



(51) International Patent Classification:
G06K 9/03 (2006.01) *H04N 1/60* (2006.01)

(21) International Application Number:
PCT/US2019/013233

(22) International Filing Date:
11 January 2019 (11.01.2019)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.** [US/US]; 10300 Energy Drive, Spring, Texas 77389 (US).

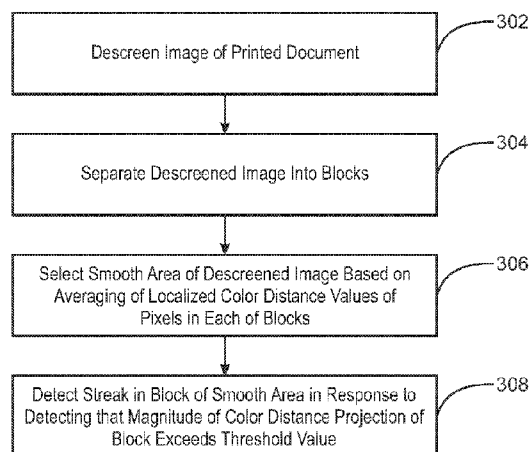
(72) Inventors: **ALLEBACH, Jan**; 11311 Chinden Blvd., Boise, Idaho 83714 (US). **MAGGARD, Richard E.**; 11311 Chinden Blvd, Boise, Idaho 83714 (US). **JESSOME, Renee J.**; 11311 Chinden Blvd, Boise, Idaho 83714 (US). **SHAW, Mark Q.**; 11311 Chinden Blvd, Boise, Idaho 83714 (US). **ZHANG, Runzhe**; 213 Halsey Dr, Apt 16, West Lafayette, Indiana 47906 (US).

(74) Agent: **SORENSEN, Blake C.** et al.; HP Inc., 3390 East Harmony Road, Mail Stop 35, Fort Collins, Colorado 80528 (US).

(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, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, 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.

(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, 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,

(54) Title: DETECTING STREAKS IN PRINTED DOCUMENTS USING BLOCKS



300
FIG. 3

(57) Abstract: In an example method, an image of a printed document is descreened. The descreened image is separated into blocks. A smooth area of the descreened image is selected based on an averaging of localized color distance values of pixels in each of the blocks. A streak in a block of the smooth area is detected in response to detecting that a magnitude of a color distance projection of the block exceeds a threshold value.

TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW,
KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

- *as to the identity of the inventor (Rule 4.17(i))*
- *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:

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

DETECTING STREAKS IN PRINTED DOCUMENTS USING BLOCKS

BACKGROUND

[0001] Electrophotographic printers are used to print documents onto print media. For example, electrophotographic printers can use electrostatically charged toner particles to print images onto the print media.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] Various features of the techniques of the present application will become apparent from the following description of examples, given by way of example only, which is made with reference to the accompanying drawings, of which:

[0003] Fig. 1 is a block diagram of an example system for detecting streaks using blocks and classifying streaks using extracted features;

[0004] Fig. 2 is a graph depicting an example sharpness feature calculation;

[0005] Fig. 3 is a process flow diagram illustrating an example method for detecting streaks using blocks;

[0006] Fig. 4 is a process flow diagram illustrating an example method for classifying streaks using extracted features;

[0007] Fig. 5A is block diagram of an example computing device to detect streaks using blocks and classify detected streaks using extracted features;

[0008] Fig. 5B is block diagram of another example computing device to detect streaks using blocks;

[0009] Fig. 6A is a drawing of an example machine-readable storage medium that can be used to detect streaks using blocks and classify detected streaks using extracted features; and

[0010] Fig. 6B is a drawing of another example machine-readable storage medium that can be used to detect streaks using blocks.

DETAILED DESCRIPTION

[0011] Electrophotographic printers are used to print documents onto print media. For example, a drum of the electrophotographic printer may be uniformly charged across its surface. An image of the document is then projected onto and synchronized with the moving drum surface. For example, in a laser or a light emitting diode (LED) printer, modulated light is projected onto the drum surface to create the latent image. The modulated light is used to create the positive image on the drum. The drum is then presented with toner that is electrostatically charged and attracted to the image. The print media is then passed between the drum and a transfer corona or a transfer roller with a polarity opposite of the charge of the toner. The toner image is then transferred from the drum to the paper by a combination of pressure and electrostatic attraction and permanently fixed to the print medium using either heat and pressure or a radiant fusing technology to melt and bond the toner into the medium. In some examples, the drum is discharged by light after fixing or fusing of the toner. Remaining toner on the drum can then be removed from the drum surface by a rotating brush under suction, or a squeegee known as a cleaning blade.

[0012] However, various issues may arise during this printing process resulting in defects such as streaks that can dramatically affect print quality. For example, streaks may be caused by a faulty organic photo conductor (OPC). In some examples, streaks may be caused by a faulty intermediate transfer belt (ITB). In further examples, the streaks may be caused by a faulty developer.

[0013] Described herein are techniques for detecting streaks using blocks. As used herein, a block refers to a portion of a printed image. For example, an image printed and scanned may be separated into a number of blocks. In some examples, the image may be descreened before being separated into blocks. In some examples, a smooth area of the descreened image is selected based on an averaging of localized color distance values of pixels in each of the blocks. A streak in a block of the smooth area is detected in response to detecting that a magnitude of a color distance projection of the block exceeds a threshold value. In particular, the techniques described herein can be used to test for vertical streaks caused by various root causes. In some examples, the techniques can also be used to identify a type of streak and a root cause of the streak.

In various examples, the detected root cause can be automatically repaired. Using logistic regression to remove false detections as described herein, the accuracy of the resulting streak detections was experimentally found to be 99.35%.

[0014] Fig. 1 is a block diagram of an example system for detecting streaks using blocks and classifying streaks using extracted features. The system 100 can be implemented using the methods 300 and 400 of Figs. 3 and 4, the computing device 502 of Figs. 5A and 5B, or the machine-readable storage medium 600 of Figs. 6A and 6B.

[0015] The system 100 includes a streak detector 102 communicatively coupled to a streak classifier 104. The system 100 includes an input test image 106. For example, the input test image 106 may be an image of a printed document. In some examples, the document may have been scanned using a scanner, such as an inline scanner of a printer. In some examples, the document may be scanned using a scanner of a multi-function printer (MFP). The streak detector 102 includes a preprocessor 112. The streak detector 102 also includes a smooth area selector 114 communicatively coupled to the preprocessor 112. The streak detector 102 further includes a block average delta E (ΔE) streak detector 116 communicatively coupled to the smooth area selector 114. The streak classifier 104 includes a feature vector extractor 118 communicatively coupled to a logistic regression model constructor 120.

[0016] In the example system 100, an input test image 106 is received from a scanner. For example, the input test image 106 may be received in real-time as a printer is printing out a document. In some examples, a streak defects detection algorithm (not shown) may monitor print quality as the printer prints out documents. In some examples, in response to detecting that print quality does not exceed a predetermined threshold, the streak detector 102 receives the input test image 106 for detecting streaks that may be causing the low print quality. The detection result 108 from the streak detector 102 may then be classified using the streak classifier 104 to classify the detected streak as a non-visible streak or a visible streak 110. In some examples, the streak classifier 104 also detects a type of the visible streak 110 and a root cause of the visible streak 110. In various examples, this information can be transmitted to a remote computer for further analysis.

[0017] The streak detector 102 can detect streaks in the input test image 106 using blocks. In particular, the preprocessor 112 receives the input test image 106. In some examples, the input test image 106 is in a sRGB color space and is gamma corrected. In various examples, the preprocessor can perform gamma un-correction and convert the color space from sRGB to CIE L*a*b*. The preprocessor 112 also descreens the input test image 106 to remove halftone patterns. For example, the preprocessor 112 can use a Gaussian filter or any other suitable filter or algorithm as part of a descreening algorithm to remove the halftone patterns. As used herein, a Gaussian filter is a filter whose impulse response is a Gaussian function or an approximation to the Gaussian function. In some examples, the Gaussian filter size is 15 x 15 pixels and the standard deviation $\sigma = 2$ pixels. As one example, the preprocessor 112 can descreen the input test image 106 using the equation:

$$f'(x, y, c) = \sum_{i=-7}^7 \sum_{j=-7}^7 f(x + i, y + j, c) g(i, j) \quad \text{Eq. 1}$$

where $f(x, y, c)$ is the input image in the CIE L*a*b* color space, $g(i, j)$ is the 15 x 15 Gaussian filter, and $f'(x, y, c)$ is the descreened result. The preprocessor 112 can then separate the descreened image into a number of blocks of predetermined size. For example, the blocks may have a predetermined size of 200 x 200 pixels.

[0018] The smooth area selector 114 receives descreened blocks of the input test image 106 and selects a smooth area on which to perform streak detection. For example, the smooth area can be selected such that streak defects are included in the smooth area but content of the printed document are excluded from the smooth area. For example, the content may be custom content including vertical lines that could be mistaken for streaks.

[0019] In various examples, the smooth area selector 114 selects the smooth area of the input test image 106 using an average delta E (ΔE) value of each block. ΔE is a metric for measuring color difference as perceived by the human eye. For example, the smooth area selector 114 can calculate ΔE of each pixel in one of the blocks using the equations:

$$\Delta E(i, j) = \sqrt{(L_{(i,j)} - L_{ave})^2 + (a_{(i,j)} - a_{ave})^2 + (b_{(i,j)} - b_{ave})^2} \quad \text{Eq. 2}$$

$$L_{ave} = \frac{\sum_{i=1}^M \sum_{j=1}^M L(i,j)}{M^2} \quad \text{Eq. 3}$$

$$a_{ave} = \frac{\sum_{i=1}^M \sum_{j=1}^M a(i,j)}{M^2} \quad \text{Eq. 4}$$

$$b_{ave} = \frac{\sum_{i=1}^M \sum_{j=1}^M b(i,j)}{M^2} \quad \text{Eq. 5}$$

where the subscript (i, j) is the pixel position in the image, the subscript ave is the average value of each channel value in one block, and M is the block size. The smooth area selector 114 then uses the ΔE values of the pixels in each block to calculate an average ΔE value for each block. For example, the smooth area selector 114 can calculate an average ΔE value for each block using the equation:

$$\Delta E_{block} = \frac{\sum_{i=1}^M \sum_{j=1}^M \Delta E(i,j)}{M^2} \quad \text{Eq. 6}$$

In addition to calculating the average ΔE values ΔE_{block} of the test image, the smooth area selector 114 may receive an average ΔE value threshold for average ΔE values ΔE_{block} used to select the smooth area of the input test image 106. In various examples, the average ΔE value threshold is a manually set parameter. If the average ΔE value threshold is too small, the streak defect will belong to the unsmooth area, otherwise, many small customer contents will belong to the smooth area. As one example, the threshold is 2. In some examples, the average ΔE value threshold can be verified by the streak feature used to calculate the severity of streaks as described with respect to Eq. 8. In various examples, the average ΔE value threshold can be automatically configured based on the streak feature. For example, the average ΔE value threshold can be automatically configured using a classifier, such as a state vector machine (SVM), a neural network, or other classification algorithm.

[0020] The block average ΔE streak detector 116 can then detect streaks based on the using ΔE projection. For example, as streaks are vertical lines in input test images 106, with vertical referring to the same direction to the printing process direction, the block average ΔE streak detector 116 can project the ΔE of each column in a block along the printing process direction. The magnitude of ΔE streak projection is higher for

detected streaks than areas without streaks. For example, the magnitude of the ΔE streak projection may be almost twice the ΔE streak projection value of the smooth area. Thus, to distinguish the streak from the smooth area, a projection threshold $T_{\Delta E_{proj}}$ may be set according to the equation:

$$T_{\Delta E_{proj}} = \Delta E_{block} \times M \quad \text{Eq. 7}$$

where M is the size of the block. As one example, the block is 200 pixels x 200 pixels. In various examples, the block average ΔE streak detector 116 marks projection magnitudes exceeding the projection threshold as detected streak defects. In some examples, once an entire input test image 106 is processed by the streak detector 102, a detection result 108 is output. For example, the detection result 108 includes streak defects detected by the streak detector 102.

[0021] The detection result 108 includes all the streaks visible to naked eye, but may also include many noise detections. The noise detections may be a result of using a single threshold projection to detect a number of different types of streaks. For example, the various types of streaks may include a developer streak, an ITB streak, an OPC streak, or a dirty scanner streak. The threshold may thus be set to include various types of streaks as well as some noise detections. The noise detections may be very small defects that may be difficult to detect by the human eye or short vertical lines of less than 1/3 of an inch. These noise detections may thus be labeled as streaks, but may be too small to be seen by human eye. In various examples, the streak classifier 104 can use a logistic regression algorithm to remove the noise detections from the detection results 108 and output visible streaks 110.

[0022] The streak classifier 104 includes a feature vector extractor 118 to extract feature vectors from the detection result 108. In various examples, a connected component algorithm can label different streak detection results before the features are extracted by the feature vector extractor 118. The connected component algorithm is an algorithm used to detect individual objects. For example, the connected components algorithm labels the pixels in detected individual objects with internal ID numbers. As one example, the internal ID numbers are numbers that are consecutively generated for each detected component.

[0023] The feature vector extractor 118 can then extract various features from the detection result 108. For example, the feature vector extractor 118 can extract an area of the detection result 108. The area may be equal to the number of pixels N included in each detection result 108.

[0024] In some examples, the feature vector extractor 118 extracts a length to width ratio of the streak detection result 108. For example, because the streak is a vertical line in the input test image 106, the greater length-width ratio of the detection result, the greater the probability of a streak defect. In various examples, the feature vector extractor 118 accesses the length and width of each streak detection result from the connected component calculation process. For example, the connected components calculation can extract a region of the object that includes a height in pixels, a width in pixels, and an area. In some examples, the connected components calculation performs a more detailed analysis that includes a calculation of the width of the object, angle of the max length, aspect ratio of height to width, average color, and other features that define each detected object.

[0025] In various examples, the feature vector extractor 118 extracts a severity of the streak detection result. For example, the feature vector extractor 118 can extract the severity based on the area of the streak detection result and the average ΔE value of the streak detection result 108. As one example, the feature vector extractor 118 can extract the severity feature S using the equation:

$$S = \Delta E_{streak-ave} \times N \quad \text{Eq. 8}$$

where N is the number of pixels included in the streak detection result 108.

[0026] In some examples, the feature vector extractor 118 extracts a luminance channel difference. For example, lightness can be used to classify streaks as light streaks versus dark streaks. Because the luminosity (L^*) channel represents the lightness value, the feature vector extractor 118 calculates the average value of the luminosity channel in the smooth area. Then, the feature vector extractor 118 calculates the luminosity channel difference of each streak detection result 108. As one example, the feature vector extractor 118 can calculate the luminosity channel difference using the equation:

$$\bar{L}_{streak} = \frac{\sum_{i=1}^N L_i - \bar{L}_{smooth}}{N} \quad \text{Eq. 9}$$

where $\bar{L}_{smooth}$ is the average value of the luminosity channel in the smooth area. In some examples, if $\bar{L}_{streak}$ is positive, then the streak detection result 108 is a light streak. In various examples, if $\bar{L}_{streak}$ is negative, then the streak detection result 108 is a dark streak.

[0027] Still referring to Fig. 1, in some examples, the feature vector extractor 118 extracts a sharpness of the streaks. Sharpness is defined by the boundaries between zones of different tones or colors. As one example, the feature vector extractor 118 calculates sharpness using the rise distance of the edge. For example, the rise distance is the distance in pixels, millimeters, or fraction of image height for the pixel level to change from 10% to 90% of its final value. This sharpness calculation method is referred to herein as a 10 - 90% rise distance. To calculate the 10 - 90% rise distance, the feature vector extractor 118 first projects the ΔE value of each streak detection result 108. Then, the feature vector extractor 118 can calculate the sharpness of streak S_{streak} using the equation:

$$S_{streak} = \frac{2 \times b}{(a_1 + a_2)} \quad \text{Eq. 10}$$

where a_1 is the number of pixels in the width of an up edge and a_2 is the number of pixels in the width of a down edge, and b is the value of 80% of the rise distance. An example sharpness feature calculation is described in detail with respect to in Fig. 2.

[0028] In various examples, after the feature vector extractor 118 extracts the various features from the detection result 108, the feature vector extractor 118 can normalize each of the extracted features for logistic regression classification. For example, the absolute value can be used to scale the change in luminance before the logistic regression is applied. In various examples, every feature is normalized to a range of values from zero to one, corresponding the lowest value and the highest value of each of each of the features. For example, a value is applied to the aspect ratio to scale the value of the aspect ratio to a defined range.

[0029] The block diagram of Fig. 1 is not intended to indicate that the example system 100 is to include all of the components shown in Fig. 1. Further, the system 100 may include any number of additional components not shown in Fig. 1, depending on the details of the specific implementation.

[0030] Fig. 2 is a graph depicting an example sharpness feature calculation. The sharpness feature calculation 200 can be performed using the feature vector extractor 118 of Fig. 1, the feature extractor 524 and processor 504 of the computing device 502 of Fig. 5A or the processor 602 of the machine-readable storage medium 600 of Fig. 6A.

[0031] The graph of Fig. 2 includes a first axis representing delta E projection values 202 and a second axis representing scan direction pixels 204. A first horizontal line indicates a value of 10% 206 of a peak value 208. A second horizontal line indicates a value of 90% 210 of the peak value 208. A horizontal double ended arrow indicates a width a_1 212 of an up edge 214. A second horizontal double-ended arrow indicates a width a_2 216 of a down edge 218. A pair of vertical lines indicate a b 220.

[0032] As shown in Fig. 2, a sharpness feature can be calculated using Eq. 10. In this example, the peak value 208 of the delta E projection value is approximately 8400. The 90% peak value 210 is therefore 7560 and the 10% peak value is 840. The 80% value b 220 is therefore 6720. Moreover, the width a_1 212 of the up edge 214 is approximately 3.8 and the width a_2 216 of the down edge 218 is approximately 3.4. Thus, the sharpness feature may be calculated as $\frac{2 \times 6720}{(3.8 + 3.4)} = 1866.667$.

[0033] It is to be understood that the graph of Fig. 2 is not intended to indicate that all of the elements of the sharpness feature calculation 200 are to be included in every case. Further, any number of additional elements not shown in Fig. 2 may be included in the sharpness feature calculation 200, depending on the details of the specific implementation. For example, the sharpness feature calculation 200 may include the use of different percentages of peak value, different up edges and down edges, etc.

[0034] Fig. 3 is a process flow diagram illustrating an example method for detecting streaks using blocks. The method 300 of Fig. 3 can be implemented using the streak detector 102 of Fig. 1, the streak detector 522 and processor 504 of the computing

device 502 of Figs. 5A and 5B or the processor 602 of the machine-readable storage medium 600 of Figs. 6A and 6B.

[0035] At block 302, an image of a printed document is descreened. In some examples, halftone patterns are removed from the image using a Gaussian filter. In various examples, the halftone patterns are removed from the image using a median filter or a human visual system (HVS) filter.

[0036] At block 304, the descreened image is separated into blocks. In some examples, the descreened image is separated into blocks of a predetermined size.

[0037] At block 306, a smooth area of the descreened image is selected based on an averaging of localized color distance values of pixels in each of the blocks.

[0038] At block 308, a streak is detected in a block of the smooth area based on a color distance projection of the block. For example, the streak can be detected in response to detecting that a projection threshold is exceeded. In some examples, the projection threshold is preset or automatically configured based on a streak feature.

[0039] It is to be understood that the process diagram of Fig. 3 is not intended to indicate that all of the elements of the method 300 are to be included in every case. Further, any number of additional elements not shown in Fig. 3 may be included in the method 300, depending on the details of the specific implementation. For example, the method may include extracting features and classifying the detected streaks as described with respect to Fig. 4.

[0040] Fig. 4 is a process flow diagram illustrating an example method for classifying streaks using extracted features. The method 400 of Fig. 4 can be implemented using the streak classifier 104 of Fig. 1, the logistic regression classifier 526 and processor 504 of the computing device 502 of Fig. 5A or the processor 602 of the machine-readable storage medium 600 of Fig. 6A.

[0041] At block 402, features are extracted from the detected streak. For example, the features can include an area of the detected streak, a length to width ratio of the detected streak, a severity of the detected streak, a luminance channel difference, a sharpness of the detected streak, or any combination thereof. In some examples, the features may be extracted and used to generate a feature vector including a number of features.

[0042] At block 404, the streak is classified as a visible streak defect or an invisible streak defect based on the extracted features. In various examples, the detected streak is processed via a trained logistic regression classifier to classify the detected streak as a visible streak defect or an invisible streak defect. For example, the logistic regression classifier may be trained with a logistic loss function using features extracted from training images of various streaks with known causes.

[0043] At block 406, a type of streak and a root cause of the detected streak is detected based on the extracted features. For example, the type of streak may be a developer streak, an ITB streak, an OPC streak, or a dirty scanner streak. In various examples, the type of streak and a root cause of the type of defect via the trained logistic regression classifier based on the extracted features.

[0044] It is to be understood that the process diagram of Fig. 4 is not intended to indicate that all of the elements of the method 400 are to be included in every case. Further, any number of additional elements not shown in Fig. 4 may be included in the method 400, depending on the details of the specific implementation. For example, the detected root cause can be automatically repaired in response to detecting the type of streak and root cause of the detected streak.

[0045] Fig. 5A is a block diagram of an example computing device 502 to detect streaks using blocks and classify detected streaks using extracted features. The computing device 502 may include a processor 504, memory 506, a machine-readable storage 508, and a network interface 510 to connect computing device 502 to network 512. For example, the network interface 510 can be a network interface card (NIC).

[0046] In some examples, the processor 504 may be a main processor that is adapted to execute the stored instructions. Moreover, more than one processor 504 may be employed. Further, the processor 504 may be a single core processor, a multi-core processor, a computing cluster, or any number of other configurations. The processor 504 may be implemented as Complex Instruction Set Computer (CISC) or Reduced Instruction Set Computer (RISC) processors, x86 Instruction set compatible processors, ARMv7 Instruction set compatible processors, multi-core, or any other microprocessor or central processing unit (CPU).

[0047] The memory 506 may be a memory device. The memory 506 may be volatile memory or nonvolatile memory. In some examples, the memory 506 may include random access memory (RAM), cache, read only memory (ROM), flash memory, and other memory systems.

[0048] The storage 508 is machine-readable storage and may include volatile and nonvolatile memory. In some examples, the machine-readable storage 508 may be electronic, magnetic, optical, or other physical storage device that stores executable instructions (e.g., code, logic). Thus, the machine-readable storage 508 medium may be, for example, RAM, an Electrically-Erasable Programmable Read-Only Memory (EEPROM), a storage drive such as a hard drive or solid state drive (SSD), an optical disc, and the like. The storage 508 may also include storage or memory external to the computing device 502. Moreover, as described herein, the machine-readable storage medium 508 may be encoded with executable instructions (e.g., executed by the processor 504) for detecting streaks using blocks and classifying detected streaks using extracted features. For example, the machine-readable storage medium 508 may be encoded with executable instructions for detecting streaks using blocks and classifying streaks using extracted features.

[0049] In some examples, a network interface 510 (e.g., a network interface card or NIC) may couple the computing device 502 to a network 512. For example, the network interface 510 may connect computing device 502 to a local network 512, a virtual private network (VPN), or the Internet. In some examples, the network interface 510 may include an Ethernet controller.

[0050] The computing device 502 may also include an electrophotographic printing mechanism 514. For example, the electrophotographic printing mechanism 514 may include a drum, a laser or LED, and toner.

[0051] The computer device 502 may include a scanner 516. In some examples, the scanner 516 can be an inline scanner. For example, the inline scanner may be installed at an output end of a printer to scan test images in real-time. In some examples, the scanner 516 can be a scanner of a multi-function printer (MFP).

[0052] The computing device 502 includes, an image preprocessor 518, a smooth area selector 520, a streak detector 522, a feature extractor 524, and a logistic

regression classifier 526. The image preprocessor 518 descreens an image of a printed document. The image preprocessor 518 also separates the descreened image into blocks. The smooth area selector 520 selects a smooth area of the descreened image based on an averaging of localized color distance values of pixels in each of the blocks. For example, the smooth area selector 520 can calculate an average ΔE value for each block using the equation 6, as described with respect to Fig. 1. The smooth area selector 520 can then select a smooth area based on a received or calculated average ΔE value threshold. The streak detector 522 detects a streak in a block of the smooth area based on a color distance projection of the block. In various examples, the streak detector 522 can detect streaks in response to detecting that a projection threshold is exceeded. For example, the projection threshold can be calculated using Eq. 7. The feature extractor 524 extracts features from the detected streak. For example, the feature extractor 524 can extract a number of features including an area of the detected streak, a length to width ratio of the detected streak, a severity of the detected streak, a luminance channel difference, a sharpness of the detected streak, or any combination thereof. In some examples, the feature extractor 524 normalizes the number of features, and processes the normalized number of features via a trained logistic regression classifier to classify the detected streak. The logistic regression classifier 526 classifies the detected streak as a visible streak defect or an invisible streak defect based on the extracted features. In some examples, the logistic regression classifier 526 classifies a type of streak and a root cause of the detected streak. For example, the type of streak may be a developer streak, an ITB streak, an OPC streak, or a dirty scanner streak.

[0053] The image preprocessor 518, smooth area selector 520, the streak detector 522, the feature extractor 524, and the logistic regression classifier 526 may be instructions (e.g., code, logic, etc.) store in the machine-readable storage 508 and executed by the processor 504 or other processor to direct the computing device 502 to implement the aforementioned actions. An application-specific integrated circuit (ASIC) may also be employed. In other words, an ASIC may be customized for the aforementioned actions implemented via the image preprocessor 518, smooth area

selector 520, the streak detector 522, the feature extractor 524, and the logistic regression classifier 526.

[0054] The block diagram of Fig. 5A is not intended to indicate that the computing device 502 is to include all of the components shown in Fig. 5A. Further, the computing device 502 may include any number of additional components not shown in Fig. 5A, depending on the details of the specific implementation. For example, the computing device 502 can include additional filters, etc.

[0055] Fig. 5B is block diagram of another example computing device to detect streaks using blocks. The example computing device 502 of Fig. 5B includes similarly numbered elements as in Fig. 5A.

[0056] The block diagram of Fig. 5B is not intended to indicate that the computing device 502 is to include all of the components shown in Fig. 5B. Further, the computing device 502 may include any number of additional components not shown in Fig. 5B, depending on the details of the specific implementation. For example, the computing device 502 can include additional filters, etc.

[0057] Fig. 6A is a block diagram showing a tangible, non-transitory, machine-readable storage medium that stores code to direct a processor to detect streaks using blocks and classify detected streaks using extracted features. The machine-readable medium 600 can include RAM, a hard disk drive, an array of hard disk drives, an optical drive, an array of optical drives, a non-volatile memory, a flash drive, a digital versatile disk (DVD), or a compact disk (CD), among others. The machine-readable storage medium 600 may be accessed by a processor 602 over a bus 604. The processor 602 may be a processor of a computing device, such as the processor 504 of Figs. 5A and 5B. In some examples, the processor 602 may be a field-programmable gate array (FPGA) processor and/or an ASIC processor. Furthermore, as indicated, the machine-readable medium 600 may include code configured to perform the methods and techniques described herein. Indeed, the various logic components discussed herein may be stored on the machine-readable medium 600. Portions 606, 608, 610, 612, and 614 of the machine-readable storage medium 600 can include image preprocessor code, smooth area detector code, streak detector code, feature extractor code, and logistic regression classifier code, respectively, which may be executable code

(machine readable instructions) that direct a processor or controller in performing the techniques discussed with respect to the preceding figures.

[0058] Indeed, the various logic (e.g., instructions, code) components discussed herein may be stored on the tangible, non-transitory machine-readable medium 600 as indicated in Figs. 6A and 6B. For example, the machine-readable medium 600 may include the image preprocessor module 606 that, when executed by a processor, direct the processor or a computing device to descreen an image of a printed document. In some examples, the image preprocessor module 606 can also direct the processor or a computing device to separate the descreened image into blocks. The machine-readable medium 600 may also include a smooth area selector module 608 that when executed by a processor to direct the processor or a computing device to select a smooth area of the descreened image based on an averaging of localized color distance values of pixels in each of the blocks. For example, an average ΔE value for each block can be calculated using Eq. 6 as discussed in Fig. 1. The machine-readable medium 600 may further include the streak detector module 610 that, when executed by a processor, direct the processor or a computing device to detect a streak in a block of the smooth area in response to detecting that a magnitude of a color distance projection of the block exceeds a threshold value. The machine-readable medium 600 may also include feature extractor module 612 that, when executed by a processor, direct the processor or a computing device to extract features from the detected streak. For example, the extracted features may include an area of the detected streak, a length to width ratio of the detected streak, a severity of the detected streak, a luminance channel difference, a sharpness of the detected streak, or any combination thereof.

[0059] Fig. 6B is a drawing of another example machine-readable storage medium that can be used to detect streaks using blocks. The machine-readable medium 600 of Fig. 6B includes similarly numbered elements of fig. 6A.

[0060] Although shown as contiguous blocks, the logic components may be stored in any order or configuration. For example, if the machine-readable medium 600 is a hard drive, the logic components may be stored in non-contiguous, or even overlapping, sectors.

[0061] While the present techniques may be susceptible to various modifications and alternative forms, the examples discussed above have been shown only by way of example. It is to be understood that the technique is not intended to be limited to the particular examples disclosed herein. Indeed, the present techniques include all alternatives, modifications, and equivalents falling within the true spirit and scope of the appended claims.

CLAIMS

What is claimed is:

1. A method comprising:
descreening an image of a printed document;
separating the descreened image into blocks;
selecting a smooth area of the descreened image based on an averaging of
localized color distance values of pixels in each of the blocks; and
detecting a streak in a block of the smooth area in response to detecting that a
magnitude of a color distance projection of the block exceeds a threshold
value.
2. The method of claim 1, comprising processing the detected streak via a
trained logistic regression classifier to classify the detected streak as a visible streak
defect or an invisible streak defect.
3. The method of claim 2, comprising training the logistic regression classifier
with a logistic loss function using feature vectors extracted from training images of
various streaks with known causes.
4. The method of claim 2, comprising extracting features from the detected
streak and processing the extracted features via the trained logistic regression
classifier.
5. The method of claim 4, comprising detecting a type of streak and a root
cause of the type of streak via the trained logistic regression classifier based on the
extracted features.
6. The method of claim 1, wherein descreening the image comprises
removing halftone patterns from the image using a Gaussian filter.

7. A system, comprising a processor to:
descreen an image of a printed document;
separate the descreened image into blocks;
select a smooth area of the descreened image based on an averaging of
localized color distance values of pixels in each of the blocks; and
detect a streak in a block of the smooth area based on a color distance projection
of the block.
8. The system of claim 7, wherein the processor is to extract features from
the detected streak and classify the detected streak as a visible streak defect or an
invisible streak defect based on the extracted features.
9. The system of claim 7, wherein the processor is to extract features from
the detected streak and classify a type of streak and a root cause of the detected
streak.
10. The system of claim 7, wherein the processor is to extract a feature from
the detected streak comprising an area of the detected streak.
11. The system of claim 7, wherein the processor is to extract a feature from
the detected streak comprising a length to width ratio of the detected streak.
12. The system of claim 7, wherein the processor is to extract a feature from
the detected streak comprising a severity of the detected streak.
13. The system of claim 7, wherein the processor is to extract a feature from
the detected streak comprising a luminance channel difference.
14. The system of claim 7, wherein the processor is to extract a feature from
the detected streak comprising a sharpness of the detected streak.

15. The system of claim 7, wherein the processor is to extract a plurality of features from the detected streak, normalize the plurality of features, and process the normalized plurality of features via a trained logistic regression classifier to classify the detected streak.

1/8

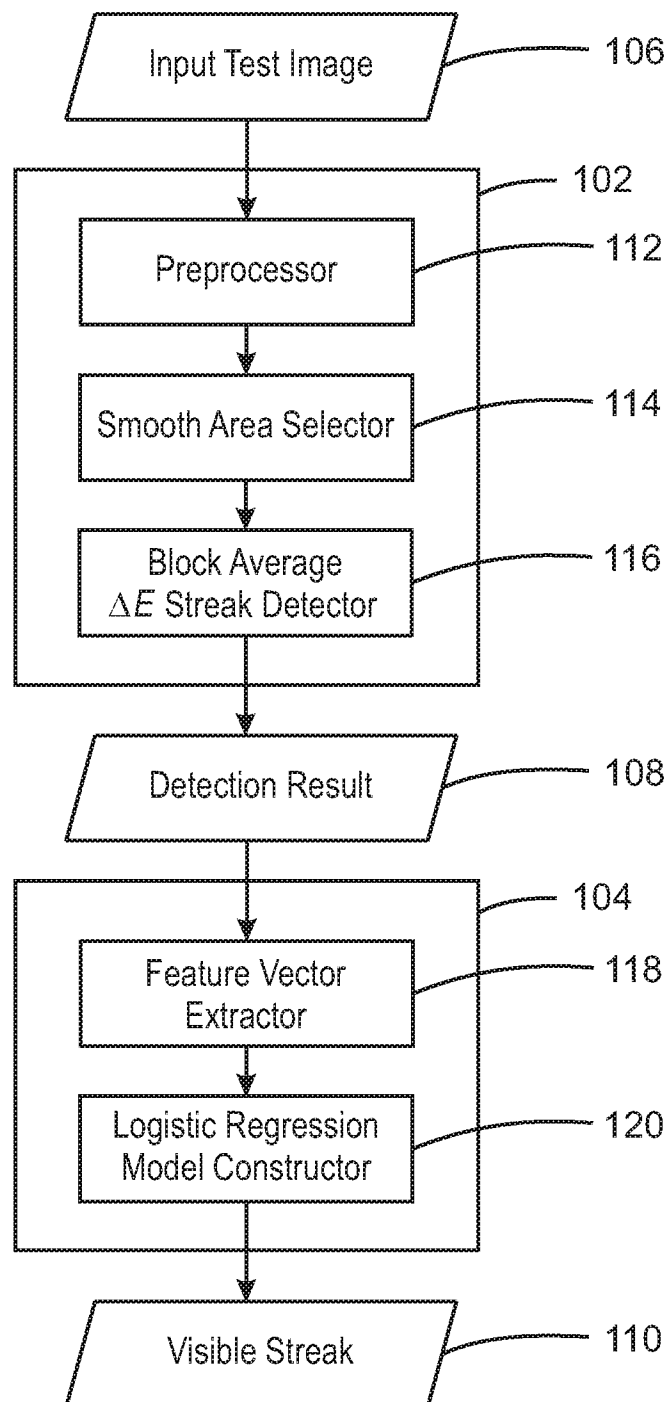
100

FIG. 1

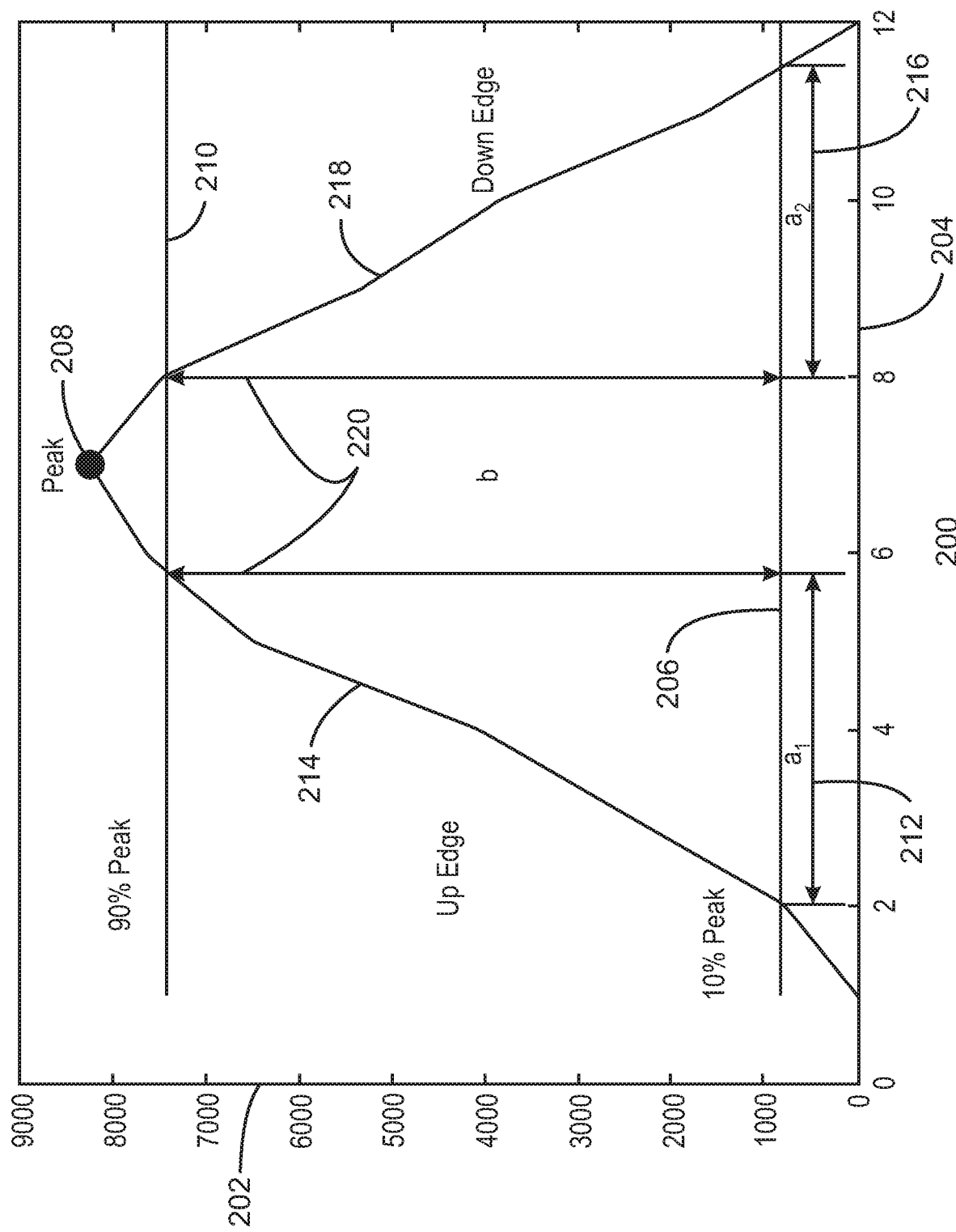
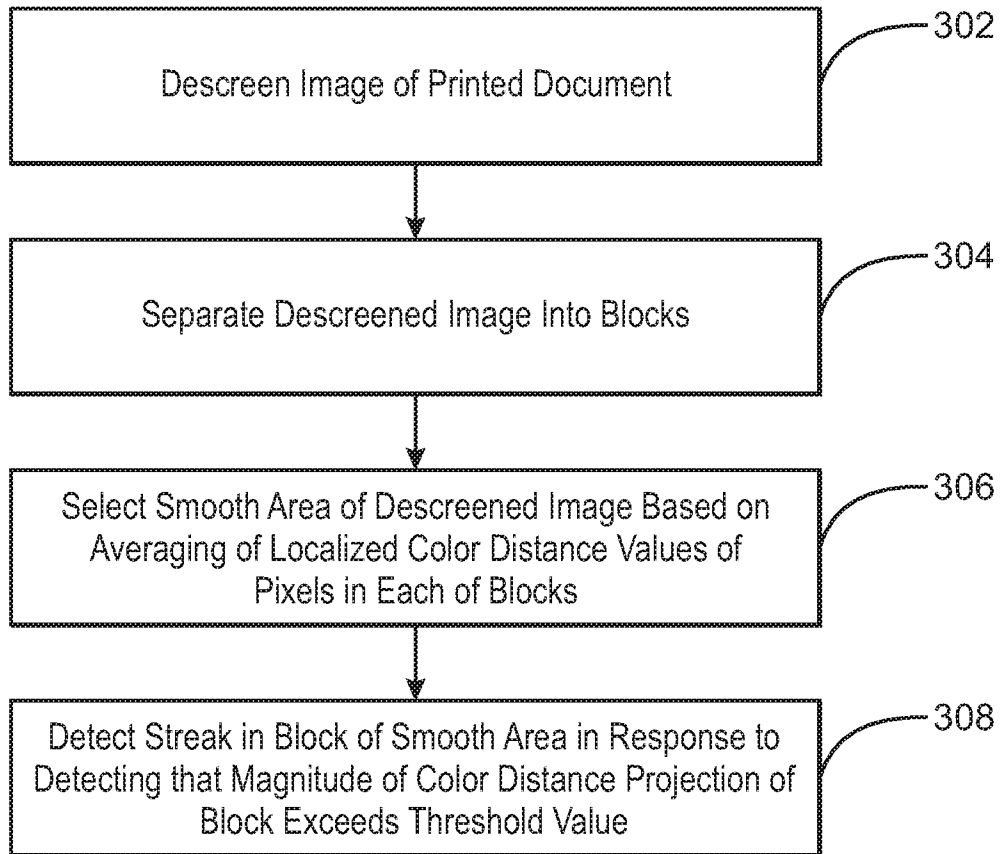


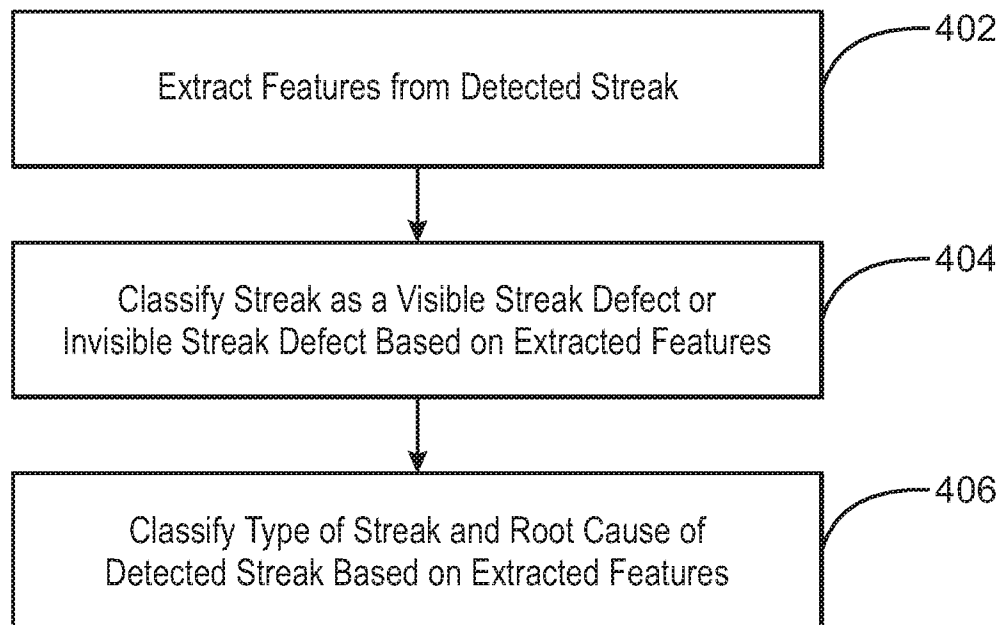
FIG. 2

3/8



300
FIG. 3

4/8



400
FIG. 4

5/8

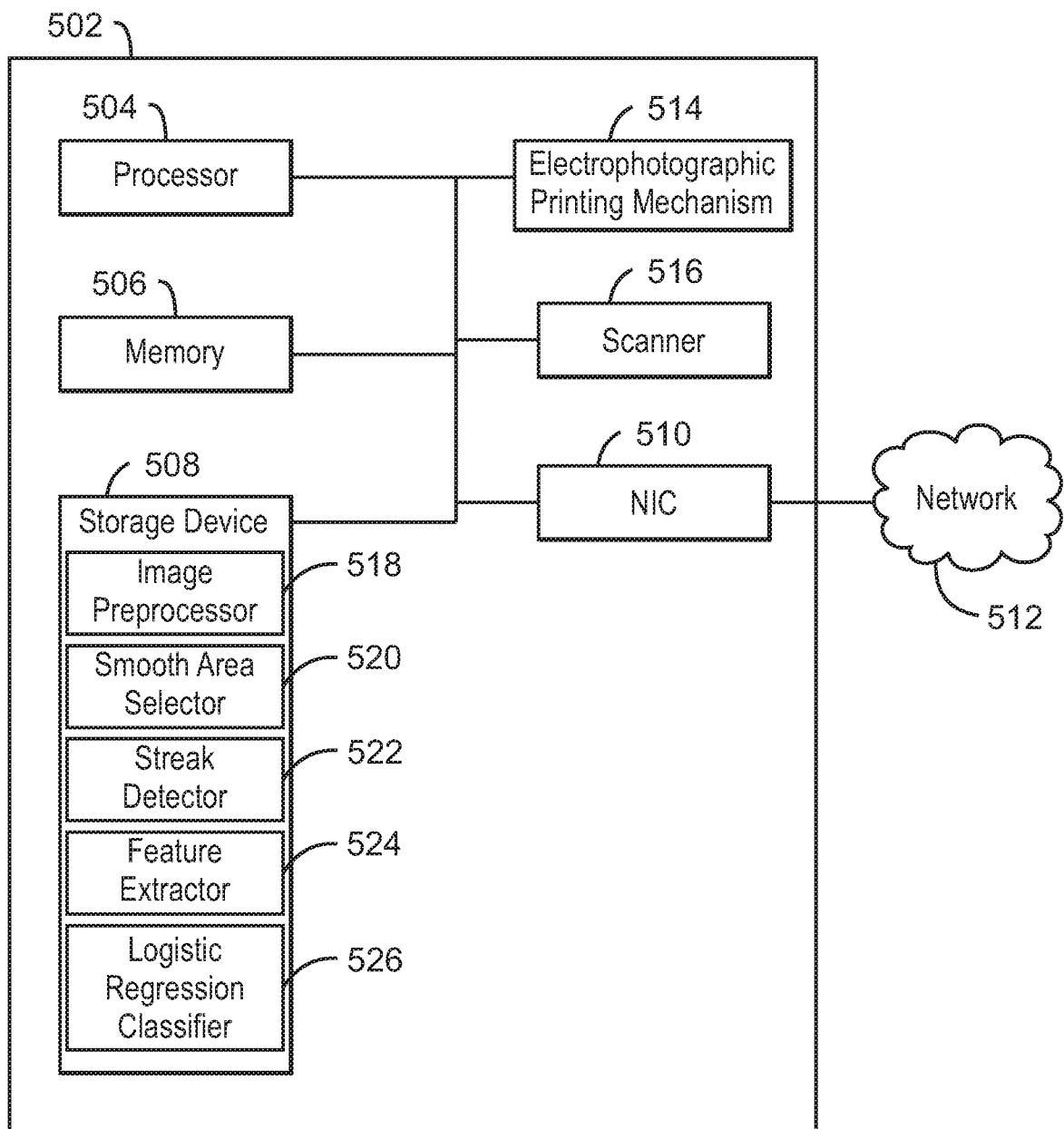


FIG. 5A

6/8

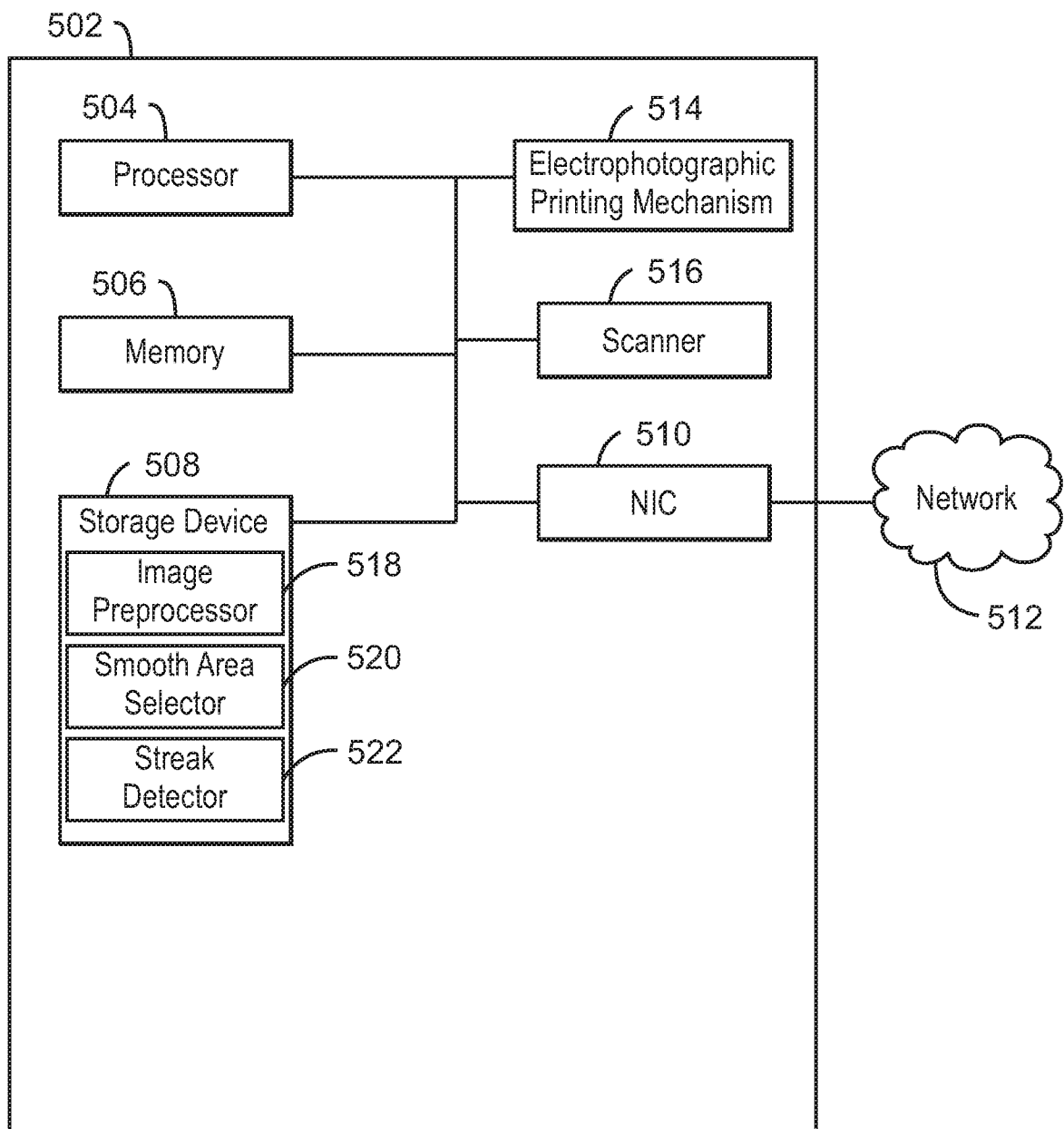


FIG. 5B

7/8

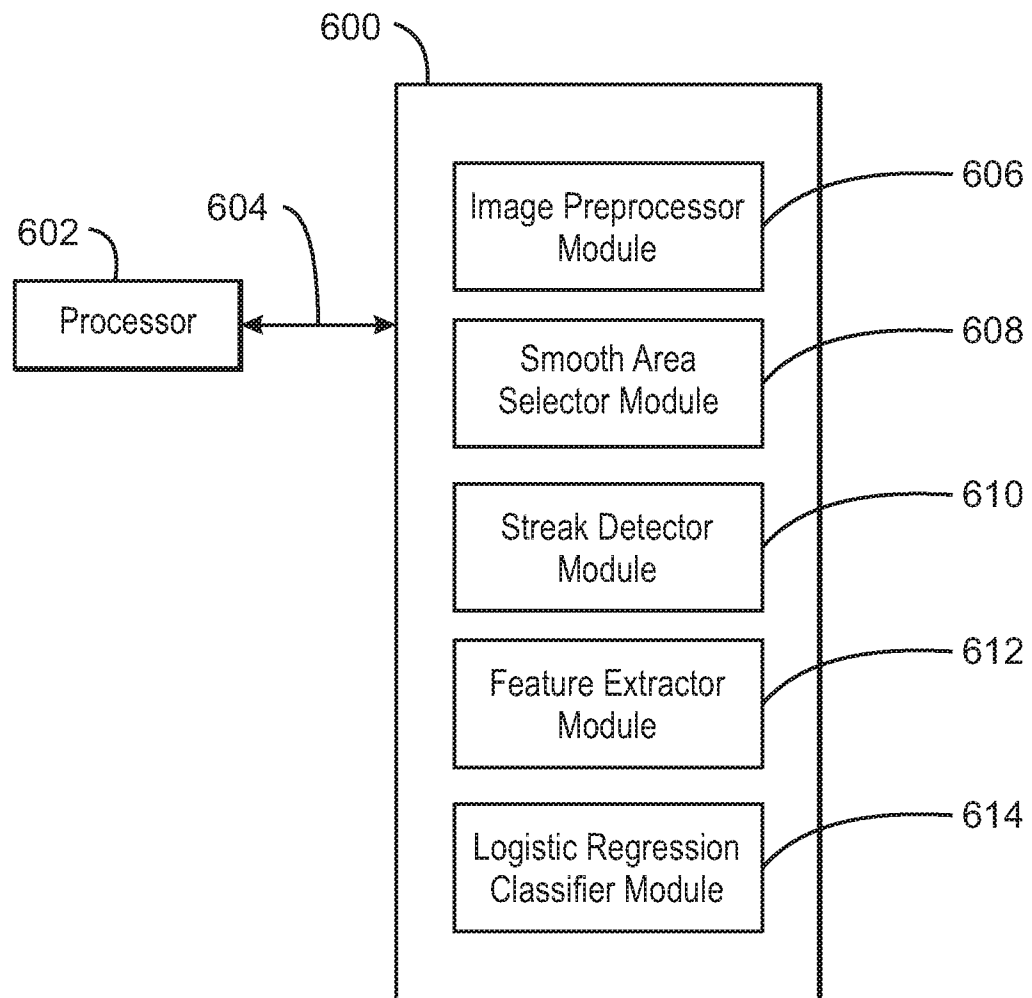


FIG. 6A

8/8

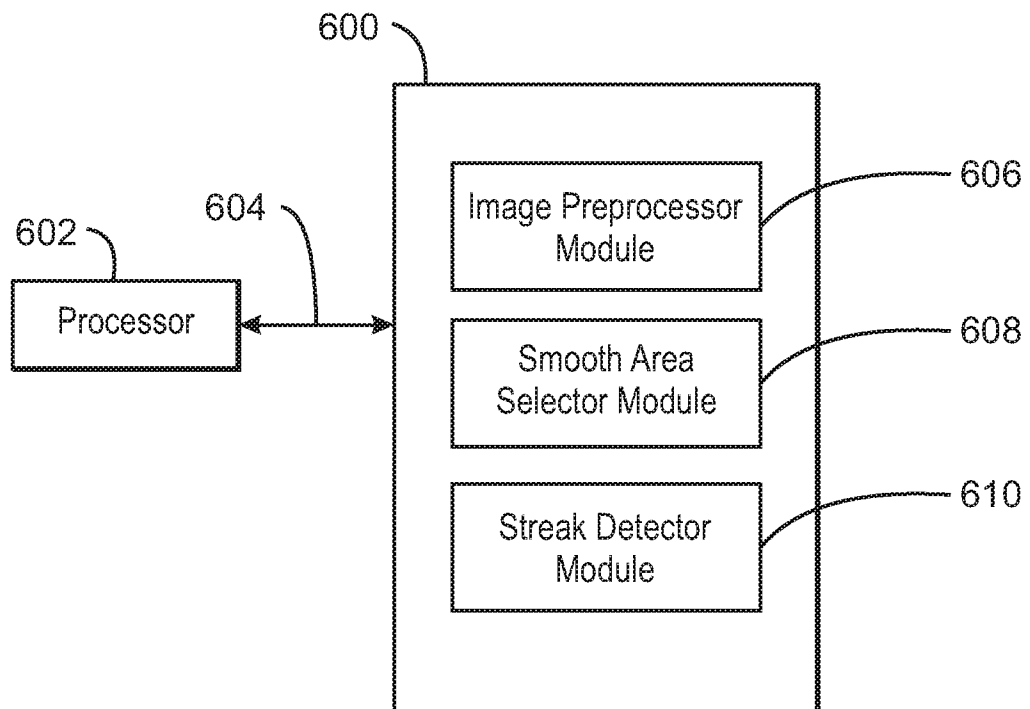


FIG. 6B

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US 2019/013233

A. CLASSIFICATION OF SUBJECT MATTER		
G06K9/03 (2006.01) H04N 1/60 (2006.01) 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)		
H04N1/00-1/62, G06K9/00-9/46, B41J29/00-29/393, B41F33/00		
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)		
PatSearch (RUPTO internal), USPTO, PAJ, Esp@cenet, DWPI, EAPATIS, PATENTSCOPE		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X Y	US 2017/0274690 A1 (FUJIFILM CORPORATION) 28.09.2017, [0064]-[0083], [0098], [0106], [0116]-[0121]	1, 7-8, 10-11 2-6, 9, 12-15
Y	US 2018/0082115 A1 (HP INDIGO B.V.) 22.03.2018, [0013]-[0016], [0020]-[0036]	2-5, 13-15
Y	US 2017/0144464 A1 (FUJIFILM CORPORATION) 25.05.2017, [0044]-[0046], [0057]	9, 12
Y	WO 2015/147151 A1 (FUJIFILM CORPORATION) 01.10.2015, [0247]	6
A	US 2012/0033238 A1 (PALO ALTO RESEARCH CENTER INCORPORATED) 09.02.2012	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 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 combined with one or more other such documents, such combination being obvious to a person skilled in the art “&” document member of the same patent family		
Date of the actual completion of the international search		Date of mailing of the international search report
03 September 2019 (03.09.2019)		12 September 2019 (12.09.2019)
Name and mailing address of the ISA/RU: Federal Institute of Industrial Property, Berezhkovskaya nab., 30-1, Moscow, G-59, GSP-3, Russia, 125993 Facsimile No: (8-495) 531-63-18, (8-499) 243-33-37		Authorized officer V. Voropai Telephone No. 8 499 240 25 91