

(51) International Patent Classification:  
*H04N 21/454* (2011.01)(21) International Application Number:  
PCT/CN2016/072664(22) International Filing Date:  
29 January 2016 (29.01.2016)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: HEWLETT PACKARD ENTERPRISE DEVELOPMENT LP [US/US]; 11445 Compaq Center Drive West, Houston, Texas 77070 (US).

(72) Inventors; and

(71) Applicants (for US only): WANG, Jin [CN/CN]; Building 6, No. 690 Bi Bo Road, Zhangjiang Micro-Electric Port, No. 6 Sanshi Street, Wuhua District, Shanghai 201203 (CN). WU, Zhu Jing [CN/CN]; Building 6, No.

690 Bi Bo Road, Zhangjiang Micro-Electric Port, No. 6 Sanshi Street, Wuhua District, Shanghai 201203 (CN). LI, Jia-Wen [CN/CN]; Building 6, No. 690 Bi Bo Road, Zhangjiang Micro-Electric Port, No. 6 Sanshi Street, Wuhua District, Shanghai 201203 (CN). ZHAO, Bonny [CN/CN]; Building 6, No. 690 Bi Bo Road, Zhangjiang Micro-Electric Port, No. 6 Sanshi Street, Wuhua District, Shanghai 201203 (CN). LU, Jun [CN/CN]; Building 6, No. 690 Bi Bo Road, Zhangjiang Micro-Electric Port, No. 6 Sanshi Street, Wuhua District, Shanghai 201203 (CN).

(74) Agent: CHINA PATENT AGENT (H.K.) LTD.; 22/F., Great Eagle Center, 23 Harbour Road, Wanchai, Hong Kong (CN).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT,

[Continued on next page]

(54) Title: ERROR BLOCK DETERMINATION

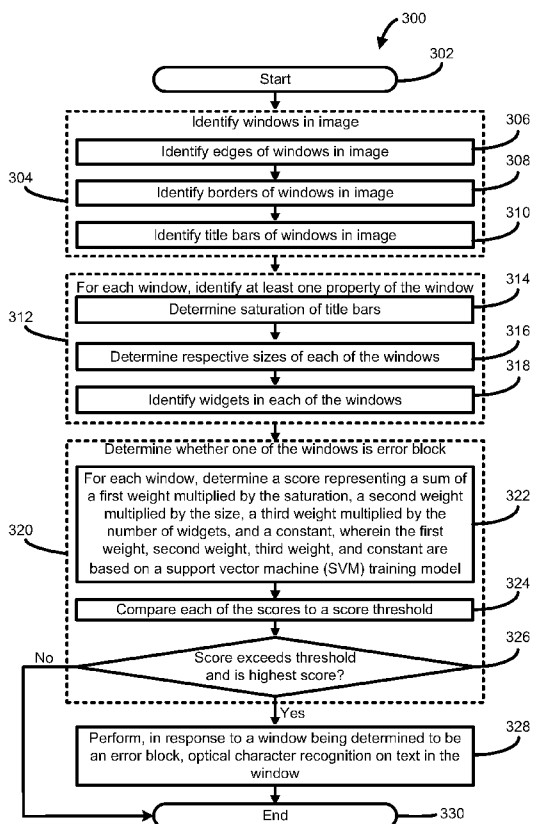


FIG. 6

(57) Abstract: A method, a non-transitory computer-readable storage medium comprising instructions to implement the method, and a corresponding system to determine whether a window is an error block are disclosed. Wherein a window in an image may be identified, then at least one property of the window may be identified. Based on the at least one property, whether the window is an error block may be determined.



HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

**Published:**

**(84) Designated States** (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ,

— *with international search report (Art. 21(3))*

## ERROR BLOCK DETERMINATION

### BACKGROUND

**[0001]** Computing system components, such as applications, servers, databases, etc., may exhibit a variety of errors. In some examples, when an error occurs, an error block window may pop up on a desktop screen and show an error message, which may provide information about the error. This information may be helpful for a user or information technology (IT) professional to understand the error and perform root cause analysis.

### BRIEF DESCRIPTION

**[0002]** Some examples are described with respect to the following figures:

**[0003]** FIG. 1 is a block diagram illustrating a non-transitory computer readable storage medium according to some examples.

**[0004]** FIGS. 2 and 4 are block diagrams illustrating systems according to some examples.

**[0005]** FIG. 5 illustrates an image according to some examples.

**[0006]** FIGS. 3 and 6 are flow diagrams illustrating methods according to some examples.

### DETAILED DESCRIPTION

**[0007]** The following terminology is understood to mean the following when recited by the specification or the claims. The singular forms “a,” “an,” and “the” mean “one or more.” The terms “including” and “having” are intended to have the same inclusive meaning as the term “comprising.”

**[0008]** In some examples, as mentioned above, when an error occurs with a component of a computing system, an error block window may pop up on a desktop screen and show an error message. In some examples, an image of the desktop screen may be obtained, and character recognition (OCR) may be performed to extract text from an error message in the error block in the image. However, such desktop screen images may additionally include various other features besides error blocks, including backgrounds, graphical features, text, windows, toolbars, icons, etc. Therefore, OCR may capture text that is not associated (e.g. outside) the error block. As a result, the recognized text may include irrelevant text such that the recognized text as a whole does not make sense.

**[0009]** Accordingly, the present disclosure provides examples in which error blocks may be identified in images, so that OCR is performed on error blocks, and unintended cases of OCR performed on parts of the images other than error blocks are reduced. Therefore, accuracy in OCR may be greatly increased. The error blocks may be identified based on the identification of properties normally associated with error blocks. For example, windows may be identified by identifying borders and title bars in an image. In some examples, the saturation of the title bar, the number of widgets in the window, and the size of the window. Based on these properties, whether the window is an error block may be determined.

**[0010]** FIG. 1 is a block diagram illustrating a non-transitory computer readable storage medium 10 according to some examples. The non-transitory computer readable storage medium 10 may include instructions 12 executable by a processor to identify a window in an image. The non-transitory computer readable storage medium 10 may include instructions 14 executable by a processor to identify at least one property of the window. The non-transitory computer readable storage medium 10 may include instructions 16 executable by a processor to determine, based on the at least one property, whether the window is an error block. A window is a graphical element that has a border and that is to be displayed on a graphical user interface. A window may or may not include a title bar. An error block is a window that provides to a user a notification regarding an error.

**[0011]** FIG. 2 is a block diagram illustrating a system 20 according to some examples. The system 20 may include a processor 22 and a memory 24. The memory 24 may include instructions 24 executable by the processor to determine that edges in an image form a window comprising a title bar and a rectangular border enclosing the window. The memory 24 may include instructions 26 executable by the processor to identify a plurality of properties of the window. The memory 24 may include instructions 28 executable by the processor to determine whether the window is an error block based on the plurality of properties.

**[0012]** FIG. 3 is a flow diagram illustrating a method 30 according to some examples. The following may be performed by a processor. At 32, a window in an image may be identified. At 34, at least two properties of the window may be determined that are selected from the group consisting of a number of widgets in the window, a saturation of a title bar

of the window, and a size of the window. At 36, a score based on values associated with the at least two properties may be determined. At 38, based on the score, whether the window is an error block may be determined.

**[0013]** FIG. 4 is a block diagram illustrating a system 100 according to some examples. The system 100 includes a data sources 102, communication network 104, and error identification system 108. The error identification system 108 may include a several components including window identifier 112, property identifier 118, error block identifier, and optical character recognition (OCR) performer 132. The window identifier 110 may include edge identifier 112, border identifier 114 and title bar identifier 116. The property identifier 118 may include color determiner 120, size determiner 122, and widget identifier 124. The error block determiner 126 may include score determiner 128 and score comparator 130.

**[0014]** In some examples, these components of the error identification system 108 may be implemented as a computing system including a processor, a memory such as non-transitory computer readable medium coupled to the processor, and instructions such as software and/or firmware stored in the non-transitory computer-readable storage medium. The instructions may be executable by the processor to perform processes defined herein. In some examples, the components may include hardware features to perform processes described herein, such as a logical circuit, application specific integrated circuit, etc. In some examples, multiple components may be implemented using the same computing system features or hardware.

**[0015]** The data sources 102 may, for example, comprise a group of networked computing systems such as servers in a data center, personal computers, and/or may comprise components of the same computing system or multiple computing systems. In other examples, the data sources 102 may comprise any other sources of data. Each of the computing systems may, for example, include components, such as applications, servers, databases, etc., that may exhibit errors.

**[0016]** The error identification system 108 may receive, though a communication interface, image data 106, which may represent an image displayed on a desktop screen while an application is operating, from the at least one of the data sources 102 via the communication network 104. In some examples, multiple sets of different image data 106

may be received such that the error identification system 108 may operate, as described herein, on each set of image data 106. However, for ease of illustration, one set of image data 106 is shown in FIG. 4. In some examples, the image data 106 may be submitted as part of an IT ticket for processing by the error identification system 108.

**[0017]** The network 104 may, for example, be a local area network (LAN), wide area network (WAN), the Internet, or any other network. In some examples, the network 104 may comprise components of an information technology (IT) infrastructure of an organization. The communication interface of the error identification system 108 may include a network interface card (NIC) or another component to receive signals from a communication path, demodulate and/or decode the data from the signals, and/or apply error correction to retrieve data packet(s), header information, etc. in accordance with various communication protocols. The communication interface may also send data to another terminal by generating data packets, header information, and so forth; encoding and/or modulating a signal based on the data to be communicated; and transmitting the signal. Moreover, the communication interface may perform decryption and encryption to facilitate secure communications according to various standards and protocols.

**[0018]** The window identifier 110 may identify windows in the image 106, e.g. by using its edge identifier 112, border identifier 114, and title bar identifier 116.

**[0019]** In describing the operation of edge identifier 112 and border identifier 114, reference will be made to elements described in FIG. 5, which illustrates an image 200 according to some examples. The edge identifier 112 may identify edges (e.g. straight or curved lines) in the image 200. The image 200 contains 81 pixels in a 9 x 9 grid, with the rows and columns respectively labeled A-I and 1-9. In some examples, detection of edges may proceed as follows. The edge identifier 112 may identify edges in the image 200 (e.g. all edges in the image). In some examples, any orientation of the edges may be identified, however in other examples the edge identifier 112 may identify horizontal and vertical edges, and may not identify other types of edges. In FIG. 5, two horizontal edges 202 and 204 (one from C3 to C8 and another from H3 to H8) and two vertical edges 204 and 208 (one from C3 to H3 and another from C8 to H8) are shown, wherein each of the edges 202, 204, 206, and 208 have a thickness of a single pixel. When identifying horizontal and vertical edges, the image 106 may be scanned line by line, both

horizontally and vertically. In some examples, two lines may be scanned at once. By comparing color contrasts between lines, the edge identifier 112 may identify the edges.

**[0020]** For example, when scanning for horizontal edges, the edge identifier 112 may begin by scanning rows A and B together, then rows B and C, etc. The edge identifier may not detect any color contrasts between pixels when scanning rows A and B together, or when scanning rows B and C together. In some examples, when scanning rows C and D together, the edge identifier 112 may detect a difference (e.g. outside a threshold) in color contrast between pixels C4 and D4, but detect a similarity (e.g. within a tolerance threshold) or equivalence in color contrast between pixels C3 and D3. The edge identifier 112 may therefore determine that there is an edge 202 that starts at D4. The edge identifier 112 may through move across the row, and continue to see a color contrast between pixels C5 and D5, between pixels C6 and D6, between pixels C7 and D7, and between pixels C8 and D8. However, the edge identifier 112 may detect a similarity or equivalence in color contrast that pixels C9 and D9, and therefore may determine that the edge 202 ends at D8. The edge identifier 112 may then detect the other horizontal edge 204 spanning H4 to H8 according to a similar method.

**[0021]** In some examples, the edge identifier 112 may then identify any vertical edges, such as the vertical edge 206 spanning D4 to H4 and the vertical edge 208 spanning D8 to H8 according to a similar method. However, in other examples, the edge identifier 112 may directly determine whether the horizontal edges 202 and 204 are connected by vertical edges. For example, the edge identifier 112 may detect a difference in color contrast between pixels C4 and D4 and between D3 and D4, and may therefore determine that there is a vertical edge 206 that starts at D4. The edge identifier 112 may through move across the column, and continue to see a color contrast between pixels E3 and E4, between pixels F3 and F4, between pixels G3 and G4, and between pixels H3 and H8. However, the edge identifier 112 may detect a similarity or equivalence in color contrast that pixels I3 and I4, and therefore may determine that the vertical edge 206 ends at H4. The edge identifier 112 may then detect the other vertical edge 208 spanning D4 to D8 according to a similar method. In some examples, vertical edges may be identified before horizontal edges rather than afterward.

**[0022]** In some examples, the edge identifier 112 may not identify a series of pixels as an edge if it does not extend a threshold length (e.g. threshold number of pixels). For example, the edge identifier 112 may discard an edge if it is less than 30 pixels long.

**[0023]** In some examples, an edge may have a thickness of more than one pixel, e.g. the edge may comprise two adjacent series of pixels. In such cases, the edge identifier 112 may count the two adjacent series of pixels as a single edge.

**[0024]** In some examples, the border identifier 114 may identify whether any combination of edges (or a single edge in the case of a circular border) defines a border enclosing an area in the image 200. Each of the identified borders enclosing an area may correspond to a possible window. In examples where horizontal and vertical edges are identified but other edges are not identified, then any of the identified borders may each be rectangular. In the example of FIG. 4, the border 210 enclosing the area within it and defined by edges 202, 204, 206, and 208 is identified. In some examples, each of the borders may not completely enclose the respective area defining the window, for example three out of four edges of a rectangle may be sufficient to identify a border.

**[0025]** Turning back to FIG. 4, the border identifier 114 may identify five borders 132, 134, 136, 138, and 140 enclosing respective areas. These five borders 132, 134, 136, 138, and 140 may be candidates for windows. In some examples, horizontal edge 164 may be detected by the edge identifier 112, but the border identifier 114 may not identify any border formed by the edge 164.

**[0026]** In some examples, the title bar identifier 116 may identify, for each identified border 132, 134, 136, 138, and 140, whether the area enclosed by the border includes a title bar. This may be determined based on whether an additional horizontal edge spans the horizontal width of the area enclosed by the border. In such case, the area between the identified additional horizontal edge and the top horizontal edge of the border (e.g. the top horizontal edge 202 in FIG. 5) may be a title bar. To identify the additional horizontal edge, similar edge detection techniques may be used as those implemented by the edge detector 112.

**[0027]** Based on whether the area enclosed by a border includes a title bar, the title bar identifier 116 may determine whether the border 140 is part of a window. For example, in FIG. 4, the areas within borders 132 and 134 do not include additional horizontal edges, and

therefore do not include title bars. Therefore, these borders 132 and 134 are not identified as parts of windows. However, the areas within borders 136, 138, and 140 respectively contain additional horizontal edges 142, 144, and 146 that, along with the top horizontal edges of borders 136, 138, and 140, and parts of the vertical edges, form the title bars 143, 145, and 147. Therefore, these borders 136, 138, and 140 and their title bars 143, 145, and 147 are identified as parts of respective windows 152, 154, and 156. Thus, the window identifier 110 may identify windows 152, 154, and 156 in the image 106.

**[0028]** The property identifier 118 may identify properties of the windows 152, 154, and 156, e.g. by using its color determiner 120, size determiner 122, and widget identifier 124.

**[0029]** In some examples, for each of the title bars 143, 145, and 147 of the identified windows 152, 154, and 156, the color determiner 120 may determine a value representing a color property of the title bar. The color property may, for example, be a color property such as saturation, chroma, brightness, colorfulness, hue, contrast, or the like. The following examples will be described relative to saturation, which is a measure that takes into account colorfulness and intensity. For example, the saturation of a color may be determined by a combination of intensity and the distribution of the color across different wavelengths. In some examples, according to the CIELAB color space specified by the International Commission on Illumination, the saturation  $S_{ab}$  may be determined according to the formula:

$$S_{ab} = \frac{\sqrt{a^{*2} + b^{*2}}}{L^*}$$

**[0030]** In this example,  $L^*$  represents the lightness of a color (where 0 represents black and 100 represents diffuse white),  $a^*$  represents the color's position between magenta and green (negative values represent green and positive values represent magenta), and  $b^*$  represents the color's position between yellow and blue (negative values represent blue and positive values represent yellow). In other examples, other measures of saturation may be used.

**[0031]** In some examples, the image may be represented in the CIE RGB (red—green—blue) color space. In such examples, each pixel of the title bars (or the image as a whole) may be converted to the CIELAB color space before calculating the saturation. In

some examples, the determined saturation value of a title bar may be an average of the values for each pixel within the title bar. In other examples, the determined saturation value of the title bar may be the saturation value for the color that appears most frequently in the title bar. In some examples, each of the saturation values may be within a range of 0 to 1 based on the above formula.

**[0032]** Based on the determination of a respective value represent the color property (e.g. saturation) of the title bars 143, 145, and 147, the color determiner 120 may designate the one of the title bars 143, 145, and 147 with a most extreme value of the color property (e.g. the title bar with the greatest saturation) as a dominant title bar. For example, title bar 145 may have the greatest saturation and therefore may be ranked as the dominant title bar. In some examples, according to the theory of visual saliency, human eyes may be better able to notice an area that has a high saturation and is a different color than the background. Therefore, error blocks may be generated with high saturation values so that a human may notice the error block in a first glance. Thus, a high saturation of a title bar may indicate that the window associated with that title bar is more likely to be an error block.

**[0033]** In some examples, for each of the identified windows 152, 154, and 156, the size determiner 122 may determine value representing a size of the window. For each of the identified windows 152, 154, and 156, the size may, for example, be an area within the respective border 136, 138, or 140. In some examples, the value representing the size (e.g. area) may be a number of pixels within with respective border 136, 138, or 140. This value may then be normalized by dividing by 1,000,000. In other examples, other measures of size may be used, for example a width or length of the window.

**[0034]** In some examples, for each of the identified windows 152, 154, and 156, the widget identifier 124 may identify any widgets in the window. In some examples, the widget identifier 124 may compare any features in the window with sample widgets in a widget library. The sample widgets may be expected to be associated with an error block. For example, the widgets in the widget library may include icons, for example an error icon or warning icon. If the widget identifier 124 determines a match between the feature in the window and a widget in the widget library, then the widget identifier 124 may classify the feature as an identified widget.

**[0035]** In some examples, the widget identifier 124 may use the SURF (Speeded Up Robust Features) algorithm to scan the windows and perform the comparison. Features extracted from an image may be detectable for different image sizes and scales, and different levels of noise and illumination in the image. Thus, the identified features may lie on high-contrast regions of the image, such as near the edge of a window.

**[0036]** In some examples, a widget may comprise multiple separated points of interest. For example, the four corners of a door may be used as four interest points with fixed relative distances between one another, such that the four corners could be identified irrespective of the location or scale of the door.

**[0037]** In the example of FIG. 4, the widget identifier 124 may identify features 150 and 151 in respective windows 152 and 154 as widgets. The features 150 and 151 may be associated with errors. However, the widget identifier 124 may not identify feature 148 as a widget, as feature 148 may not be associated with errors.

**[0038]** In some examples, the property identifier 118 may identify additional properties relative to those identified by the color determiner 120, size determiner 122, and widget determiner 124.

**[0039]** The error block determiner 126 may determine, for each identified window 152, 154, and 156, and based on the at least one property identified by the property determiner 118, whether the respective window is an error block. The error block determiner 126 may perform this task, for example, by using its score determiner 128 and score comparator 130.

**[0040]** In some examples, the score determiner 128 may determine a score representing a sum of a first weight multiplied by a first value representing the first property and a second weight multiplied by a second value representing the second property. In some examples, depending on the number of properties of the window detected by the property identifier 118, any additional number of weights multiplied by values representing properties may be added to the sum. In some examples, a constant may be added to the sum. In some examples, the score may be determined according to the formula:

$$score = w_1 \times saturation + w_2 \times number\ of\ widgets + w_3 \times window\ size + b.$$

**[0041]** Thus, in this example, the score of a window may represent a sum of a first weight multiplied by the saturation value  $S_{ab}$  determined by the color determiner 120, a second

weight multiplied by the number of widgets identified by the widget identifier 124, a third weight multiplied by the size of the window determined by the size determiner 122. In this example, the saturation value may be within the range of 0 and 1 as mentioned earlier, the number of widgets may be less than 10 in most examples, and the size may be the number of pixels in the window divided by 1,000,000 as mentioned earlier.

**[0042]** In some examples, a threshold score may define the score above which the window is to be classified as an error block. Therefore, in some examples, the score comparator 130 may, for each of the identified windows 152, 154, and 156, compare the score for the window to the threshold score. In addition, the score comparator 130 may determine which of the determined scores is the highest. In response to a score exceeding the threshold score and being the highest determined score associated with any window in the image 106, the score comparator 130 may determine that the window is an error block. This example may be used where no more than a single error block may be expected to pop up on a screen at a given time. In other examples, any windows whose scores exceed the score threshold, irrespective of whether the window is the highest determined score associated with any window in the image 106, the score comparator 130 may determine that the window associated with the score is an error block. This example may be used where multiple error blocks may be expected to pop up on a screen at a given time.

**[0043]** In the example of FIG. 4, the score comparator 130 may determine that the scores associated windows 152 and 154 exceed the threshold score but that the score associated with window 156 does not exceed the threshold score. Additionally, the score comparator 130 may determine that the score associated with window 154 is the highest score. Therefore, the score comparator 130 may determine that the window 154 is an error block. Additionally, in examples where any windows whose scores exceed the score threshold, irrespective of whether the window is the highest determined score associated with any window in the image 106, the score comparator 130 may determine that the window 152 is also an error block.

**[0044]** In some examples, the parameters  $w_1$ ,  $w_2$ ,  $w_3$ , and  $b$  may be based on a support vector machine (SVM) training model. For example, given a score threshold, and a training set of windows where it is known which of the windows are error blocks, the SVM training model may determine values for the parameters  $w_1$ ,  $w_2$ ,  $w_3$ , and  $b$  for use by the

score determiner 128 to generate optimal values to increase the true positive and true negative rates, and decrease the false positive and false negative rates.

**[0045]** In an experimental example, a training set of 300 windows were used by the SVM model to determine models the following values for the parameters:  $w_1 = 4.26395$ ,  $w_2 = 0.34763$ ,  $w_3 = -1.49507$ , and  $b = -0.77221$ . Additionally, in the experimental example, 1582 images were then analyzed using the error identification system 108 to yield 343 true positives (21.7%), 1116 true negatives (70.5%), 43 false positives (2.7%), and 80 false negatives (5.1%), for an overall correct identification rate of 92.2%.

**[0046]** In some examples, the OCR performer 132 may perform OCR on text in any windows identified to be error blocks. The text may include error messages. For example, the OCR performer 132 may perform OCR on text 158 in the window 154 identified as an error block. Additionally, in some examples, the OCR performer 132 may perform OCR on text 162 if the window 152 is also identified as an error block. However, the OCR performer 132 may not perform OCR on text 160, as window 156 may not be identified as an error block.

**[0047]** FIG. 6 is a flow diagram illustrating a method 300 according to some examples. In some examples, the orderings shown may be varied, some elements may occur simultaneously, some elements may be added, and some elements may be omitted. In describing FIG. 6, reference will be made to elements described in FIG. 4. In examples, any of the elements described earlier relative to FIG. 4 may be implemented in the process shown in and described relative to FIG. 6.

**[0048]** At 302, the method 300 may start.

**[0049]** At 304, the window identifier 110 may identify windows in the image 106, e.g. by using its edge identifier 112 to identify edges at 306, border identifier 114 to identify borders at 308, and title bar identifier 116 to identify title bars at 310. Any processes previously described as implemented by the window identifier 110, edge identifier 112, border identifier 114, and title bar identifier 116 may be implemented at 304, 306, 308, and 310.

**[0050]** At 312, the property identifier 118 may identify properties of the identified windows, e.g. by using its color determiner 120 to determine color properties (e.g. saturation) of title bars, size determiner 122 to determine sizes of windows, and widget

identifier 124 to identify widgets in the windows. Any processes previously described as implemented by the property identifier 118, color determiner 120, size determiner 122, and widget identifier 124 may be implemented at 312, 314, 316, and 318.

**[0051]** At 320, the error block determiner 126 may determine, for each identified window, and based on the at least one property identified by the property determiner 118, whether the respective window is an error block. The error block determiner 126 may perform this task, for example, by using its score determiner 128 at 322 and score comparator 130 at 322. Any processes previously described as implemented by the error block determiner 126, score determiner 128, size determiner 122, and score comparator 130 may be implemented at 320, 322, and 324.

**[0052]** At 326, if any of the scores (determined by score determiner 128) associated with any of the windows in an image exceeds the threshold score and is the highest score associated with any of the windows in the image, then the method 300 may identify the window associated with the highest score as an error block and proceed to 328. In some examples, the condition that the score is the highest score associated with any window in the image may not be used, such that e.g. multiple scores may be classified as error blocks. In these examples, the method may also proceed to 328. If none of the scores exceed the threshold score, then the method may proceed to 330.

**[0053]** At 328, the OCR performer 132 may perform OCR on text in any windows identified to be error blocks.

**[0054]** At 330, the method may end.

**[0055]** Any of the processors discussed herein may comprise a microprocessor, a microcontroller, a programmable gate array, an application specific integrated circuit (ASIC), a computer processor, or the like. Any of the processors may, for example, include multiple cores on a chip, multiple cores across multiple chips, multiple cores across multiple devices, or combinations thereof. In some examples, any of the processors may include at least one integrated circuit (IC), other control logic, other electronic circuits, or combinations thereof. Any of the non-transitory computer-readable storage media described herein may include a single medium or multiple media. The non-transitory computer readable storage medium may comprise any electronic, magnetic, optical, or other physical storage device. For example, the non-transitory computer-

readable storage medium may include, for example, random access memory (RAM), static memory, read only memory, an electrically erasable programmable read-only memory (EEPROM), a hard drive, an optical drive, a storage drive, a CD, a DVD, or the like.

**[0056]** All of the features disclosed in this specification (including any accompanying claims, abstract and drawings), and/or all of the elements of any method or process so disclosed, may be combined in any combination, except combinations where at least some of such features and/or elements are mutually exclusive.

**[0057]** In the foregoing description, numerous details are set forth to provide an understanding of the subject disclosed herein. However, examples may be practiced without some or all of these details. Other examples may include modifications and variations from the details discussed above. It is intended that the appended claims cover such modifications and variations.

**CLAIMS**

1. A non-transitory computer-readable storage medium comprising instructions executable by a processor to:
  - identify a window in an image;
  - identify at least one property of the window; and
  - determine, based on the at least one property, whether the window is an error block.
2. The non-transitory computer-readable storage medium of claim 1 wherein the at least one property is associated with a title bar of the window.
3. The non-transitory computer-readable storage medium of claim 2 wherein the at least one property comprises a saturation of the title bar.
4. The non-transitory computer-readable storage medium of claim 2 further comprising instructions executable by the processor to:
  - identify another window in an image; and
  - identify at least one other property of the other window,wherein the instructions to determine whether the window is an error block comprise instructions executable by the processor to determine, based on the at least one property and the at least one other property, which of the window and the other window is the error block.
5. The non-transitory computer-readable storage medium of claim 1 wherein the at least one property comprises a size of the window.
6. The non-transitory computer-readable storage medium of claim 1 wherein the at least one property comprises a widget in the window.

7. The non-transitory computer-readable storage medium of claim 6 wherein the instructions to identify the widget comprise instructions executable by the processor to:
- identify a feature in the window;
  - compare the feature to a sample widget in a widget library; and
  - classify the feature as the widget in response to the widget matching the sample widget,
- wherein the widget comprises an icon associated with an error.
8. The non-transitory computer-readable storage medium of claim 1 wherein the instructions to identify the window comprise instructions executable by the processor to identify a border and a title bar of the window.
9. The non-transitory computer-readable storage medium of claim 8 wherein the border is rectangular and encloses the window.
10. The non-transitory computer-readable storage medium of claim 9 wherein the instructions to identify the window comprise instructions executable by the processor to identify edges in the image that exceed a threshold length, and to determine that the edges form the border.
11. The non-transitory computer-readable storage medium of claim 1 further comprising instructions executable by the processor to perform, in response to the window being determined to be an error block, optical character recognition on the window to identify text in the window.
12. The non-transitory computer-readable storage medium of claim 1 wherein the at least one property comprises a first property and a second property, wherein the instructions to determine whether the window is an error block comprise instructions executable by the processor to:

determine a score representing a sum of a first weight multiplied by a first value representing the first property and a second weight multiplied by a second value representing the second property;

compare the score to a threshold score; and

in response to the score exceeding the threshold score, determine that the window is the error block.

13. The non-transitory computer-readable storage medium of claim 1 wherein the first weight and the second weight are based on a support vector machine (SVM) training model.

14. A system comprising:

a processor; and

a memory comprising instructions executable by the processor to:

determine that edges in an image form a window comprising a title bar and a rectangular border enclosing the window;

identify a plurality of properties of the window; and

determine whether the window is an error block based on the plurality of properties.

15. A method comprising:

by a processor:

identifying a window in an image;

determining at least two properties of the window selected from the group consisting of a number of widgets in the window, a saturation of a title bar of the window, and a size of the window;

determining a score based on values associated with the at least two properties; and

determining, based on the score, whether the window is an error block.

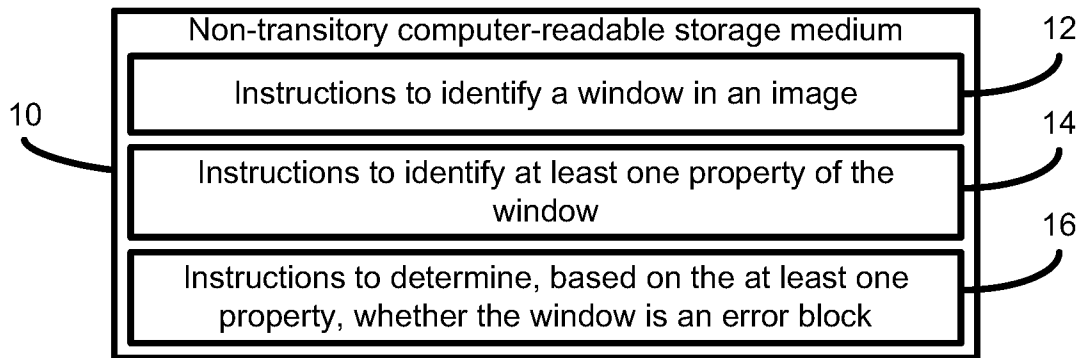


FIG. 1

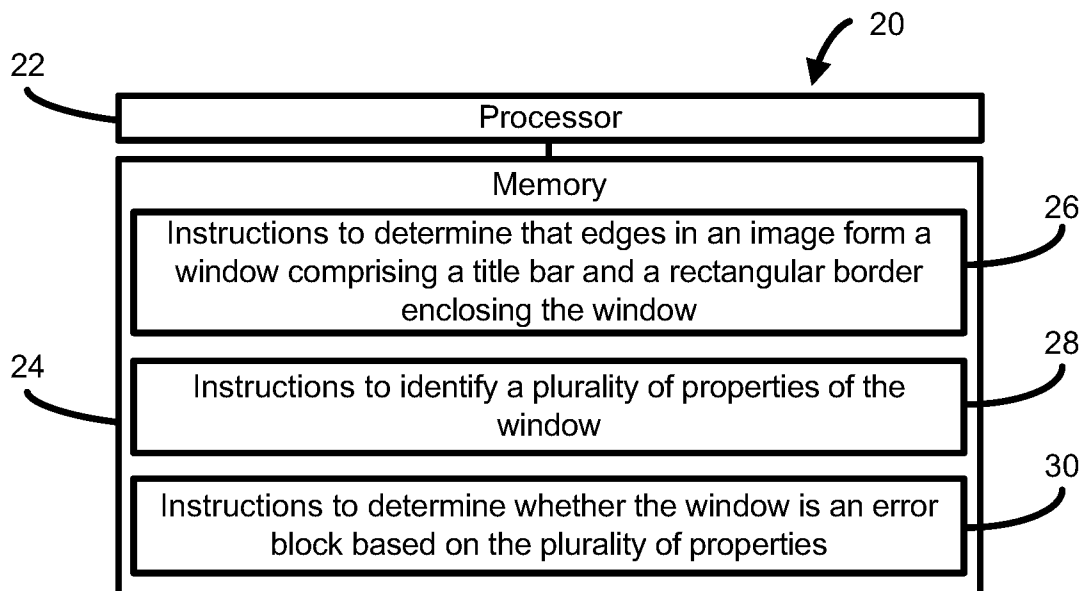


FIG. 2

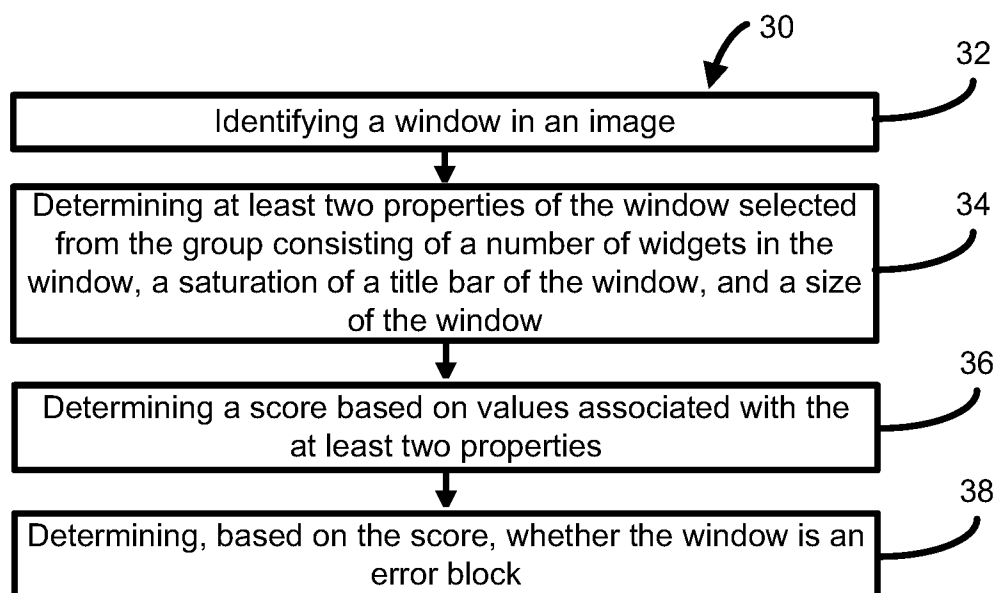


FIG. 3

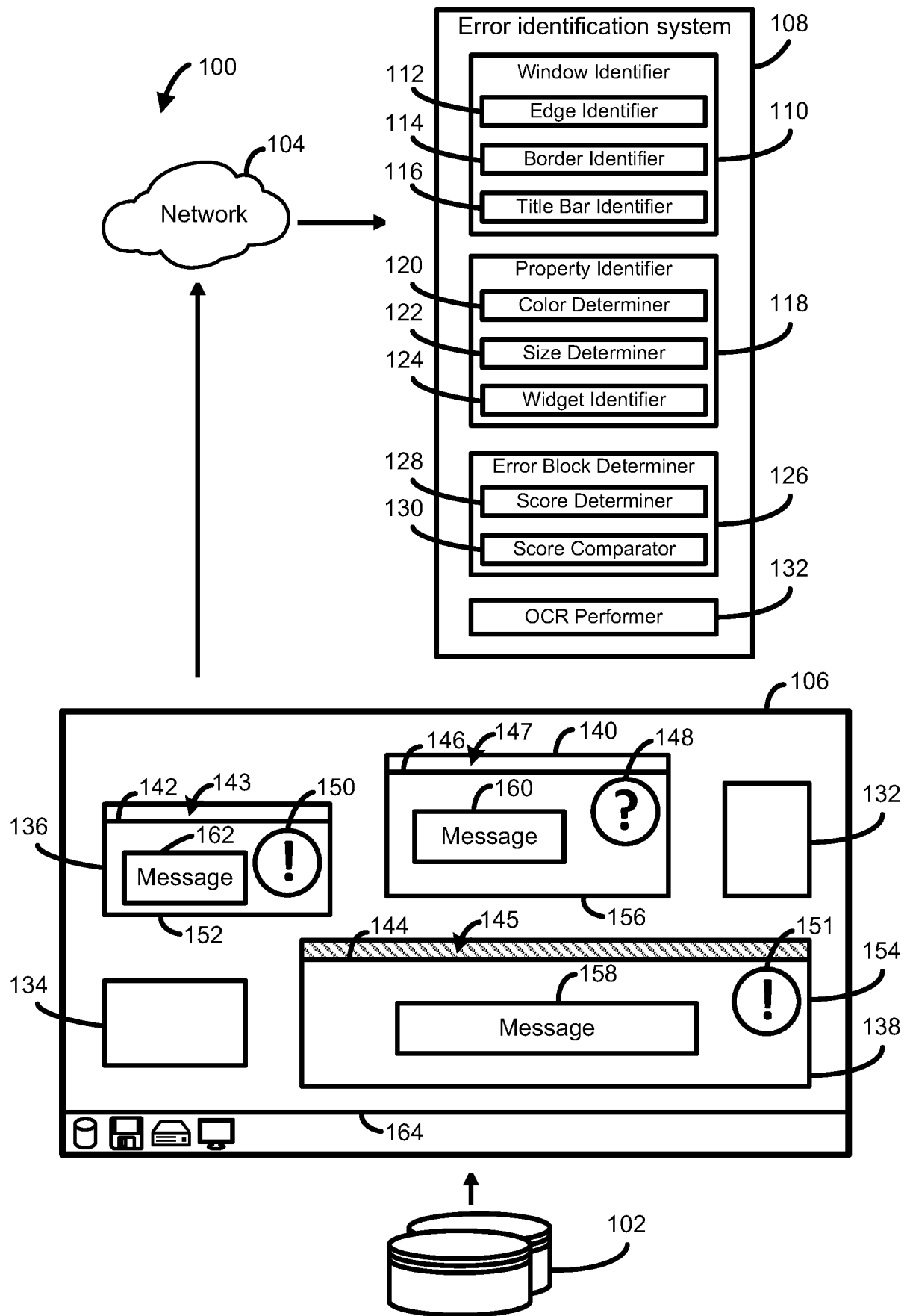


FIG. 4

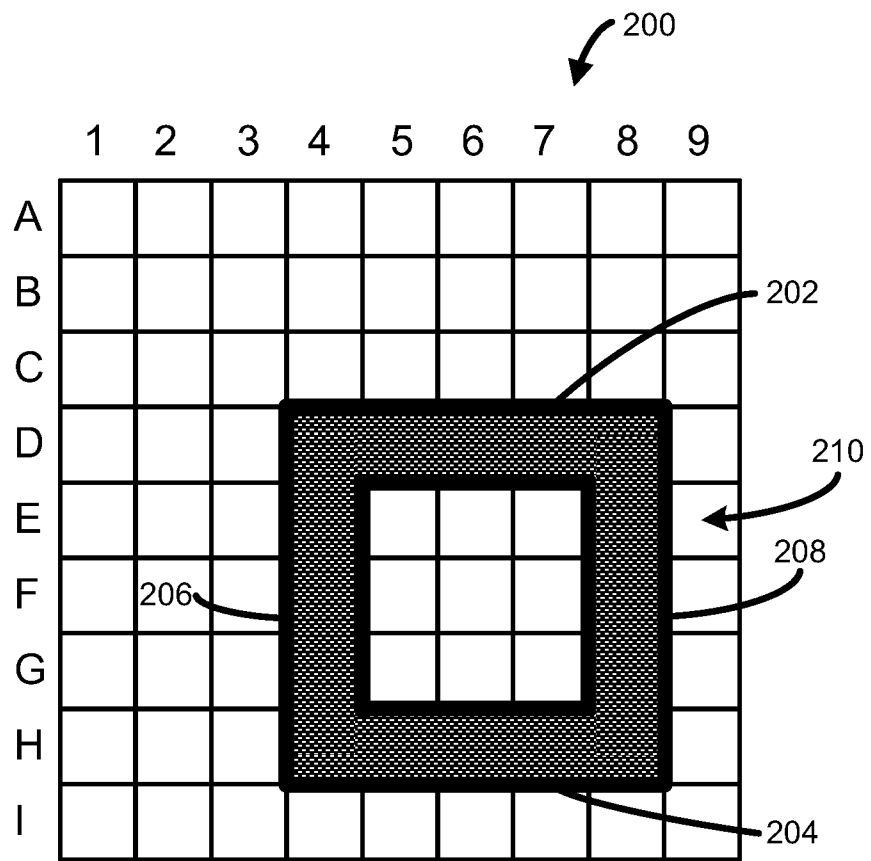


FIG. 5

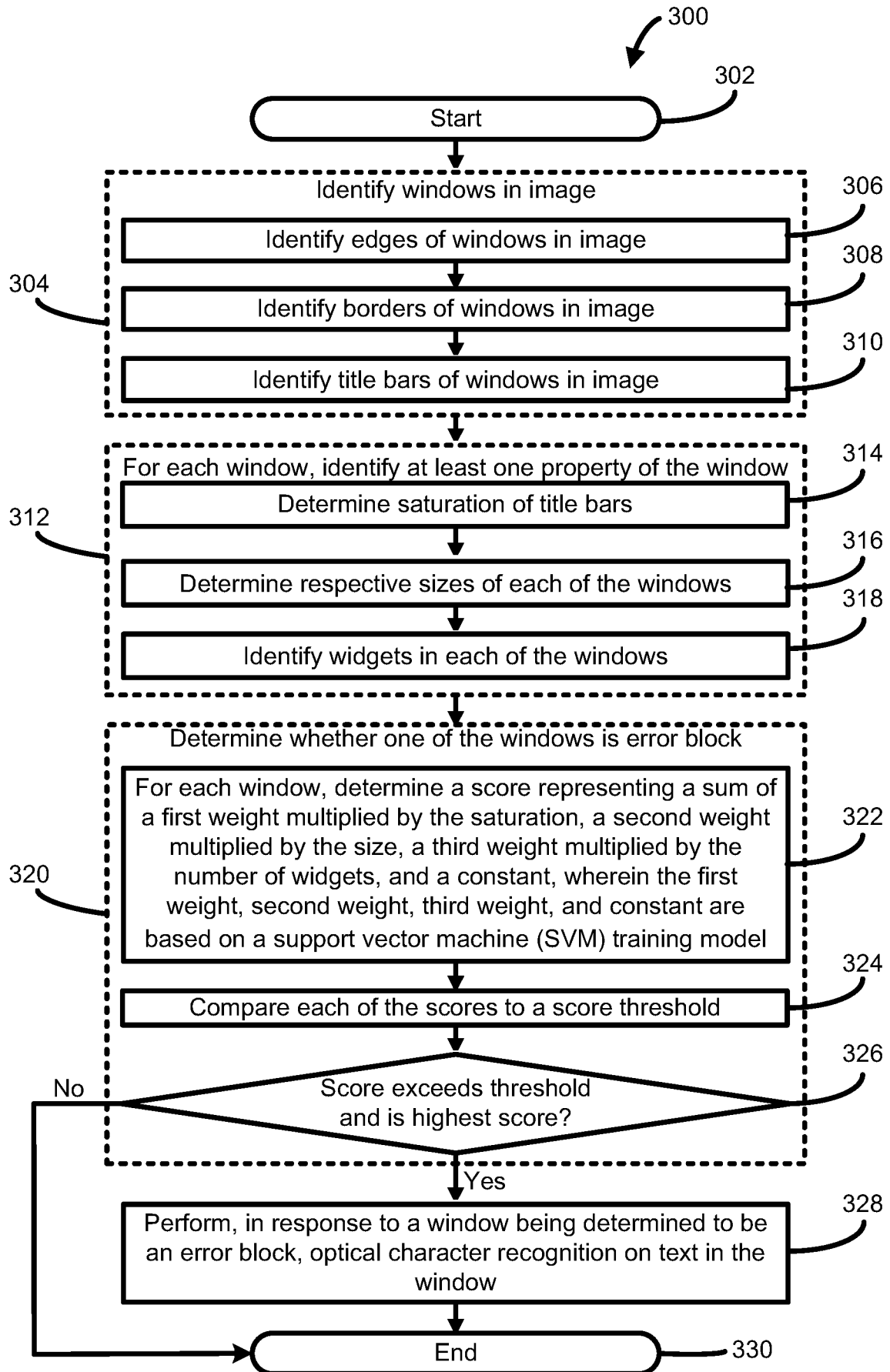


FIG. 6

## INTERNATIONAL SEARCH REPORT

International application No.

**PCT/CN2016/072664****A. CLASSIFICATION OF SUBJECT MATTER**

H04N 21/454(2011.01)i

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)

H04N, 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 practicable, search terms used)

WPI, EPODOC, CNPAT, CNKI: OCR, optical w character w recognition, error, mistake?, detect+, identif+, text, character?, window?, pop w up, message?, notice, monitor, screen, display+, title, saturation, border, edge, size, enclos+, propert+, feature?, score, compar+, weight?, SVM, train+, library, widget?, extract+.

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

| Category* | Citation of document, with indication, where appropriate, of the relevant passages                                                            | Relevant to claim No. |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------|-----------------------|
| X         | CN 103530561 A (BEIJING QIHOO TECHNOLOGY CO., LTD. ET AL.) 22 January 2014 (2014-01-22)<br>description, paragraphs [0017]-[0042] and figure 1 | 1-15                  |
| A         | CN 103019687 A (BEIJING QIHOO TECHNOLOGY CO., LTD. ET AL.) 03 April 2013 (2013-04-03)<br>the whole document                                   | 1-15                  |
| A         | CN 103458300 A (TIANJIN SAMSUNG ELECTRONICS CO., LTD. ET AL.) 18 December 2013 (2013-12-18)<br>the whole document                             | 1-15                  |
| A         | US 2012144282 A1 (LOEB, MICHAEL R. ET AL.) 07 June 2012 (2012-06-07)<br>the whole document                                                    | 1-15                  |



Further documents are listed in the continuation of Box C.



See patent family 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 application or patent 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 combined with one or more other such documents, such combination being obvious to a person skilled in the art

“&amp;” document member of the same patent family

Date of the actual completion of the international search

**21 September 2016**

Date of mailing of the international search report

**24 October 2016**

Name and mailing address of the ISA/CN

**STATE INTELLECTUAL PROPERTY OFFICE OF THE  
P.R.CHINA  
6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing  
100088  
China**

Authorized officer

**LI, Ping**

Facsimile No. (86-10)62019451

Telephone No. (86-10)82246959

**INTERNATIONAL SEARCH REPORT**  
**Information on patent family members**

International application No.

**PCT/CN2016/072664**

| Patent document<br>cited in search report |            |    | Publication date<br>(day/month/year) | Patent family member(s) |            |    | Publication date<br>(day/month/year) |
|-------------------------------------------|------------|----|--------------------------------------|-------------------------|------------|----|--------------------------------------|
| CN                                        | 103530561  | A  | 22 January 2014                      | None                    |            |    |                                      |
| CN                                        | 103019687  | A  | 03 April 2013                        | None                    |            |    |                                      |
| CN                                        | 103458300  | A  | 18 December 2013                     | None                    |            |    |                                      |
| US                                        | 2012144282 | A1 | 07 June 2012                         | US                      | 2015269220 | A1 | 24 September 2015                    |
|                                           |            |    |                                      | US                      | 8136028    | B1 | 13 March 2012                        |