance of a paper equivalent of the electronic book and, in this case, a paper novel. The document reader window 101 may comprise a variety of portions including a title bar 101 listing the title of the document and a body 102. In the body 102 of the display window, various portions of a document may be displayed. Figure 2 shows an example where a title 104, a chapter number 105, a chapter title 106, and the text of the chapter 107 are displayed. Similar to an actual book, margins 108, 109, 110, and 111 appear
around the displayed text. As referred to herein, the displayed elements may be independently referenced. Here for example, object 103 “we” has a drawing annotation placing a box around it as placed there by the user.

Various schemes exist with which to store electronically displayable information as shown in Figures 3A and 3B. With respect to the storage of text, the industry standard is Unicode UCS-2. Unicode UCS-2 encodes text using two bytes per character. The letters from the standard English alphabet to complex symbols including foreign letters and symbols are all encoded using two bytes. Figure 3A shows a pictorial representation of a four-letter word as stored in four pairs of bytes. Another storage scheme includes UTF-8 in which standard letters (for example, US-ASCII characters) are encoded using only a single byte. Foreign characters and symbols from the Unicode UCS-2 set are encoded with two or three bytes. Part of the first byte is used to indicate how many total bytes define the complete character as shown in Figure 3B. The remaining bytes are restricted to numeric values that cannot be confused with those used to define a single-byte character. For large texts using standard letters, a UTF8-encoded file may have a size half of that as Unicode. However, in the situation in which a number of foreign characters or symbols, the size of the stored file may actually be larger than that of Unicode due to the greater number of three byte representations of a letter or symbol. Other variable byte-length character encodings have been used in industry, for example, the Shift-JIS standard encodes characters (drawn from a smaller set than Unicode draws from) in one or two bytes. Unlike in UTF-8, the second byte of a two-byte character may contain a value that may also be used by itself to represent a single-byte character.

Figure 4 shows different bytes for storing characters in UTF8 and Unicode in accordance with embodiments of the present invention. An example of the two schemes discussed with respect to Figures 3A and 3B is shown in Figure 4. The word “banana” takes
twelve bytes to represent it in Unicode while only using six bytes in UTF8. The word “façade” requires twelve bytes in Unicode and seven bytes in UTF8. Other storage schemes are known in the art but not shown here for simplicity. The difference between UTF8 and Unicode is provided by way of example only and not intended to limit the invention to the use of storage scheme over the other.

The difference in the storage modes becomes relevant in the technique used to fix the file position for an annotation. If the file position is determined with one storage scheme, porting the file position to another storage scheme may not result in the same desired file position for an annotation. Thus, all annotations may be fixed to a file position based on the use of a single scheme. Preferably, the scheme used to hold the document while the document is being displayed is the scheme that is used to determine the file position. So, irrespective of whether the document is closed and compressed to another scheme, when reopened in the display scheme, the file position for the annotation remains the same as when created. Unicode may be the scheme used to display the document. Alternatively, UTF8 may be used as well as any other textual encoding or compression scheme to access the document for display.

Figure 5 shows a process for determining the file position of an object in accordance with embodiments of the present invention. In step 501, a user selects an object on the screen. The user may select the object via a cursor controlled through a mouse, touch-pad, trackball, or like pointing device. Alternatively, the user may use a stylus or finger if the surface of the display can accommodate such input.

In step 502, the system determines which object was selected by the user. This step relates to the conversion of the physical coordinates from the display device to coordinates inside the reader window. From this conversion, the object selected by the user is known.
Step 502A is optional. It relates to the user selection of an action post selection of the object. If the user is supplied with a menu after selection of the object and the function of adding an annotation is provided on the menu, step 502A relates to the selection of the adding the annotation function. An example of adding an annotation is described in detail in U.S. Serial No. BW 03797.84618, filed December 7, 1999, entitled “Method and Apparatus for Capturing and Rendering Text Annotations For Non-Modifiable Electronic Content” whose contents are incorporated by reference for any essential subject matter.

Step 503 relates to the determination of the file position of the selected object. The file position may include the first byte of the selected object. Alternatively, the file position may be the first byte of the last character (or even the character following the last character) of the selected object. Selecting the first byte of the first character to determine the file position provides the advantage of displaying any annotation on the page of the beginning of the object, rather than on the next page if the object spans a page. Anyone of skill in the art will appreciate that any byte of the selected object (or surrounding the selected object) may be selected to provide the file position of the object. Alternatively, one may select a line in which the object resides or the paragraph or the portion of the page (e.g., the top, middle or bottom of the page).

The file position may be determined by counting the number of bytes from some known file position to the location of, for example, the first character of the selected object.

The known file position may be the beginning of the file, or may be, for example, a previously noted file position for the beginning of the current paragraph. The counting step may be performed before or after generation of the annotation. Alternatively, the counting step may be performed in the background while the annotation is being created by the user. Note that annotation file positions may always stored as UTF-8 offsets within the text, as it stood before binary compression. However, the algorithm used to display the text works with
Unicode characters. Therefore, in this example, it is necessary to work back from the selected object to a character with a known UTF-8 file position.

Because the binary file-format of the original publication (electronic book, document, etc.) intermixes markup (tags) with text, it is necessary to discount the bytes taken by such tags when calculating the file-position for the selected object (to which the annotation will be anchored). However, of the said tags, many if not most do not take up a character-position on the display surface. Therefore, it is necessary to keep track of the starting file position of every run of text on the display, which corresponds to an unbroken run of text in the file. An “unbroken” run of text refers to text in the file that is not broken by a start- or an end- tag.

Therefore, the steps involved in accurately determining the file position for anchoring the annotation to the selected object are:

1) Look up in our data structures what display character position is the start of an “unbroken” run described in the preceding paragraphs.

2) Fetch from the same data structure the file-position associated with the starting display-character position.

3) Determine the string which runs from the run-start position to the selection-start position. This string contains some number of Unicode characters.

4) Determine how many UTF-8 bytes would be required to hold a UTF-8-encoded version of the string from step 3).

5) Add the UTF-8 bytecount from step 4 to the file-position from step 2.

Step 504 relates to creating a file to persist the annotation. While shown after step 503, it will be appreciated that it may occur prior to or during the determination of the file position of the object. In step 505, the file position is placed in the header of the file (or portion of the file) storing the created annotation. Alternatively, the file position may be appended to the file being viewed.
Figure 6 shows another process for determining the file position of an object in accordance with embodiments of the present invention. As shown in step 601, a user navigates to a page. Once on the page, the system determines the file position of the first byte of the first object on the page as shown in step 602. The file position may be determined every time a new page is displayed. Alternatively, the system may pause (for example, two seconds) before starting to determine the file position for the first byte in order to allow the user to navigate to a new page before starting the file position determination. This delay provides the advantage of minimizing system workload when a user is quickly flipping pages. Once the user settles down with a given page, the system may then determine the file position of the first byte.

In step 603, the file position of the page is temporarily stored in memory.

In step 604, the system waits for either selection of an object or navigation to another page. More options are contemplated that do not need the file position for execution (for example, looking up a term in a reference document as disclosed in U.S. Serial No. 03797.84619 filed December 7, 1999, entitled “Method and Apparatus for Installing and Using Reference Materials In Conjunction With Reading Electronic Content”, whose contents are incorporated herein by reference in its entirety for any enabling disclosure).

In step 605, once an object is selected, the relative position of the selected object is determined with reference to the first byte of the first object on the displayed page.

In step 606, the file position of the first byte of the first object on the page as determined in step 602 is retrieved from memory (as stored in step 603) and added to the relative position of the first byte of the selected object as determined in step 605 to determine the file position of the selected object.

In step 607, the file position of the selected object is stored along with the created annotation. These steps relating to the determination of the file position may occur before or...
after the annotation for the object. Alternatively, the file position may be preformed in the background while the annotation is being created. Those skilled in the art will appreciate that any number of techniques may be used to determine object position and still be considered to be within the scope of the present invention.

Figure 7 relates to a process for displaying the created annotation when navigating to the page. In step 701, a user navigates to a page.

In step 702, the system determines the file position of the first object on the page.

In step 703, the system determines the file position of the last object on the page.

In step 704, the annotations stored for the document are searched to determine if any have file positions located between the file position determined in step 702 and the file position determined in step 703.

In step 705, if no annotations with a file position are located for display on the displayed page, the system waits for user input (including, for example, navigation to a new page or selection of an object for annotation, or any other action described herein).

In step 706, an annotation has been found that relates to an object on the page. The location of the object on the page is determined and the annotation is displayed for the object. The system for determining the location of the object may include subtracting the file position of the first object on the page from the file position of the annotated object. This difference is then used to determine how many bytes from the first character of the page is the annotated object. At this point, further annotations may be made, by returning from step 706 to step 705.

Alternatively, the system may count again from the beginning of the document to determine which object has been annotated. It will be appreciated by those skilled in the art that that numerous methods exist for displaying the annotation for the annotated object. The above examples are not intended to be limiting.
In the context of displaying the annotations that are determined to exist in a given "page" of the content (the unit of text being viewed by the user at any given time), the computer system will first validate a global state, which determines whether annotations should be rendered at all. For example, the user is provided with the ability to globally specify whether to show or hide drawing annotations (as well as text notes, bookmarks, highlights, etc.). Prior to displaying a particular annotation of an object, the computer system will check this global setting to determine whether or not to render the specific annotation. If the user has chosen to hide annotations of that particular type, the annotation will not be rendered.

Figures 8A and 8B show various storage techniques for storing annotations in accordance with embodiments of the present invention. Figure 8A shows a document 801 that has modifiable (803-806) and non-modifiable (802) portions. Files of this type include Infotext file formats as are known in the art. Annotations 806 may be stored in combination with the non-modifiable content 802. An annotation 806 may be stored in a file with header 803 and body 806. The header 803 includes, for example, the file position 804 of the object with which the annotation 806 is associated. It may also include an indication of the type of annotation 806 in file portion 805. As discussed above, the annotation 806 may include a highlight, a bookmark, a drawing to be overlaid over the object, or a text annotation.

Figure 8B shows the non-modifiable content 809 as a separate file apart from the annotation file. The annotation file 807 of Figure 8B has similar constituent elements to that of annotation 807 of Figure 8A. Annotation file 807 may include a file portion 808 that indicates to which non-modifiable document (here, 809) it is linked. Using the approach set forth in Figure 8B, one file may store all annotations for a user with the non-modifiable content portions 809 being stored separately. This approach has the advantage of being able
to quickly scan all annotations at one time rather than accessing all documents 801 (as including non-modifiable portions 802) to obtain all annotations stored therein.

Figure 9 shows a display window for sorting, modifying, searching, and renaming the annotations stored in a system. The window 900 includes a title identifier 901 to alert the user that he or she is in an annotation pane 900. The window 900 may include two panes 902 and 903 (other panes may be added as needed). Panes 902 and 903 may provide a listing of annotations 904 by document. Alternatively, they may provide a listing of all annotations in a person’s system. When in pane 902 (here, entitled “Notes”), the user may sort the list of annotations by type (highlight, drawing, text, bookmark). Selecting an annotation allows one to navigate to the location in the document containing the annotation. Selecting and holding the annotation allows one to remove, change the appearance of, hide or show that particular annotation, or rename the annotation. The second pane 903 (here, entitled “View”) may allow a user to sort annotations based on their properties. For example, one may sort on the time created, time last accessed, by type, alphabetically, and on book order. Further, individual annotations may be switched on or off using controls on the page. Also, if all annotations have been switched off (or just those of a specific type of annotations have been switched off) and another annotation is created (or another annotation in that class), all annotations of that type may be switched back on. This may be extended to include all annotations being switched on if hidden and a new annotation added.

Figure 10 shows a representative example of an ink-annotated document before application of this invention and the same document with the same annotations after the application of an embodiment of the invention.

Figure 11 shows an enlarged view of Figure 10. In particular, Figure 11 shows frame 1101 of information (here, text) with annotations rendered on top of the information. In frame 1101, the last captured annotation is displayed as opaque above all other information (text
and other annotations). See, for example, the rendering of the text “Clear Type” 1103 is obscured by an opaque annotation overlying the text 1103. Frame 1102 shows the same information (here, the text “Clear Type” 1104) as not being obscured. In Frame 1102, the system has added the two colors of the text to the annotation to obtain the final displayed color for each pixel. In this example, the color of the annotation has been added to the white background to produce the colored annotation. When the annotation overlies other colors, the color of the annotation is added to the other existing colors. In this case, the color of the text “Clear Type” is added to the color of the annotation to produce the final rendered color. As described in greater detail below, the system uses a process of subtractively rendering colors.

The process subtracts the complement of a second color from a first color to produce a third color as a combination of the first and second colors.

As a result of the subtractive rendering process, each displayed color is a combination of all layers of annotation and underlying colors and the color of each pixel represents a history of all colors applied to the pixel. For example, an original display of information may be black text on a white background. A first annotation in yellow is added to the displayed information. The black text remains black and the white background turns yellow to represent the annotation. A second annotation in magenta is added across the first annotation. The resulting image produces black text with a first yellow annotation, a second magenta annotation, and a third red color resulting from the combination of the first and second annotation colors, showing the overlapping area of the first and second annotations.

Subtractive rendering of colors, while not present in electronic media, is present in physical paper. Colors on a physical paper, and in real-life cases, render through a subtractive process (e.g. the appearance of red color on white paper results from the suppression (subtraction) of blue and green from the white on the paper). The solution to the above problem for displaying ink-marks cleanly on top of underlying electronic information
displayed on a computer screen is to similarly render user-added marks in a subtractive manner - in other words, do subtractive rendering of user-added ink marks on the underlying electronic information. Benefits of subtractive rendering include the following:

(a) Subtractive rendering does not compromise the underlying text which is being marked up any more than marking similar text on paper with a similarly colored ink pen; and

(b) Subtractive rendering provides a user-experience that closely matches what the user experiences and expects with physical ink and paper.

The subtractive rendering is achieved by manipulating user-added ink marks on a point by point basis, by combining the color the user wants to add at a point with the color that is already at the given point. The process for subtractively rendering combinations of colors is described below.

In the subtractive rendering process, for any given point P, adding ink of Color X to any existing point of Color Z, actually requires removing the complement of Color X from the existing Color Z. (e.g. to set a point on a white background to red, one needs to subtract cyan --the complement of red -- from white).

Denoting the complement of Color X as Color Y, the following mathematical relationships exist for the RGB based numerical-representation of the colors X, Y, and Z:

\[
\begin{align*}
X \lor Y &= \text{WHITE} \quad \{\text{binary-or of } X, Y\} \\
X &= \sim Y \quad \{X \text{ is the same as the complement of } Y\} \\
Y &= \sim X \quad \{Y \text{ is the same as the complement of } X\}
\end{align*}
\]

Mathematically, subtractive rendering of Color X on existing Color Z

\[
\begin{align*}
= (Z \sim X) \quad \{\text{Removing the complement of Color } X \text{ from Color } Z\} \\
= (Z \sim Y) \quad \{\text{Removing Color } Y \text{ from Color } Z\}
\end{align*}
\]

Mathematically, removing Color Y from Color Z is achieved by doing a bit-wise binary AND of the Color Z with the complement of the Color Y.
Hence,

\[(Z \& \sim X) = (Z \& Y) = (Z \& \sim Y) = (Z \& X)\]

In other words, subtractively combining Color Z and Color X can be achieved by doing a bit-wise binary AND of the numeric representation of the two colors Z and X.

Denoting the value \((Z\&X)\) as a new color Color V, for any given point P having Color Z, the effect of subtractively adding a new Color X to point P is achieved by setting the point P to a new color \(V = (Z\&X)\).

Color V then effectively captures all the rendition history of point P in the same way that the previous color Color Z captured all the rendition history of the point prior to this step.

The subtractive combination of additional points of color to an existing point of color on the display screen effectively mimics the subtractive combination of colors on paper.

For the red, green, and blue components of an RGB signal, the system applies the process to each component separately.

When the context of these color combinations is expanded from a single point to all points in the added ink “stroke” or in displayed electronic text, effective results include the following.

a) Text of a color darker than the ink color is never obscured (since most text is rendered in black, the original document text almost always persists in spite of ink “added on top”).

b) Ink is never rendered opaque but instead ink points are a combination of existing text/ink and the new ink and can convey sufficient information about the “history” of the point.

c) Multiple ink annotations can overlay the text and complement or supplement each other easily. The user can thus use a combination of different colored inks to create more
meaningful annotations and convey more information than
might have been possible with a single opaque color.

d) The overall annotation experience is greatly improved and
matches or surpasses the ink-on-paper-book annotation
experience.

The actual implementation of doing subtractive color combinations at each rendition
point may be accomplished using the existing Windows® operating system API as described
below.

For the following example, the subtractive rendering operation includes doing a
mathematical binary AND between the color of a given pixel (Color A) and the color that one
wants to subtractively render upon the given pixel (Color B). Thus, in one embodiment, the
operation that needs to happen is to change the resulting color of the pixel to (A & B).

The Windows® operating system provides some APIs for manipulating on-screen
colors as part of drawing operations. One of these APIs is the SetROP2 API (described
below) that allows us to set the “mix mode” on the GDI device context upon which the
annotation is being rendered. (The “mix mode” defines how new and existing colors will be
combined or “mixed”). By specifying the “mix mode” as R2_MASKPEN, one can instruct
the Windows® operating system to automatically perform a bitwise AND operation on
existing and added colors at any point. Once the “mix mode” is set on the Windows®
operating system device context upon which the drawing operation is being performed,
subsequent drawing operations on the device context follow the specified mode.

Hence one possible manner in which the software code to implement subtractive
rendering in the Windows® operating system may be implemented as follows:

    // turn on subtractive rendering by specifying R2_MASKPEN
    // so the Windows® operating system will do an AND
// operation between the
// existing and new colors.
int nOldMixMode = SetROP2(hDC, R2_MASKPEN);
// ...(perform the drawing operations)...

// reset the mix mode
SetROP2(hDC, nOldMixMode);
... etc.

The following is the documentation for the SetROP2 API from the April 2000 MSDN®
developer program:

SetROP2

The SetROP2 function sets the current foreground mix mode. GDI uses the
foreground mix mode to combine pens and interiors of filled objects with the colors already
on the screen. The foreground mix mode defines how colors from the brush or pen and the
colors in the existing image are to be combined.

int SetROP2(HDC hdc, int fnDrawMode);

Parameters

hdc

[in] Handle to the device context.

fnDrawMode

[in] Specifies the mix mode. This parameter can be one of the following values.

<table>
<thead>
<tr>
<th>Mix mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R2_BLACK</td>
<td>Pixel is always 0.</td>
</tr>
<tr>
<td>R2_COPYPEN</td>
<td>Pixel is the pen color.</td>
</tr>
<tr>
<td>R2_MASKNOTPEN</td>
<td>Pixel is a combination of the colors common to both the screen and the inverse of the pen.</td>
</tr>
</tbody>
</table>
R2_MASKPEN
Pixel is a combination of the colors common to both the pen and the screen.

R2_MASKPENNOT
Pixel is a combination of the colors common to both the pen and the inverse of the screen.

R2_MERGEONTOPEN
Pixel is a combination of the screen color and the inverse of the pen color.

R2_MERGEOPEN
Pixel is a combination of the pen color and the screen color.

R2_MERGEOPENNOT
Pixel is a combination of the pen color and the inverse of the screen color.

R2_NOP
Pixel remains unchanged.

R2_NOT
Pixel is the inverse of the screen color.

R2_NOTCOPYOPEN
Pixel is the inverse of the pen color.

R2_NOTMASKPEN
Pixel is the inverse of the R2_MASKPEN color.

R2_NOTMERGEOPEN
Pixel is the inverse of the R2_MERGEOPEN color.

R2_NOTXOROPEN
Pixel is the inverse of the R2_XOROPEN color.

R2_WHITE
Pixel is always 1.

R2_XOROPEN
Pixel is a combination of the colors in the pen and in the screen, but not in both.

Return Values
If the function succeeds, the return value specifies the previous mix mode.

If the function fails, the return value is zero.

Windows NT/2000: To get extended error information, call GetLastError.

Mix modes define how GDI combines source and destination colors when drawing with the current pen. The mix modes are binary raster operation codes, representing all possible Boolean functions of two variables, using the binary operations AND, OR, and XOR (exclusive OR), and the unary operation NOT. The mix mode is for raster devices only; it is not available for vector devices.

The following information describes various platforms and declarations:

Windows NT/2000: Requires Windows NT 3.1 or later.

Windows 95/98: Requires Windows 95 or later.

Header: Declared in Wingdi.h; include Windows.h.

Library: Use Gdi32.lib.
Figure 12 shows a subtractive rendering process in accordance with embodiments of the present invention. In step 1201, X is the new annotation color, Z is the original pixel color. In step 1202, the binary representation of Z is obtained. In step 1203, the binary representation of X is obtained. In step 1204, the new color to be rendered for a pixel is determined based on a bit-wise AND of X and Z. This process is performed for each of the components of the RGB representations of X and Z.

As luminance and chrominance values (YUV) may be converted to RGB values, it is considered within the scope of the invention to receive YUV inputs, convert to RGB values, perform subtractive rendering as described herein, then convert the result back to YUV values as needed.

Figure 13 shows overlying layers of annotations. Original text is shown in screen 1301. Annotation 1302 of color A is shown overlying a portion of screen 1301. On top of annotation 1302 is annotation 1303 of color B that overlaps in region R 1304. In the prior art, the final display of all annotations would display region R 1304 as color B. Here, region R is displayed as a combination of colors A and B. Using the subtractive rendering approach of the present invention, region R 1304 is rendered as a bit-wise binary AND of A and B, in other words, R=(A&B).

While the subtractive rendering process described herein relates to ink annotations, it is readily appreciated that the same subtractive rendering process may likewise be applied to other annotation processes as well including highlighting and text annotations. In the example of text annotations, one may enter a text annotation and have the text annotation be displayed in first color over a background or other annotation of another color.

In the foregoing specification, the present invention has been described with reference to specific exemplary embodiments thereof. Although the invention has been described in terms of various embodiments, those skilled in the art will recognize that various
modifications, embodiments or variations of the invention can be practiced within the spirit and scope of the invention as set forth in the appended claims. All are considered within the sphere, spirit, and scope of the invention. The specification and drawings are, therefore, to be regarded in an illustrative rather than restrictive sense. Accordingly, it is not intended that the invention be limited except as may be necessary in view of the appended claims.
Claims

1. A computer-implemented method for annotating a system having a display for
   displaying a page, said method comprising the steps of:
   receiving an annotation;
   determining a color of an area of said page underlying said annotation;
   rendering said annotation as a combination of an annotation color and said color of
   said area.

2. The computer-implemented method according to claim 1, wherein the
   annotation is generated through interaction with a stylus.

3. The computer-implemented method according to claim 1, wherein the
   annotation is generated through interaction with a mouse.

4. The computer-implemented method according to claim 1, wherein the
   annotation is a highlight.

5. The computer-implemented method according to claim 1, wherein the
   annotation is a drawing.

6. The computer-implemented method according to claim 1, wherein the
   annotation is a text annotation.

7. The computer-implemented method according to claim 1, wherein said
   rendering step comprises the step of:
   subtracting a complement of said annotation color from said color of said area.

8. The computer-implemented method according to claim 1, wherein said
   rendering step comprises the step of:
   performing a binary AND operation between said color of said area and said
   annotation color.
9. The computer-implemented method according to claim 1, wherein said annotation is an annotation of an object.

10. The computer-implemented method according to claim 1, wherein said annotation overlies non-modifiable content.

11. A computer-readable medium having a program stored thereon, said program used in conjunction with a system having a display for displaying a page, said program comprising the steps of:

   receiving an annotation;

   determining a color of an area of said page underlying said annotation;

   rendering said annotation as a combination of an annotation color and said color of said area.

12. The computer-readable medium according to claim 11, wherein the annotation is generated through interaction with a stylus.

13. The computer-readable medium according to claim 11, wherein the annotation is generated through interaction with a mouse.

14. The computer-readable medium according to claim 11, wherein the annotation is a highlight.

15. The computer-readable medium according to claim 11, wherein the annotation is a drawing.

16. The computer-readable medium according to claim 11, wherein the annotation is a text annotation.

17. The computer-readable medium according to claim 11, wherein said rendering step comprises the step of:

   subtracting a complement of said annotation color from said color of said area.
18. The computer-readable medium according to claim 11, wherein said rendering step comprises the step of:

performing a binary AND operation between said color of said area and said annotation color.

19. The computer-readable medium according to claim 11, wherein said annotation is an annotation of an object.

20. The computer-readable medium according to claim 11, wherein said annotation overlays non-modifiable content.

21. A display system for displaying an annotation comprising:

a display having pixels;

a driver for driving said display, said driver rendering a pixel of said pixels as a combination of a first color and a second color,

wherein the first color is a background color and said second color is an annotation color.

22. The display system according to claim 21, wherein said driver renders said pixel as subtracting a complement of said annotation color from said first color.

23. The display system according to claim 21, wherein said first color is redefined as a combination of said first color and said second color.

24. The display system according to claim 21, wherein said display system displays non-modifiable content.
Figure 5

User Input To Select Object

Determine Which Object Selected

Determine No. Of Byte Of First Byte Of Object

Open File For Annotation

Place Object File Position In Header Of Annotation

Add Annotation 502A
Figure 7

Navigate to Page

701

Determine File Position of First Object on Page

702

Determine File Position of Last Object on Page

703

Search Annotations for File Position Between First Object's File Position and Last Object's File Position

704

Await User Input

705

No

Yes

Display Annotation With Object On Page

706
With my insistence on eliminating paperwork, I must sound "anti-paper." I am against paper forms, but even I still print out long electronic documents I want to read and annotate. Most people, when they’re trying to organize a long document, like to spread the pages out on a table so that they can see them all at once — hard to do with a PDA. Until we get a breakthrough in flat-screen technology — and furious research is going on at Xerox, the MIT Media Lab, Kent State, Microsoft and other academic and corporate research sites in the United States and Japan — books and magazines still can’t beat for readability and portability.

High-quality displays are a necessity in the information-rich future. Microsoft showed, in late 1998, a technology code-named ClearType that allows color LCD screens to display text dramatically better than before. Combining this with improved hardware will be revolutionary. Some future screens will be flexible so that you can roll or fold the display and take it with you, like a newspaper. Other screens will have the computer circuitry embedded in them, so that an entire PC could be as thin as the display part of a current laptop. One new technology enables a screen to retain its image after
Let $X = \text{Annotation Color}$, $Z = \text{Original Pixel Color}$

Obtain Binary Representation of $Z$

Obtain Binary Representation of $X$

Perform Binary Bit wise AND of $X$ and $Z$
Figure 13
INTERNATIONAL SEARCH REPORT

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7  G06F3/033  G06F3/14

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED
Minimum documentation searched (classification system followed by classification symbols)
IPC 7  G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the International search (name of data base and, where practical, search terms used)
EPO-Internal, WPI Data, PAJ, IBM-TDB, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>US 5 920 694 A (SCHWARTZ KRISTA S ET AL) 6 July 1999 (1999-07-06) column 13, line 11 - column 14, line 25; figures 19-22</td>
<td>1-6, 9-16, 19-21</td>
</tr>
<tr>
<td>X</td>
<td>US 5 986 665 A (SUMMER MATTHEW ET AL) 16 November 1999 (1999-11-16) column 1, line 43 - line 65 column 4, line 25 - line 45</td>
<td>1,2,5, 11,12, 15,21,23</td>
</tr>
</tbody>
</table>

X Patent family members are listed in annex.

* Special categories of cited documents:

*A* document defining the general state of the art which is not considered to be of particular relevance

*E* earlier document but published on or after the international filing date

*L* document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

*O* document referring to an oral disclosure, use, exhibition or other means

*P* document published prior to the international filing date but later than the priority date claimed

**T** later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

**X** document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

**Y** document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is compared with one or more other such documents, such combination being obvious to a person skilled in the art.

**A** document member of the same patent family

Date of the actual completion of the International search 5 October 2001

Date of mailing of the International search report 12/10/2001

Name and mailing address of the ISA
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fac. (+31-70) 340-3016

Authorized officer
Durand, J
<table>
<thead>
<tr>
<th>Patent document cited in search report</th>
<th>Publication date</th>
<th>Patent family member(s)</th>
<th>Publication date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>EP 0645726 A2</td>
<td>29-03-1995</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JP 7182282 A</td>
<td>21-07-1995</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EP 0616448 A2</td>
<td>21-09-1994</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JP 7049834 A</td>
<td>21-02-1995</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 5704042 A</td>
<td>30-12-1997</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 5872923 A</td>
<td>16-02-1999</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 5781727 A</td>
<td>14-07-1998</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 5717856 A</td>
<td>10-02-1998</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 5717857 A</td>
<td>10-02-1998</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 5948022 A</td>
<td>07-09-1999</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 6061717 A</td>
<td>09-05-2000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 5819038 A</td>
<td>06-10-1998</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 5838914 A</td>
<td>17-11-1998</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 5920693 A</td>
<td>06-07-1999</td>
</tr>
<tr>
<td>JP 11327789 A</td>
<td>30-11-1999</td>
<td>NONE</td>
<td></td>
</tr>
</tbody>
</table>