(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
10 November 2016 (10.11.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/178807 A1

(51) International Patent Classification:
G06T 11/00 (2006.01)

(21) International Application Number:
PCT/US2016/027569

(22) International Filing Date:
14 April 2016 (14.04.2016)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
14/704,957 5 May 2015 (05.05.2015) US

(71) Applicant: **GOOGLE INC.** [US/US]; 1600 Amphitheatre Parkway, Mountain View, CA 94043 (US).

(72) Inventor: **NEULANDER, Ivan**; c/o Google Inc., 1600 Amphitheatre Parkway, Mountain View, CA 94043 (US).

(74) Agents: **GARNER, Casey** et al.; Patent Law Works LLP, 201 S. Main Street, Suite 250, Salt Lake City, UT 84111 (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, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, 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.

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

Published:

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

WO 2016/178807 A1

(54) Title: PAINTERLY PICTURE GENERATION

(57) Abstract: A system for generating painterly pictures comprises a processor and a memory storing instructions that, when executed, cause the system to receive an input image; add a base layer of brush strokes based on a sampling of underlying pixel values in the input image, the base layer comprising coarse large brush strokes that are well-distributed in a uniform pattern; add a uniform layer of brush strokes iteratively based on a sampling of underlying pixel values in the input image and directionally based on a gradient of the input image; and add a detail layer of brush strokes based on a determined saliency map. The disclosure also includes similar methods and computer program products for generating painterly pictures.

PAINTERLY PICTURE GENERATION

TECHNICAL FIELD

[0001] The present disclosure relates to computer generated images. More specifically, the present disclosure relates to computer generated painterly pictures.

BACKGROUND

[0002] Computer applications are sometimes used to create images in a variety of endeavors such as film, video games, simulators and art. Computer applications can generate still images or video and may do so in two dimensions (2D) and three dimensions (3D). While some systems create such computer imagery based on data structures defining a virtual word, others transform existing images into new images.

[0003] Some systems transform photographic images into painterly pictures that resemble artistically rendered paintings. With the popularity of smart phones with cameras and an ever increasing use of the Internet, social networks and other media sharing systems, there is an increased desire for photographic manipulation and transformation including transforming photographic images into painterly pictures.

[0004] While generating aesthetically pleasing paintings is an age-old artistic struggle, creating painterly pictures based on photographic images using a computer system is a much more modern endeavor. Generating aesthetically pleasing computer generated painterly pictures is difficult due to the inherent artistic and philosophical underpinnings that define what is aesthetically pleasing. Moreover, the inherent processing limits of computers in general, and smart phones in particular (or other limited processing power devices), make generating aesthetically pleasing computer generated painterly pictures in an efficient and timely manner all the more difficult.

SUMMARY

[0005] The present disclosure relates to systems and methods for automatically generating painterly pictures. According to one innovative aspect of the subject matter in this disclosure, a system having one or more processors and a memory storing instructions that, when executed, cause the system to: receive an input image; add a base layer of brush strokes based on a sampling of underlying pixel values in the input image, the base layer comprising coarse large brush strokes that are well-distributed in a uniform pattern; add a uniform layer

of brush strokes iteratively based on a sampling of underlying pixel values in the input image and directionally based on a gradient of the input image; and add a detail layer of brush strokes based on a determined saliency map.

[0006] In general, another innovative aspect of the subject matter described in this disclosure may be implemented in methods that include receiving, using one or more processors, an input image; adding, using one or more processors, a base layer of brush strokes based on a sampling of underlying pixel values in the input image, the base layer comprising coarse large brush strokes that are well-distributed in a uniform pattern; adding, using one or more processors, a uniform layer of brush strokes iteratively based on a sampling of underlying pixel values in the input image and directionally based on a gradient of the input image; and adding, using one or more processors, a detail layer of brush strokes based on a determined saliency map.

[0007] Other implementations of one or more of these aspects include corresponding systems, apparatus, and computer programs, configured to perform the actions of the methods, encoded on computer storage devices.

[0008] These and other implementations may each optionally include one or more of the following features. For example, features may include the saliency map being based on areas of facial recognition within the input image. The saliency map may be based on a gradient image of the input image. Placement of the uniform layer of brush strokes may be based on a random or quasi-random algorithm. Brush stroke size in a subsequent iteration may be about half of a brush stroke size of a previous iteration of the uniform layer. In some implementations, sizes and outlines of the uniform layer of brush strokes may be jittered using a multi-dimensional low-discrepancy sequence. In other implementations, sizes and outlines of the uniform layer of brush strokes may be jittered using a low-discrepancy halton sequence. In yet other implementations, sizes and outlines of the uniform layer of brush strokes may be jittered using a low-discrepancy Hammersley sequence. The brush strokes may be added to priority queue in order to sort them by size. The brush strokes in the priority queue may be placed on a canvas. The brush strokes of the base layer may substantially cover the canvas.

[0009] These implementations are particularly advantageous in a number of respects. For instance, a uniform layer with a non-uniform detail layer based on image gradients and facial detection provides a visually pleasing painterly rendering. Moreover, the technology allows for efficient rendering on devices with limited processing power. Further, regions between high and low detail transition smoothly from detailed to uniform brush strokes in a

natural way. Thus, the technology provides improved image manipulation while consuming less computational processing resources while resulting in a more aesthetically pleasing resultant image.

[0010] It should be understood, however, that the above features and advantages are not all-inclusive and many additional features and advantages are contemplated and fall within the scope of the present disclosure. Moreover, it should be understood that the language used in the present disclosure has been principally selected for readability and instructional purposes, and not to limit the scope of the subject matter disclosed herein.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] The present disclosure is illustrated by way of example, and not by way of limitation in the figures of the accompanying drawings in which like reference numerals are used to refer to similar elements.

[0012] Figure 1 is a flow chart of an example method for generating painterly pictures.

[0013] Figure 2 is a flow chart of an example method for adding a base layer brush strokes.

[0014] Figure 3 is a flow chart of an example method for adding uniform layers of brush strokes.

[0015] Figure 4 is a flow chart of an example method for adding detail layers of brush strokes.

[0016] Figure 5 is a graphic representations of an example input image.

[0017] Figure 6 is a high-level block diagram illustrating an example system for generating painterly pictures.

[0018] Figure 7 is a block diagram illustrating an example system including a painterly rendering module.

[0019] Figure 8 is a block diagram illustrating an example painterly rendering module.

DETAILED DESCRIPTION

[0020] Systems, methods and interfaces for generating painterly pictures are disclosed. While the systems, methods and interfaces of the present disclosure are now described in the context of client-server system, it should be understood that the systems, methods and interfaces can be applied to systems other than a client-server system operably

connected over a network where painterly rendering is performed on a single server. For example, a client device, third party servers, social network servers, or media sharing servers may provide some or all of the functionality described herein and may employ a cluster of servers in order to provide such functionality. As additional examples, client hardware may be a mobile phone or tablet device.

[0021] In general, the systems, methods and interfaces for generating painterly pictures layers paint strokes using three combined strategies: 1) base layer; 2) uniform placement; 3) detail placement. The base layer largely covers the canvas with a coarse, uniform layer of large, essentially round, and nearly opaque paint strokes. The uniform placement generates varying numbers of evenly spaced semitransparent paint strokes that uniformly cover the canvas. The detail placement may leverage face detection signals and gradient data in conjunction with image-based importance sampling in order to vary the size and placement of paint strokes so as to provide finer detail near faces, producing an output of higher fidelity to the photographic input.

[0022] These strategies capture the general color scheme and large-scale features of the input using a small number of strokes, and prevent the canvas from showing through any gaps left by the subsequent steps. The canvas may be a digital representation of a painter's canvas where the painterly may place brush strokes. Thus, in some instances, a blank canvas may be embodied in a multidimensional array where each array item represents a pixel with a color value and opaqueness. In one example, the blank canvas may be set such that all the pixels have 0% opaqueness. In another example, the blank canvas may be set such that all of the pixels have 100% opaqueness and are set to an off-white cotton or linen color.

[0023] Figure 1 is a flow chart of an example method 100 for generating painterly pictures. The method 100 begins by receiving 102 an input image. Generally, the input image is a photographic image which a user wishes to transform into a painterly picture. More generally however, the input image may be any type of digital image. For example, the input image may be a file formatted as a bitmap (BMP), joint photographic experts group (JPEG), or a portable networks graphic (PNG) file. In one example, the communication module 802 discussed in further detail with reference to Figure 8 below may receive the input image, check that the image is properly formatted, and transform the input image to data structures defined by the painterly rendering module 604 discussed in further detail with reference to the figures below.

[0024] The method 100 may then add 104 base layer brush strokes to a priority queue in order to sort the brush strokes by size. In one example, the base layer module 816

described with reference to Figure 8 adds the base layer brush strokes to the priority queue. The base layer brush strokes are generated based on a sampling of underlying pixels values in the input image. As discussed further in Figure 2 and Figures 5-8, the base layer module 816 may sample the input image for characteristics including the color scheme of the input image. The brush strokes of the base layer are generally coarse large brush strokes that are well-distributed in a uniform pattern. Moreover, the brush strokes of the base layer may be substantially opaque and may substantially or completely cover the canvas on an output image. As an example, the brush strokes of the base layer may be at least 98% opaque on a scale where 100% is completely opaque and 0% is completely transparent. In other examples, the brush strokes of the base layer may cover at least 95% opaque, or at least 90% opaque.

[0025] The brush module 806 may be invoked by the uniform layer module 818 to determine 106 a brush size for a uniform layer of brush strokes to be applied to the input image after the base layer brush strokes are applied. The uniform layer module 818 may make several iterative passes over the input image. In some instances, each subsequent pass over the input image reduces the brush stroke size by about half until a minimum brush size threshold is reached for the uniform layer. In one example, the minimum brush size threshold may be a 20/0 brush. In other instances, the minimum brush size threshold may be as fine as a 30/0 or even a single pixel brush. In some cases, the minimum brush size threshold may be a user definable parameter entered through the user interface module 822. Thus as illustrated in the exemplary block 108, the method 100 passes over the input image and adds brush strokes to the priority queue with the determined brush size. The method may then determine 110 whether to make another pass over the input image.

[0026] The detail layer module 820 adds 112 a detail layer of brush strokes to the priority queue. In some instances, the detail layer brush strokes are based on a determined saliency map. Additionally, similar to the uniform layer of brush strokes, the detail layer brush strokes may be based on a sampling of underlying pixel values in the input image and directionally based on a gradient of the input image. While the method 100 does not depict an iterative loop over adding 112 detail brush strokes to the priority queue, in some instances, the method 100 iterates over the input image to create the detail layer of brush strokes. Moreover, in some instances, the iterative placement of detailed brush strokes is driven by a saliency map in which iterative passes to the input image create increasingly smaller brush strokes. Further details regarding adding 112 detail brush strokes to the priority queue are discussed below with reference to Figure 4. In some instances, the detail brush strokes are

generated using an image-based importance sample, which randomly selects pixels according to the probability distribution defined by the saliency map. It takes as inputs a) the saliency map and b) a pair of random or quasi-random numbers, and produces a pair of pixel coordinates plus a size scalar. These serve, respectively, as the center and nominal length (the latter optionally clamped to a smaller value) of the detail brush stroke being generated.

[0027] The brush strokes for the base layer, the uniform layer and the detail layer may be added to a priority queue as described above. Thus, the canvas module 808 may place 114 the brush strokes onto the canvas by iterating through the priority queue. In some cases, the priority queue may be sorted by brush stroke sizes prior to placing the brush strokes on the canvas. In alternatively implementations, brush strokes may be placed directly on the canvas as they are determined rather than adding the brush strokes to a queue prior to placing the brush strokes on the canvas. In some implementations, the priority queue itself serves as a sorting mechanism, and when used orders the brush strokes from larges to smallest.

[0028] Once all of the brush strokes are placed 114 onto the canvas, the method may present an output image to the user. In some cases the output image may be displayed on a screen within a web browser, rendered by an image processing application executed on a mobile computing device, or may be made available as a file download in a predetermined digital image file format.

[0029] Figure 2 is a flow chart of an example method 104 for adding a base layer brush strokes. In particular the method 104 is further detail of the block 104 shown in Figure 1 where the method 100 adds a base layer of brush strokes to the priority queue. For example, the method 104 determines 202 base layer brush properties. This determination of base layer brush properties may be performed by the base layer module 816 in conjunction with the brush module 806. In some instances, the base layer brush properties may be based on the size of the input image, the variation of the underlying color scheme of the input image, the size of the output image, etc. The user interface (UI) module 822 may provide a user interface that allows a user of the painterly rendering module 604 to set or refine values for the base layer brush properties.

[0030] The base layer module 816 may then sample 204 the input image for a color scheme and add 206 brush strokes with the determined brush properties. Moreover, the added brush strokes are based on the underlying input image color scheme. The brush strokes of the base layer may generally be large brush strokes that substantially cover the canvas. As an example, the brush strokes of the base layer may cover at least 98% of the canvas. In other examples, the brush strokes of the base layer may cover at least 95% of the canvas, or at least

90% of the canvas. Examples of brush sizes include from smallest to largest, 20/0, 12/0, 10/0, 7/0, 6/0, 5/0, 4/0 (also written as 0000), 000, 00, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 18, 20, 22, 24, 25, 26, 28, 30. In some instances, iterative passes of the uniform layer and detail layer of brush strokes may iteratively use the next brush size as outlined above (e.g. use the next smallest brush size; an 18 brush and then a 16 brush).

[0031] Figure 3 is a flow chart of an example method 300 for adding uniform layers of brush strokes to the priority queue. The method 300 includes determining 106 a brush size as discussed above with reference to Figure 1 and iteratively passing over the input image by determining 110 whether to make any further passes. In certain non-limiting instances, brush stroke size in each subsequent iteration may be smaller (e.g., about half of the brush stroke size) than the brush stroke size of each previous iteration of the uniform layer. Additionally, the method 300 further details adding brush strokes to the uniform layer as discussed in block 108 discussed with reference to Figure 1. Thus blocks 302, 304, 306, 308, and 310 in Figure 3 are further details of the block 108.

[0032] In order to form a uniform layer of brush strokes, the method 300 identifies 302 placement points and brush stroke jitter. Placement of the uniform layer of brush strokes may be based on a random or quasi-random algorithm by employing the randomize module 810.

[0033] The color of brush strokes is determined 304 with the sampling module 804. In some implementations, the sampling module 804 may make the color determination by combining (according to some weighting scheme) the colors of the input pixels that fall within that stroke. In other implementations, the color determination may be based on the input image, brush size and placement point. In one instance, the underlying pixels of the input image are normalized for color and a color is determined based on whether a majority or a plurality pixels have a color. In some instances, a threshold number of normalized pixels need to have a common color otherwise a run-off process occurs where the colors of the underlying pixels are further normalized. This process of re-normalization may repeat until the color of the paint stroke is determined by meeting the threshold number of pixels with a common color.

[0034] The gradient of the brush strokes is determined 306 with the gradient module 814. The gradient module 814 may generate a gradient image based on the input image that indicates gradients at various points in the input image. The directionality of the brush strokes may be based on the gradient image determined by the gradient module 814. Thus in some

instances, the gradient module 814 may take into account the brush size and placement point of the brush stroke and return a directionality and amplitude for the brush stroke.

[0035] The brush module 806 and/or the randomize module 810 may jitter 308 brush strokes sizes and outlines. By jittering the brush strokes sizes and outlines, the painterly picture may take on a more realistic feel where an artist generally varies each brush stroke slightly. In one non-limiting example, sizes and outlines of the uniform layer of brush strokes may be jittered using a low-discrepancy halton sequence.

[0036] The canvas module 808 adds 310 the uniform layer of brush strokes to the priority queue. The brush strokes are generated based on brush size, placement point, color, gradient and jitter. Further details regarding the uniform layer are discussed in particular with reference to Figures 5 and 8 below.

[0037] Figure 4 is a flow chart of an example method 112 for adding detail layer brush strokes to the priority queue. In particular, the method 112 is further detail of the block 112 shown in Figure 1 where the method 100 adds a detail brush strokes to the priority queue. For example, the method 112 identifies 402 predefined objects (e.g., human faces and/or non-human animal faces) in the input image and computes 404 a gradient image based on the input image. The detail layer module 820 and/or the gradient module 814 and/or the saliency module 812 may identify faces in the input image and compute the gradient image.

[0038] The saliency module 812 may create 406 a saliency map based on the identified faces and gradient image. The saliency map may be a heat-type map which may show where to place detailed brush strokes. In some instances, the saliency map may indicate that detailed brush strokes may be placed in the areas of the input image where predefined objects (e.g., faces) are identified. The detailed brush strokes are added 408 to the priority queue based on the saliency map. In some cases, the detailed brush strokes may be iteratively added similar to the brush strokes in the uniform layer. However, in general the detailed brush strokes are not uniformly placed. Rather, the detailed brush strokes are placed based on the saliency map. Finer brush strokes may be reserved for areas of higher saliency.

[0039] Figure 5 is a graphic representations of an example black and white input image 500. The image 500 depicted in Figure 5 is a rather simple picture that allows discussion of some of the features of the technology without added distraction. Moreover, the simplicity of the image 500 of Figure 5 facilitates clearer photocopying on many modern photocopy machines and thus allows duplication of this disclosure as necessary. Nevertheless, the principles discussed herein are applicable to photographic images as well as non-photographic images.

[0040] The input image 500 contains a number of faces 502a-c that may be recognized by the saliency module 812 and/or the detail layer module 820. Thus, in processing the input image 500 with the technology, a painterly picture may be rendered that has greater detail in and around the faces 502a-c.

[0041] The input image 500 also contains at least one tree 506 and one mountain 504. As depicted in the input image 500, the shaded and unshaded areas of the mountain 504 create a stark contrast of black and white, while the tree 506 has many leaves and branches of black and white that tend to be clustered together forming an area of high detail. Thus the mountain 504 and the tree 506 may be processed by the gradient module 814 differently. Moreover, the gradient module 814 may handle the whitespace of the input image 500 significantly different than either the mountain 504 or the tree 506. In some instances, the whitespaces of the input image 500 may be painted with brush strokes that are largely uniform (e.g. uniform layer of brush strokes) while the mountain 504 for instance may be rendered with brush strokes that are directionally placed so that brush stroke sweeps are in the direction of an elongated shadow. Additionally, the tree 506 may be processed as an area of high saliency by the gradient module 814 and/or the saliency module 812.

[0042] The combination of the base layer, the uniform layer and the detail layer in the technology may render well-stratified uniform placement of brush stroke coverage over the full canvas with non-uniform placement of brush stroke coverage restricted to regions of high detail. As discussed the non-uniform placement may generally be guided by the saliency information produced by the saliency module 812. The saliency module 812 may use facial detection on the input image in conjunction with gradient information provided by the gradient module 814. The resultant painterly picture incorporates a high density of fine brush strokes into regions of high saliency along with coarser and more stylized brush strokes elsewhere on the canvas. The technology thereby provides an elegant and efficient solution that runs rapidly on a variety of devices and underlying platforms. In some cases, the technology may generate painterly pictures in near real-time (i.e., without intended delay) and, in some cases, may be used in a variety of near real-time applications (e.g. video chat, etc.).

[0043] The technology may use the same style of importance sampling for populating detailed regions with non-uniform brush strokes. Additionally uniformly placed brush strokes may be independently generated using a low-discrepancy sequence to produce high stratification with less processing power over the less detailed areas. In some instances, the output painterly picture is rendered by compositing together, in order of decreasing size,

numerous simulated brush strokes of semitransparent color, whose sizes and orientations are based on the details of the input image. In some instances, brush strokes may incorporate smooth variation of color and opacity and may be slightly randomized. The level of randomization may be adjusted using the UI module 822.

[0044] In some non-limiting instances, the base layer quickly covers the canvas with a coarse, uniform layer of large, essentially round, and nearly opaque brush strokes without incurring significant computational costs. The base layer brush strokes capture the general color scheme and large-scale features of the input image using a small number of strokes, and may prevent the canvas from showing through any gaps left by subsequent layers of brush strokes.

[0045] The uniform layer includes varying numbers of evenly spaced semitransparent brush strokes that uniformly cover the canvas. In some instances, the sizes of the brush strokes may vary over a broad range, typically from a few pixels up to the size of a base layer brush stroke. In some instances, the uniform layer is applied in multiple passes. In some non-limiting instances, each pass during the uniform layer may use half the brush size of the previous pass. In some cases, brush sizes may include minor randomization or pseudo-randomization. For example, a low-discrepancy halton sequence may be used to generate the placement point of the brush strokes and to jitter their relative sizes and outlines. Moreover, a halton sequence may allow the brush strokes to be well-stratified. Each brush stroke is colored based on the input pixels it covers, and may be elongated and oriented based on gradient vectors over the pixels it covers.

[0046] The gradient image may be computed with a Gaussian blur and a gamma correction of the vector norms. Blurring the gradient vector image may generally spread the influence of edges into surrounding pixels. In this way, the technology may simulate the way a physical paint brush diffuses paint along sharp edges. In some instances, gamma correction may be used to yield more uniform brush stroke lengths.

[0047] The detail layer leverages face detection signals and gradient data in conjunction with image based importance sampling in order to vary the size and placement of brush strokes so as to provide finer detail near faces. The resultant painterly picture may have higher detail fidelity to the photographic input.

[0048] The saliency map may be created such that the saliency map is proportional at each pixel to the per-pixel vector norm of the gradient image. In some instance, facial detection may identify a tight bounding rectangle for each face detected. The tight bounding rectangle may be slightly enlarged and a radial falloff function may be applied to the edge of

the bounding rectangle so that that the rectangle interior becomes an ellipse bounded by the rectangle, and falling off toward the edges. The pixel values within the ellipse may serve to scale saliency pixels that were derived from the gradient vector norms, thereby producing a natural shape and falloff for the facial saliency values.

[0049] Figure 6 is a high-level block diagram illustrating an example system 600 for generating painterly pictures. The illustrated system 600 includes client devices 620a-620n (also referred to herein individually and collectively as 620) that are accessed by users 624a-624n (also referred to herein individually and collectively as 624), a server 602 having a painterly rendering module 604.

[0050] The client devices 620a-620n in Figure 6 are used by way of example. While Figure 6 illustrates two client devices 620a and 620n, the present disclosure applies to any system architecture having one or more client devices 620. Furthermore, while only one network 610 is coupled to the client devices 620a-620n, and the server 602, in practice one or more networks 604 can be connected to these entities.

[0051] While shown as operational on the server 602 in Figure 6, in some implementations all or part of the painterly rendering module 604 may be operational on the one or more of the client devices 620, server 604 or other servers. The painterly rendering module 604 is also coupled for communication with the client device 620a, which is connected to the network 610. The user 624a interacts with the client device 620a. Similarly, the client device 620n is coupled to the network 610 and the user 624n interacts with the client device 624n. The user 624 is a human user of the client device 620. It should be recognized that the painterly rendering module 604 can be stored in any combination of the devices and servers, or in only one of the devices or servers. The painterly rendering module 604 will be describe in more detail with reference to Figures 7 and 8.

[0052] The client devices 620 can be any computing device including one or more memory and one or more processors, for example, a laptop computer, a desktop computer, a tablet computer, a mobile telephone, a smart phone device, a portable game player, a portable music player, a television with one or more processors embedded therein or coupled thereto or any other electronic device capable of accessing a network. In some implementations, the system 600 includes a combination of different types of client devices 620. For example, a combination of a personal computer and a mobile phone. As will be described below, it should be understood that the present technologies can operate on different models other than

a client-server architecture. For example, the client devices 620 may include the painterly rendering module 604 and include different services.

[0053] The browser 622 can be a web browser stored on the on the client device 620 and configured for two-way communications with the server 604. For example, the browser 622 is a software application for retrieving, presenting and traversing information resources on the web.

[0054] The network 610 enables communications between the client devices 620a-n, the server 602. Thus, the network 610 can include links using technologies including, for example, Wi-Fi, Wi-Max, 2G, Universal Mobile Telecommunications System (UMTS), 3G, Ethernet, 802.11, integrated services digital network (ISDN), digital subscriber line (DSL), asynchronous transfer mode (ATM), InfiniBand, Peripheral Component Interconnect Express (PCIe) Advanced Switching, etc. Similarly, the networking protocols used on the network 610 can include the transmission control protocol/Internet protocol (TCP/IP), multi-protocol label switching (MPLS), the User Datagram Protocol (UDP), the hypertext transfer protocol (HTTP), the simple mail transfer protocol (SMTP), the file transfer protocol (FTP), lightweight directory access protocol (LDAP), Code Division Multiple Access (CDMA), Wideband Code Division Multiple Access (WCDMA), Global System for Mobile communications (GSM), High-Speed Downlink Packet Access (HSDPA), etc. The data exchanged over the network 610 can be represented using technologies and/or formats including the hypertext markup language (HTML), the extensible markup language (XML), etc. In addition, all or some of links can be encrypted using encryption technologies, for example, the secure sockets layer (SSL), HTTP Secure (HTTPS) and/or virtual private networks (VPNs) or Internet Protocol security (IPsec). In some implementations, the entities can use custom and/or dedicated data communications technologies instead of, or in addition to, the ones described above. Depending upon the implementation, the network 610 can also include links to other networks.

[0055] In some implementations, the network 610 is a partially public or a wholly public network, for example, the Internet. The network 610 can also be a private network or include one or more distinct or logical private networks (e.g., virtual private networks, Wide Area Networks ("WAN") and/or Local Area Networks ("LAN")). Additionally, the communication links to and from the network 610 can be wired or wireless (i.e., terrestrial or satellite-based transceivers). In some implementations, the network 610 is an Internet Protocol (IP) based wide or metropolitan area network.

[0056] The network 610 may have any number of configurations including a star configuration, token ring configuration or other configurations. Furthermore, the network 610 may include a local area network (LAN), a wide area network (WAN) (e.g., the Internet), and/or any other interconnected data path across which multiple devices may communicate. In some implementations, the network 610 may be a peer-to-peer network. The network 610 may also be coupled to or include portions of a telecommunications network for sending data in a variety of different communication protocols. In some implementations, the network 610 includes short-wavelength communication networks or a cellular communications network for sending and receiving data including via short messaging service (SMS), multimedia messaging service (MMS), hypertext transfer protocol (HTTP), direct data connection, wireless application protocol (WAP), electronic messages, etc.

[0057] Figure 7 is a block diagram illustrating an example system 602 including a painterly rendering module. In the illustrated implementations, the server 602 includes the network interface (I/F) module 702, a processor 704, a memory 706, and a storage device 708. These components of the social network server 602 are communicatively coupled to a bus or software communication mechanism 720 for communication with each other.

[0058] The network interface module 702 is coupled to the network 610 by signal line 612. The network interface module 702 is also coupled to the bus 720. The network interface module 702 includes ports for wired connectivity including but not limited to universal serial bus (USB), SD, or Category 6a cable (CAT-6a as defined in the American national Standards Institute's and Telecommunication's Industry Association's ANSI/TIA-568-C.1 standard), etc. The network interface module 702 links the processor 704 to the network 610 that may in turn be coupled to other processing systems. The network interface module 702 provides other connections to the network 610 using network protocols, e.g., TCP/IP, HTTP, HTTPS and SMTP. In some implementations, the network interface module 702 includes a transceiver for sending and receiving signals using WiFi, Bluetooth® or cellular communications for wireless communication.

[0059] The processor 704 may include an arithmetic logic unit, a microprocessor, a general purpose controller or some other processor array to perform computations and provide electronic display signals to a display device. In some implementations, the processor 704 is a hardware processor. The processor 704 is coupled to the bus 720 for communication with the other components. Processor 704 processes data signals and may include various computing architectures including a complex instruction set computer (CISC) architecture, a

reduced instruction set computer (RISC) architecture, or an architecture implementing a combination of instruction sets. Although only a single processor is shown in Figure 7, multiple processors may be included. It should be understood that other processors, operating systems, sensors, displays and physical configurations are possible.

[0060] The memory 706 stores instructions and/or data that may be executed by the processor 704. In the illustrated implementation, the memory 706 stores the painterly rendering module 604 and the publication module 712. The memory 706 is coupled to the bus 720 for communication with the other components of the server 602. The instructions and/or data may include code for performing any and/or all of the techniques described herein. The memory 706 may be a dynamic random access memory (DRAM) device, a static random access memory (SRAM) device, flash memory or some other memory devices.

[0061] The storage device 708 may be a non-transitory memory that stores data. For example, the storage device 708 may be a dynamic random access memory (DRAM) device, a static random access memory (SRAM) device, flash memory or some other memory device. In some implementations, the storage device 708 also includes a non-volatile memory or similar permanent storage device and media, for example, a hard disk drive, a floppy disk drive, a compact disc read only memory (CD-ROM) device, a digital versatile disc read only memory (DVD-ROM) device, a digital versatile disc random access memories (DVD-RAM) device, a digital versatile disc rewritable (DVD-RW) device, a flash memory device, or some other non-volatile storage device. In some implementations, the storage device 708 stores data received from and sent to users 624 to accomplish the functionalities described herein of the painterly rendering module 604.

[0062] Software communication mechanism 720 may be an object bus (e.g., Common Object Request Broker Architecture (CORBA)), direct socket communication (e.g., TCP/IP sockets) among software modules, remote procedure calls, UDP broadcasts and receipts, HTTP connections, function or procedure calls, etc. Further, any or all of the communication could be secure (secure shell (SSH), HTTPS, etc.). The software communication mechanism 720 can be implemented on any underlying hardware, for example, a network, the Internet, a bus, a combination thereof, etc.

[0063] The publication module 712 can be software or routines for publishing media. In Figure 10, the publication module 712 is shown as a module operational a part of memory 706. In some implementations, the publication module 712 may be part of a social network

application. In some implementations, the publication module 712 may be adapted for uploading and providing media and social network metadata to the social network application. In some implementations, the publication module 712 may make media available to client devices 620 connected to the network 610. In some instances, the publication module operates on painterly picture renderings provided by the painterly rendering module 604.

[0064] The painterly rendering module 604 is configured to perform a number of functions including, but not limited to: receiving, using one or more processors, an input image; adding, using one or more processors, a base layer of brush strokes based on a sampling of underlying pixel values in the input image, the base layer comprising coarse large brush strokes that are well-distributed in a uniform pattern; adding, using one or more processors, a uniform layer of brush strokes iteratively based on a sampling of underlying pixel values in the input image and directionally based on a gradient of the input image; and adding, using one or more processors, a detail layer of brush strokes based on a determined saliency map. The painterly rendering module 604 includes a number of components as will be described in more detail below with reference to Figure 8. These components cooperate to provide the functions described above and described in detail herein

[0065] Figure 8 is a block diagram illustrating an example painterly rendering module 604. The painterly rendering module 604 includes a communication module 802, a sample module 804, a brush module 806, a canvas module 808, a randomization module 810, a saliency module 812, a gradient module 814, a base layer module 816, a uniform layer module 818, a detail layer module 820, and a UI module 822.

[0066] The communication module 802 can be software or routines for handling communications between the sample module 804, the brush module 806, the canvas module 808, the randomization module 810, the saliency module 812, the gradient module 814, the base layer module 816, the uniform layer module 818, the detail layer module 820, and the UI module 822. The communication module 802 also facilitates communication with the other components of the server 602 as well as communication with the other components of the system 600 via the network interface module 702 and the network 610. For example, the communication module 802 may send signals between servers and the components of the painterly rendering module 604. In some implementations, the communication module 802 is a set of instructions executable by the processor 704. In some implementations, the communication module 802 is stored in the memory 706 of the server 602 and is accessible

and executable by the processor 704. In some implementations, the communication module 802 is adapted for cooperation and communication with the processor 704, the memory 706 and other components of the painterly rendering module 604 via the bus 720.

[0067] The sample module 804 can be software or routines for sampling an input image. For example, the sample module 804 may scan the input image in a specified region to determine the color of a brush stroke. The sample module 804 may also provide scanned data to the saliency module 812 and/or the gradient module 814 to generate gradient images and saliency maps. In some implementations, the sample module 804 is a set of instructions executable by the processor 704. In some implementations, the sample module 804 is stored in the memory 706 of the server 602 and is accessible and executable by the processor 704. In some implementations, the sample module 804 is adapted for cooperation and communication with the processor 704, the memory 706 and other components of the painterly rendering module 604 via the bus 720.

[0068] The brush module 806 can be software or routines for managing brush properties. The brush module 806 is coupled to receive from the base layer module 816, uniform layer module 818, and the detail layer module 820 request for brush strokes with particular properties. In some implementation the brush module 806 may communicate with the randomize module to randomize brush strokes and may further communicate with canvas module 808 to place brush strokes on the canvas and/or add brush strokes to the priority queue and placed on the canvas at a later time. The brush module 806 may also invoke functionality within the gradient module 814 and/or saliency module 812 in passing brush strokes to the canvas module 808. In some implementations, the brush module 806 is a set of instructions executable by the processor 704. In some implementations, the brush module 806 is stored in the memory 706 of the server 602 and is accessible and executable by the processor 704. In some implementations, the brush module 806 is adapted for cooperation and communication with the processor 704, the memory 706 and other components of the painterly rendering module 604 via the bus 720.

[0069] The canvas module 808 can be software or routines for sorting brush strokes by size using a priority queue (if any), placing brush strokes on a canvas, and presenting the canvas to the user as output. The sorting may be performed so that largest brush strokes are placed on the canvas first and the smallest, most detailed brush strokes are placed on the canvas last. In some implementations, the canvas module 808 is a set of instructions executable by the processor 704. In some implementations, the canvas module 808 is stored

in the memory 706 of the server 602 and is accessible and executable by the processor 704. In some implementations, the canvas module 808 is adapted for cooperation and communication with the processor 704, the memory 706 and other components of the painterly rendering module 604 via the bus 720.

[0070] The randomization module 810 can be software or routines for randomizing or pseudo-randomizing functionality in the painterly rendering module 604. For example, the randomization module 810 may be used to determine placement of uniform layer and/or detail layer brush strokes. Moreover, the randomization module 810 may be user to jitter brush stroke sizes and outlines. In some implementation, a low-discrepancy sequence may be used within the randomization module 810. In some implementations, the randomization module 810 is a set of instructions executable by the processor 704. In some implementations, the randomization module 810 is stored in the memory 706 of the server 602 and is accessible and executable by the processor 704. In some implementations, the randomization module 810 is adapted for cooperation and communication with the processor 704, the memory 706 and other components of the painterly rendering module 604 via the bus 720. In some implementations, the quasi-random numbers themselves may be precomputed and stored on disk or in memory, further saving processing time and energy consumption at runtime.

[0071] The saliency module 812 can be software or routines for detecting and maintaining salient details in an input image. The saliency module 812 can provide facial detection functionality on an input image and create a saliency map with the facial detection information and information received from the gradient module 814. In some implementations, the saliency module 812 is a set of instructions executable by the processor 704. In some implementations, the saliency module 812 is stored in the memory 706 of the server 602 and is accessible and executable by the processor 704. In some implementations, the saliency module 812 is adapted for cooperation and communication with the processor 704, the memory 706 and other components of the painterly rendering module 604 via the bus 720.

[0072] The gradient module 814 can be software or routines for computing a gradient image. More specifically, the gradient module 814 may receive request from the uniform layer module 818 and/or the detail layer module 820 so that the painterly rendering module 604 may place the brush strokes in the canvas module 808 with the correct directionality and amplitude. In some implementations, the gradient module 814 is a set of instructions

executable by the processor 704. In some implementations, the gradient module 814 is stored in the memory 706 of the server 602 and is accessible and executable by the processor 704. In some implementations, the gradient module 814 is adapted for cooperation and communication with the processor 704, the memory 706 and other components of the painterly rendering module 604 via the bus 720.

[0073] The base layer module 816 can be software or routines for placing brush strokes in the canvas module 808 for the base layer. In some implementations, the base layer module 816 is a set of instructions executable by the processor 704. In some implementations, the base layer module 816 is stored in the memory 706 of the server 602 and is accessible and executable by the processor 704. In some implementations, the base layer module 816 is adapted for cooperation and communication with the processor 704, the memory 706 and other components of the painterly rendering module 604 via the bus 720.

[0074] The uniform layer module 818 can be software or routines for placing brush strokes in the canvas module 808 for the uniform layer. In some implementations, the uniform layer module 818 is a set of instructions executable by the processor 704. In some implementations, the uniform layer module 818 is stored in the memory 706 of the server 602 and is accessible and executable by the processor 704. In some implementations, the uniform layer module 818 is adapted for cooperation and communication with the processor 704, the memory 706 and other components of the painterly rendering module 604 via the bus 720.

[0075] The detail layer module 820 can be software or routines for placing brush strokes in the canvas module 808 for the detail layer. In some implementations, the detail layer module 820 is a set of instructions executable by the processor 704. In some implementations, the detail layer module 820 is stored in the memory 706 of the server 602 and is accessible and executable by the processor 704. In some implementations, the detail layer module 820 is adapted for cooperation and communication with the processor 704, the memory 706 and other components of the painterly rendering module 604 via the bus 720.

[0076] The UI module 822 can be software or routines for generating a user interface element. The UI module 822 also may receive input images and present a rendering painterly picture for to the user. The UI module 822 may also allow the user to adjust or tweak variables to the functionality described above. For example, brush stroke properties and thresholds may be set using the UI module 822. In some implementations, the UI module 822 is a set of instructions executable by the processor 704. In some implementations, the UI

module 822 is stored in the memory 706 of the server 602 and is accessible and executable by the processor 704. In some implementations, the UI module 822 is adapted for cooperation and communication with the processor 704, the memory 706 and other components of the painterly rendering module 604 via the bus 720.

[0077] In the above description, for purposes of explanation, numerous specific details were set forth. It will be apparent, however, that the disclosed technologies can be practiced without any given subset of these specific details. In other instances, structures and devices are shown in block diagram form. For example, the disclosed technologies are described in some implementations above with reference to user interfaces and particular hardware.

[0078] Reference in the specification to "one implementation" or "an implementation" means that a particular feature, structure, or characteristic described in connection with the implementation is included in at least one implementation of the disclosed technologies. The appearances of the phrase "in one implementation" in various places in the specification are not necessarily all referring to the same implementation.

[0079] Some portions of the detailed descriptions above were presented in terms of processes and symbolic representations of operations on data bits within a computer memory. A process can generally be considered a self-consistent sequence of steps leading to a result. The steps may involve physical manipulations of physical quantities. These quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. These signals may be referred to as being in the form of bits, values, elements, symbols, characters, terms, numbers or the like.

[0080] These and similar terms can be associated with the appropriate physical quantities and can be considered labels applied to these quantities. Unless specifically stated otherwise as apparent from the prior discussion, it is appreciated that throughout the description, discussions utilizing terms for example "processing" or "computing" or "calculating" or "determining" or "displaying" or the like, may refer to the action and processes of a computer system, or similar electronic computing device, that manipulates and transforms data represented as physical (electronic) quantities within the computer system's registers and memories into other data similarly represented as physical quantities within the computer system memories or registers or other such information storage, transmission or display devices.

[0081] The disclosed technologies may also relate to an apparatus for performing the operations herein. This apparatus may be specially constructed for the required purposes, or it may include a general-purpose computer selectively activated or reconfigured by a computer program stored in the computer. Such a computer program may be stored in a computer readable storage medium, for example, but is not limited to, any type of disk including floppy disks, optical disks, compact disc read-only memories (CD-ROMs), and magnetic disks, read-only memories (ROMs), random access memories (RAMs), erasable programmable read-only memories (EPROMs), electrically erasable programmable read-only memory (EEPROMs), magnetic or optical cards, flash memories including USB keys with non-volatile memory or any type of media suitable for storing electronic instructions, each coupled to a computer system bus.

[0082] The disclosed technologies can take the form of an entirely hardware implementation, an entirely software implementation or an implementation containing both hardware and software elements. In some implementations, the technology is implemented in software, which includes but is not limited to firmware, resident software, microcode, etc.

[0083] Furthermore, the disclosed technologies can take the form of a computer program product accessible from a non-transitory computer-usable or computer-readable medium providing program code for use by or in connection with a computer or any instruction execution system. For the purposes of this description, a computer-usable or computer-readable medium can be any apparatus that can contain, store, communicate, propagate, or transport the program for use by or in connection with the instruction execution system, apparatus, or device.

[0084] A computing system or data processing system suitable for storing and/or executing program code will include at least one processor (e.g., a hardware processor) coupled directly or indirectly to memory elements through a system bus. The memory elements can include local memory employed during actual execution of the program code, bulk storage, and cache memories which provide temporary storage of at least some program code in order to reduce the number of times code must be retrieved from bulk storage during execution.

[0085] Input/output (I/O) devices (including but not limited to keyboards, displays, pointing devices, etc.) can be coupled to the system either directly or through intervening I/O controllers.

[0086] Network adapters may also be coupled to the system to enable the data processing system to become coupled to other data processing systems or remote printers or storage devices through intervening private or public networks. Modems, cable modems and Ethernet cards are just a few of the currently available types of network adapters.

[0087] Finally, the processes and displays presented herein may not be inherently related to any particular computer or other apparatus. Various general-purpose systems may be used with programs in accordance with the teachings herein, or it may prove convenient to construct more specialized apparatus to perform the required method steps. The required structure for a variety of these systems will appear from the description below. In addition, the disclosed technologies were not described with reference to any particular programming language. It will be appreciated that a variety of programming languages may be used to implement the teachings of the technologies as described herein.

[0088] The foregoing description of the implementations of the present techniques and technologies has been presented for the purposes of illustration and description. It is not intended to be exhaustive or to limit the present techniques and technologies to the precise form disclosed. Many modifications and variations are possible in light of the above teaching. It is intended that the scope of the present techniques and technologies be limited not by this detailed description. The present techniques and technologies may be implemented in other specific forms without departing from the spirit or essential characteristics thereof. Likewise, the particular naming and division of the modules, routines, features, attributes, methodologies and other aspects are not mandatory or significant, and the mechanisms that implement the present techniques and technologies or its features may have different names, divisions and/or formats. Furthermore, the modules, routines, features, attributes, methodologies and other aspects of the present technology can be implemented as software, hardware, firmware or any combination of the three. Also, wherever a component, an example of which is a module, is implemented as software, the component can be implemented as a standalone program, as part of a larger program, as a plurality of separate programs, as a statically or dynamically linked library, as a kernel loadable module, as a device driver, and/or in every and any other way known now or in the future in computer programming. Additionally, the present techniques and technologies are in no way limited to implementation in any specific programming language, or for any specific operating system or environment. Accordingly, the disclosure of the present techniques and technologies is intended to be illustrative, but not limiting.

WHAT IS CLAIMED IS:

1. A computer-implemented method comprising:

receiving, using one or more processors, an input image;

adding, using one or more processors, a base layer of brush strokes based on a sampling of underlying pixel values in the input image;

adding, using one or more processors, a uniform layer of brush strokes iteratively based on a sampling of underlying pixel values in the input image and directionally based on a gradient of the input image; and

adding, using one or more processors, a detail layer of brush strokes based on a determined saliency map.

2. The method of claim 1, wherein the saliency map is determined based on one or more areas within the input image that include faces as identified by facial recognition.

3. The method of claim 1, wherein the saliency map is determined based on a gradient image of the input image.

4. The method of claim 1, wherein placement of the uniform layer of brush strokes is based on a random or quasi-random algorithm.

5. The method of claim 1, wherein brush stroke size in a subsequent iteration is about half of a brush stroke size of a previous iteration of the uniform layer of brush strokes.

6. The method of claim 1, wherein sizes and outlines of the uniform layer of brush strokes are jittered using a low-discrepancy sequence.

7. The method of claim 1, wherein brush strokes are added to a priority queue, and the method further comprises:

sorting, using one or more processors, the priority queue; and

placing, using one or more processors, the brush strokes in the priority queue on a canvas.

8. The method of claim 1, wherein the brush strokes of the base layer substantially cover a canvas.

9. A computer program product comprising a non-transitory computer usable medium including a computer readable program, wherein the computer readable program when executed on a computer causes the computer to:

receive an input image;

add a base layer of brush strokes based on a sampling of underlying pixel values in the input image;

add a uniform layer of brush strokes iteratively based on a sampling of underlying pixel values in the input image and directionally based on a gradient of the input image; and

add a detail layer of brush strokes based on a determined saliency map.

10. The computer program product of claim 9, wherein the saliency map is determined based on one or more areas within the input image that include faces as identified by facial recognition.

11. The computer program product of claim 9, wherein placement of the uniform layer of brush strokes is based on a random or quasi-random algorithm.

12. The computer program product of claim 9, wherein brush stroke size in a subsequent iteration is about half of a brush stroke size of a previous iteration of the uniform layer of brush strokes.

13. A system comprising:

a processor; and

a memory storing instructions that, when executed, cause the system to:

receive an input image;

add a base layer of brush strokes based on a sampling of underlying pixel values in the input image;

add a uniform layer of brush strokes iteratively based on a sampling of underlying pixel values in the input image and directionally based on a gradient of the input image; and

add a detail layer of brush strokes based on a determined saliency map.

14. The system of claim 13, wherein the saliency map is determined based on one or more areas within the input image that include faces as identified by facial recognition.

15. The system of claim 13, wherein the saliency map determined is based on a gradient image of the input image.

16. The system of claim 13, wherein placement of the uniform layer of brush strokes is based on a random or quasi-random algorithm.

17. The system of claim 13, wherein brush stroke size in a subsequent iteration is about half of a brush stroke size of a previous iteration of the uniform layer of brush strokes.

18. The system of claim 13, wherein sizes and outlines of the uniform layer of brush strokes are jittered using a low-discrepancy sequence.

19. The system of claim 13, wherein brush strokes are added to a priority queue, and the memory further storing instructions that, when executed, cause the system to:

sort the priority queue; and

place the brush strokes in the priority queue on a canvas.

20. The system of claim 13, wherein the brush strokes of the base layer substantially cover a canvas.

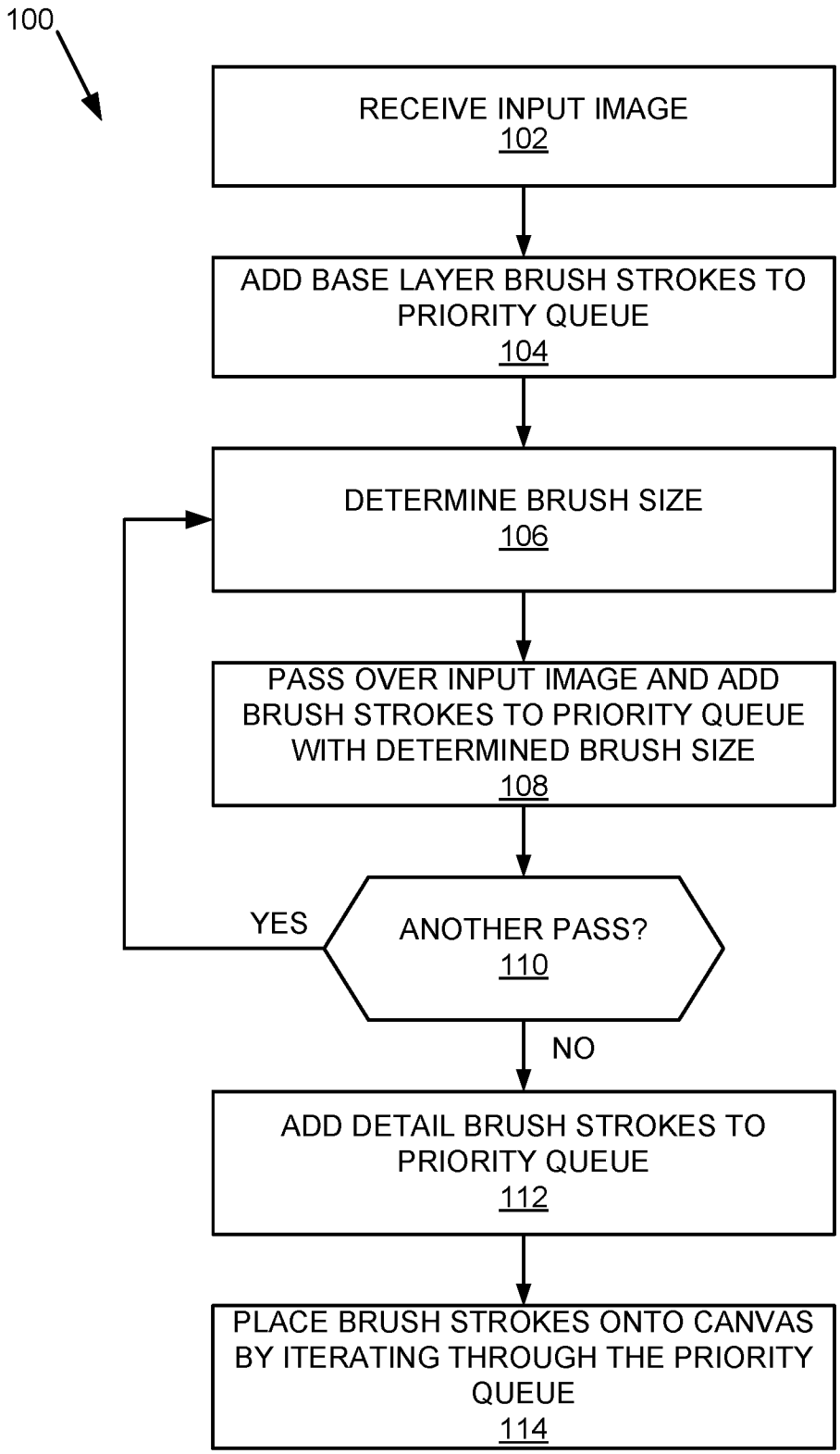
100
RECEIVE INPUT IMAGE
102
ADD BASE LAYER BRUSH STROKES TO PRIORITY QUEUE
104
DETERMINE BRUSH SIZE
106
PASS OVER INPUT IMAGE AND ADD BRUSH STROKES TO PRIORITY QUEUE WITH DETERMINED BRUSH SIZE
108
ANOTHER PASS?
110
YES
NO
ADD DETAIL BRUSH STROKES TO PRIORITY QUEUE
112
PLACE BRUSH STROKES ONTO CANVAS BY ITERATING THROUGH THE PRIORITY QUEUE
114

Figure 1

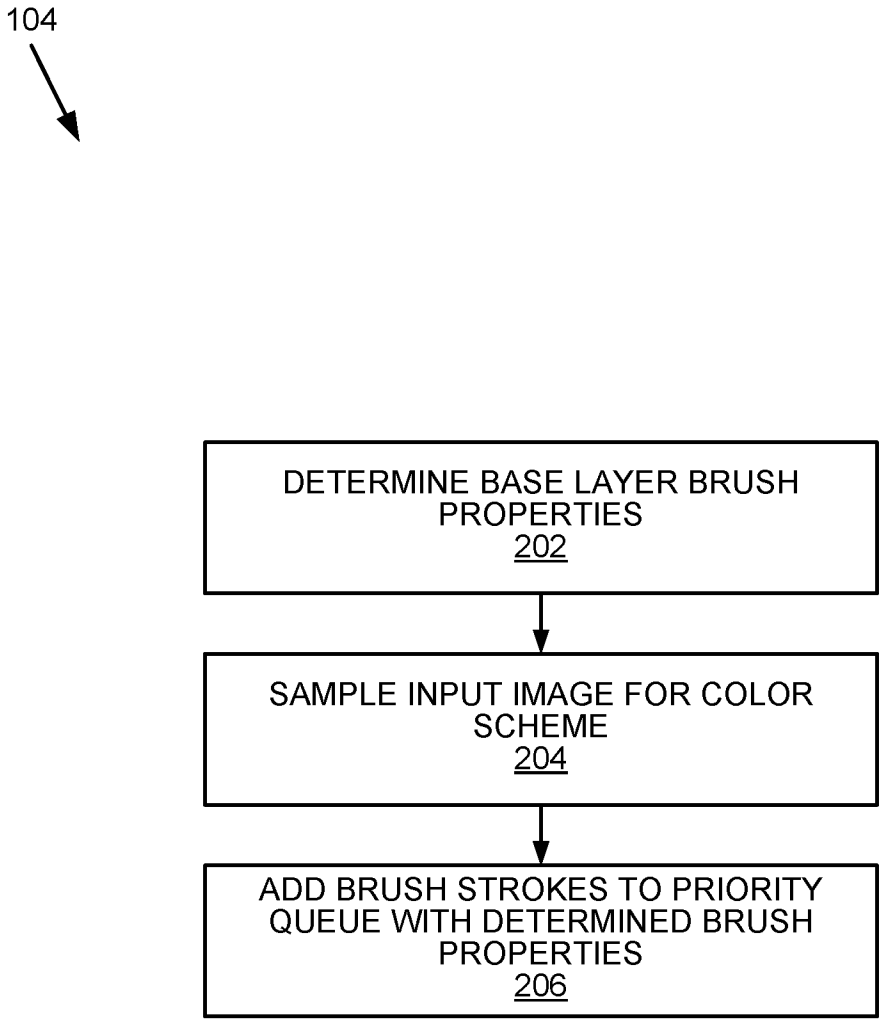
104
DETERMINE BASE LAYER BRUSH PROPERTIES
202
SAMPLE INPUT IMAGE FOR COLOR SCHEME
204
ADD BRUSH STROKES TO PRIORITY QUEUE WITH DETERMINED BRUSH PROPERTIES
206

Figure 2

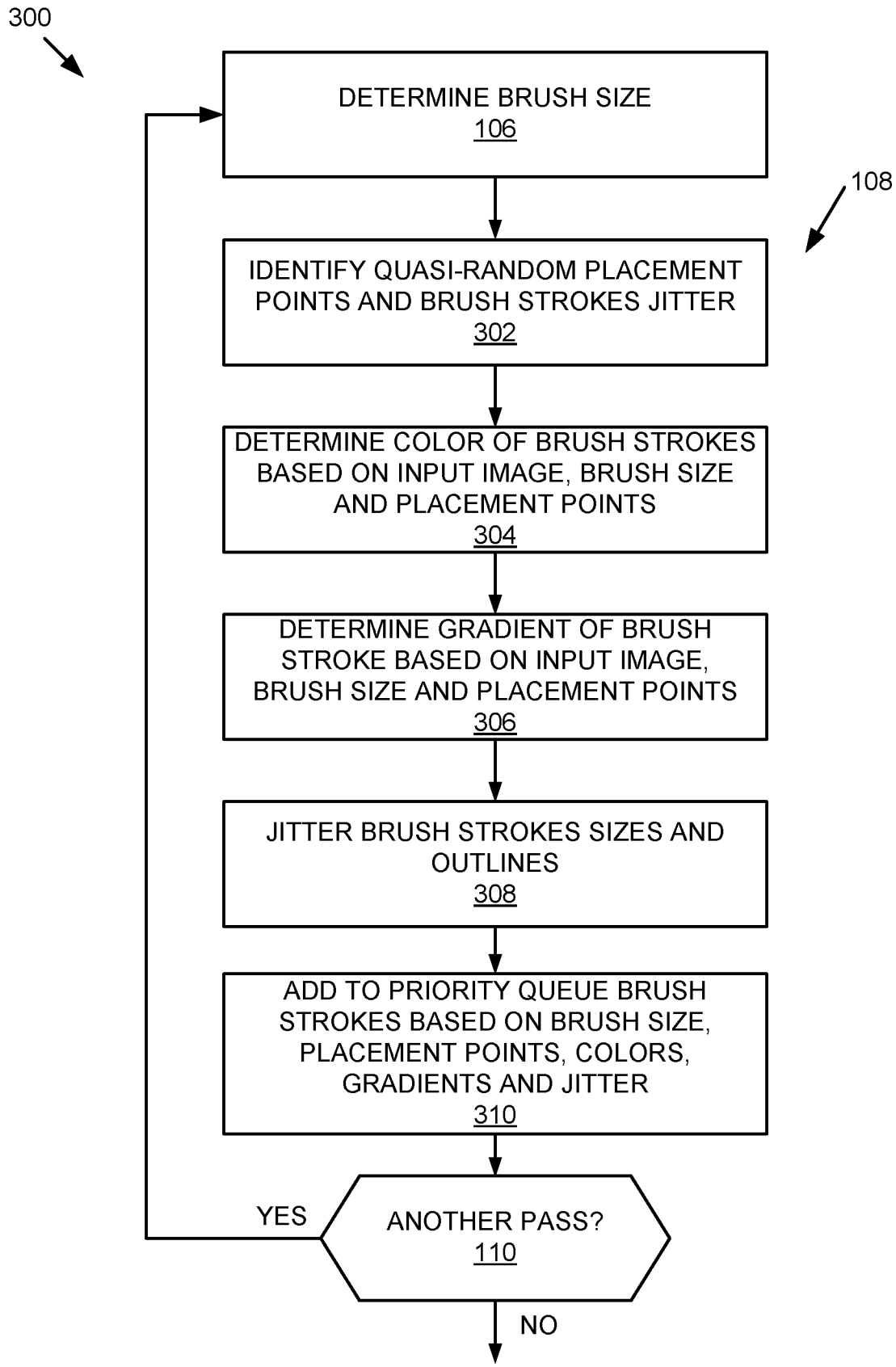
300
DETERMINE BRUSH SIZE
106
108
IDENTIFY QUASI-RANDOM PLACEMENT POINTS AND BRUSH STROKES JITTER
302
DETERMINE COLOR OF BRUSH STROKES BASED ON INPUT IMAGE, BRUSH SIZE AND PLACEMENT POINTS
304
DETERMINE GRADIENT OF BRUSH STROKE BASED ON INPUT IMAGE, BRUSH SIZE AND PLACEMENT POINTS
306
JITTER BRUSH STROKES SIZES AND OUTLINES
308
ADD TO PRIORITY QUEUE BRUSH STROKES BASED ON BRUSH SIZE, PLACEMENT POINTS, COLORS, GRADIENTS AND JITTER
310
ANOTHER PASS?
110
YES
NO

Figure 3

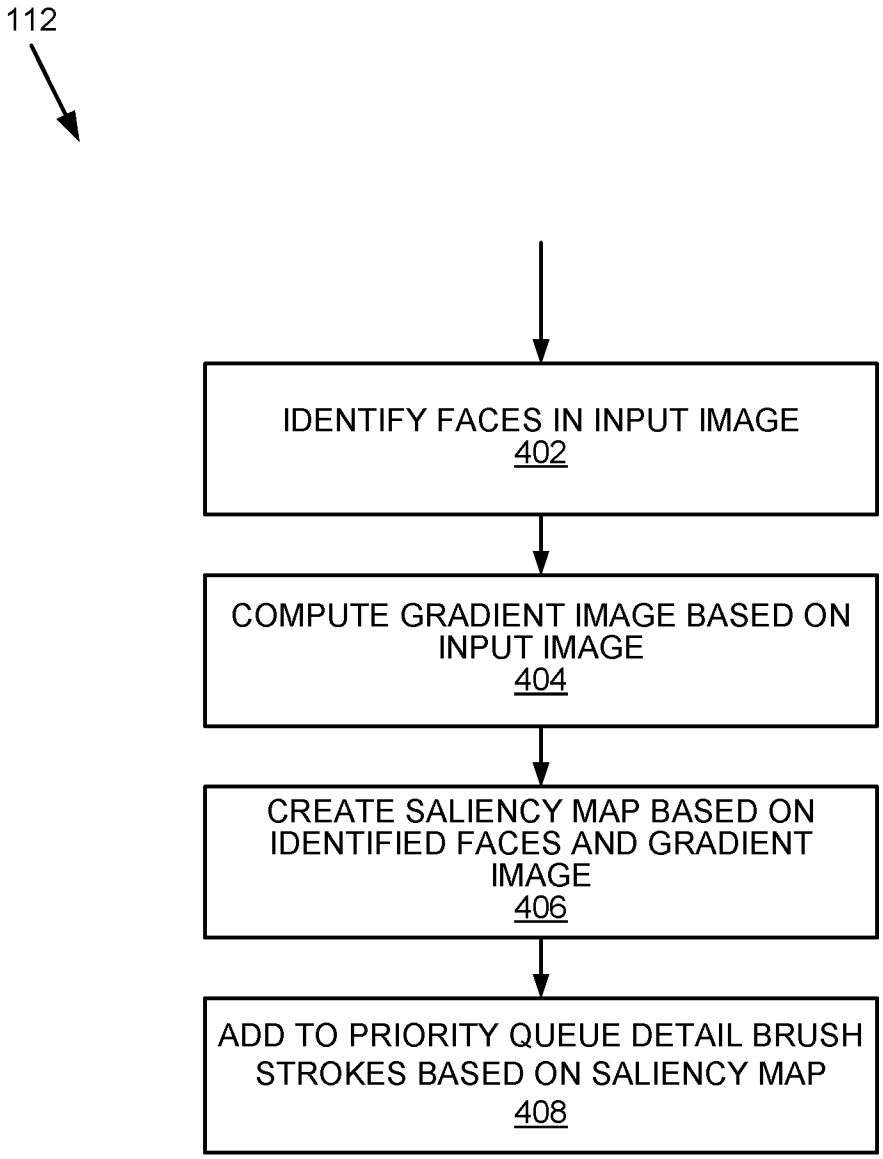
112
IDENTIFY FACES IN INPUT IMAGE
402
COMPUTE GRADIENT IMAGE BASED ON INPUT IMAGE
404
CREATE SALIENCY MAP BASED ON IDENTIFIED FACES AND GRADIENT IMAGE
406
ADD TO PRIORITY QUEUE DETAIL BRUSH STROKES BASED ON SALIENCY MAP
408

Figure 4

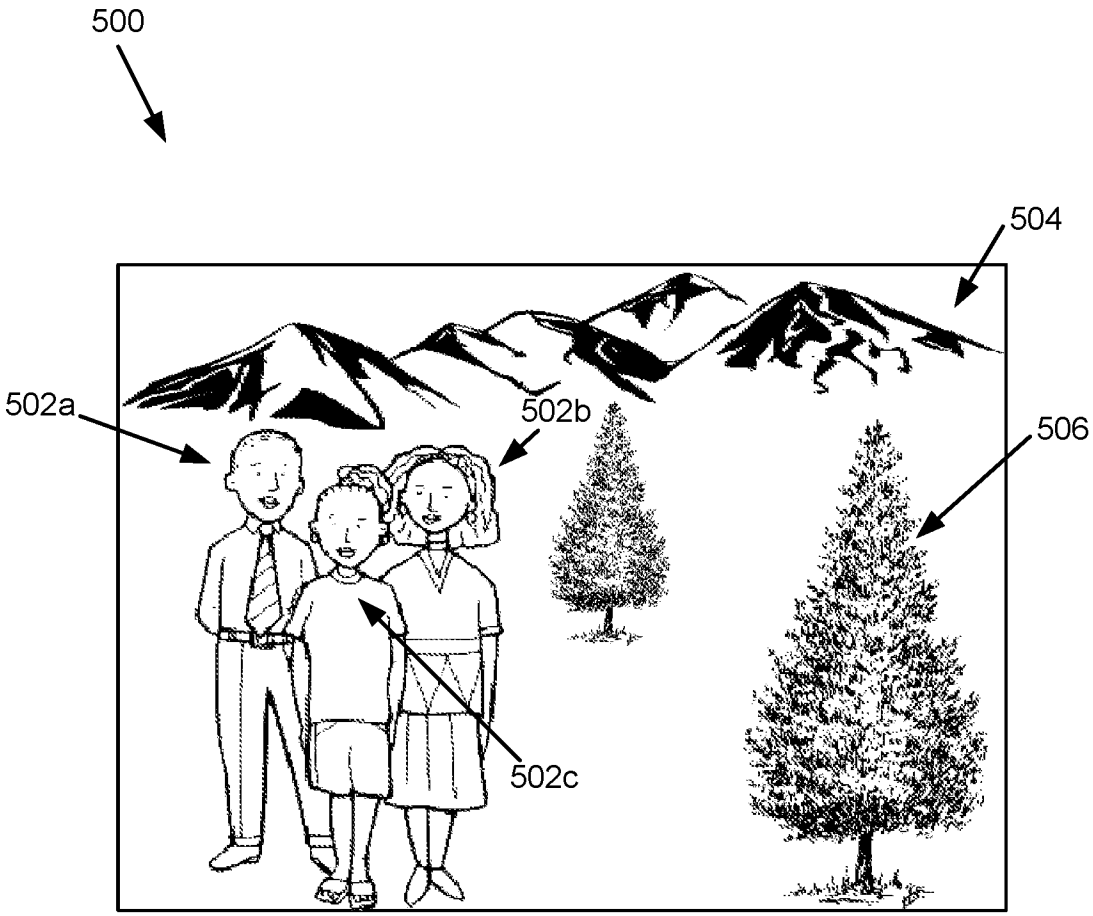

Figure 5

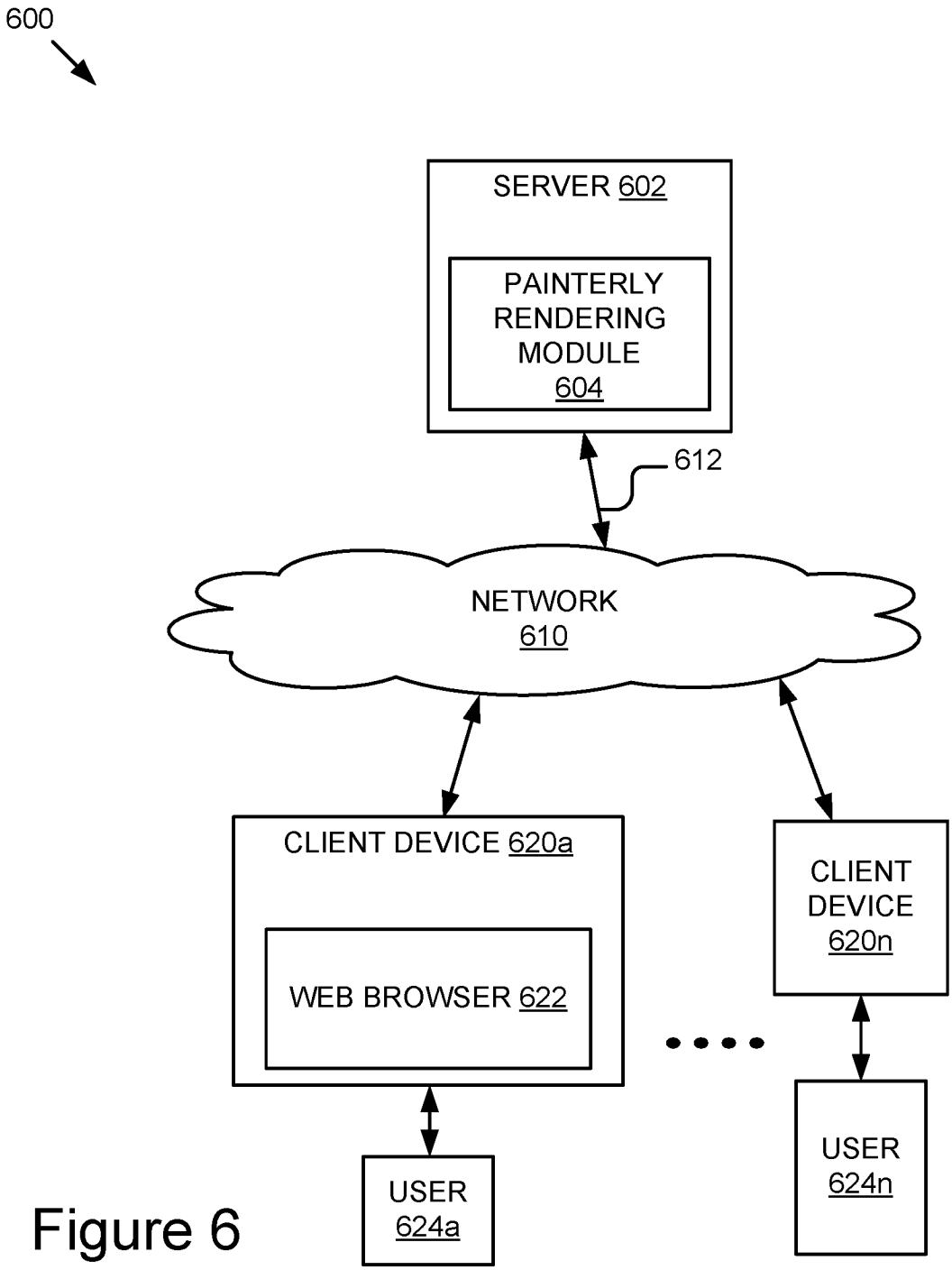
600
SERVER 602
PAINTERLY RENDERING MODULE 604
612
NETWORK 610
CLIENT DEVICE 620a
WEB BROWSER 622
CLIENT DEVICE 620n
USER 624a
USER 624n

Figure 6

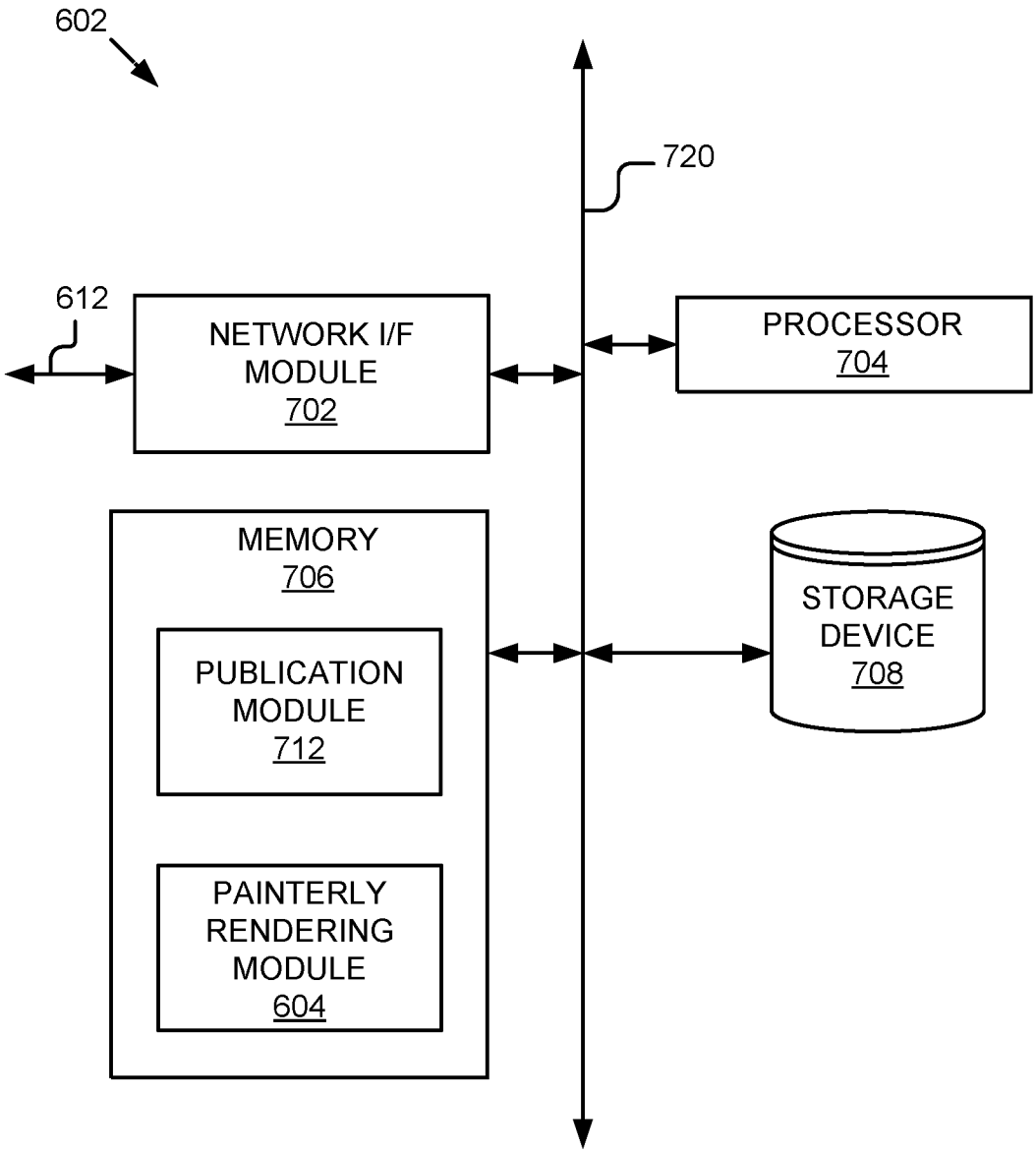
602
720
612
NETWORK I/F MODULE 702
PROCESSOR 704
MEMORY 706
PUBLICATION MODULE 712
PAINTERLY RENDERING MODULE 604
STORAGE DEVICE 708

Figure 7

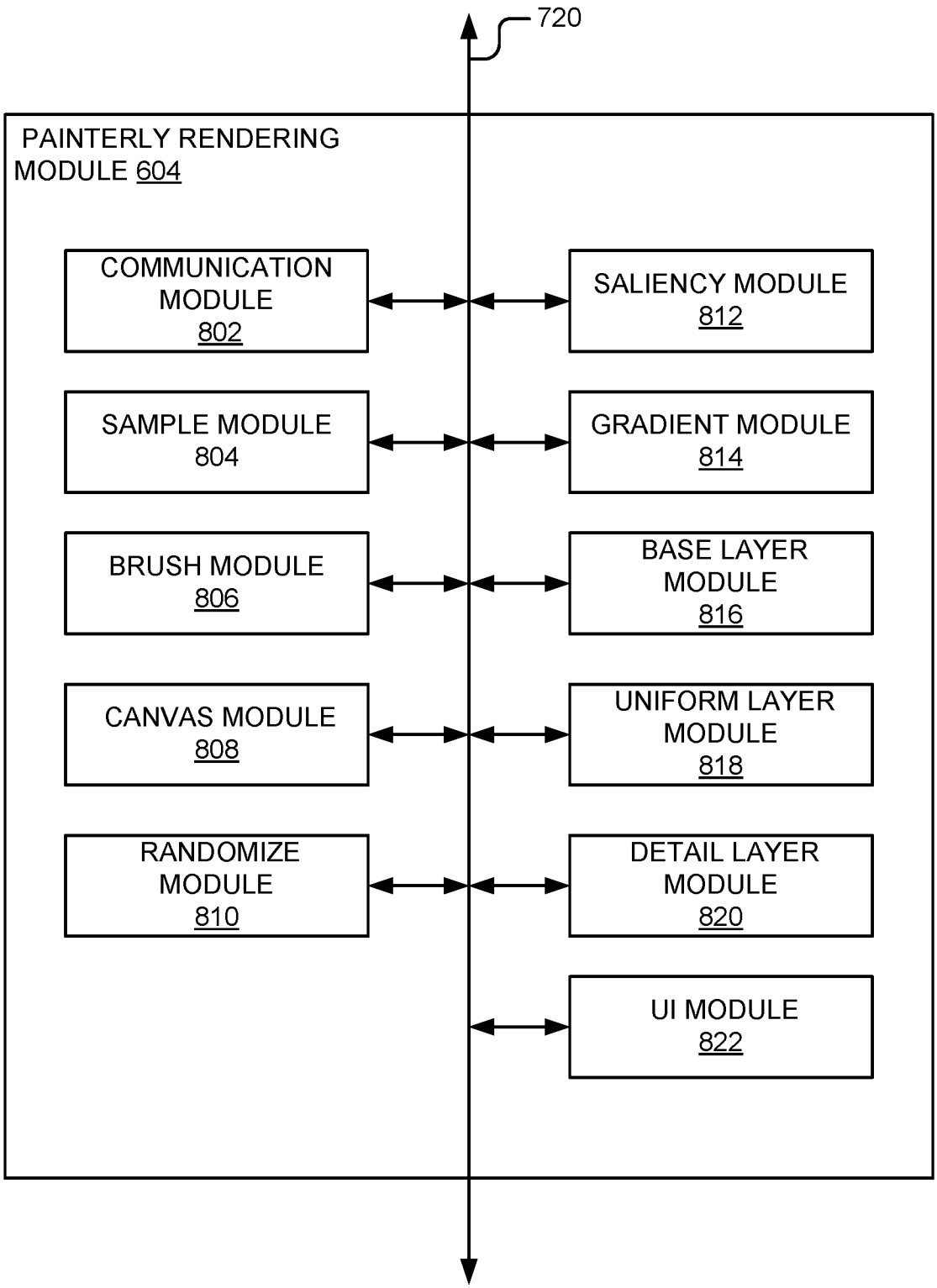
720
PAINTERLY RENDERING MODULE 604
COMMUNICATION MODULE 802
SALIENCY MODULE 812
SAMPLE MODULE 804
GRADIENT MODULE 814
BRUSH MODULE 806
BASE LAYER MODULE 816
CANVAS MODULE 808
UNIFORM LAYER MODULE 818
RANDOMIZE MODULE 810
DETAIL LAYER MODULE 820
UI MODULE 822

Figure 8

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2016/027569

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06T11/00
ADD.

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)
G06T

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)
EPO-Internal, COMPENDEX, INSPEC, IBM-TDB, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	HAYS J ET AL: "IMAGE AND VIDEO BASED PAINTERLY ANIMATION", PROCEEDINGS NPAR 2004. 3RD. INTERNATIONAL SYMPOSIUM ON NON-PHOTOREALISTIC ANIMATION AND RENDERING. ANNECY, FRANCE, JUNE 7 - 9, 2004; [SYMPOSIUM ON NON - PHOTOREALISTIC ANIMATION AND RENDERING], NEW YORK, NY : ACM, US, 7 June 2004 (2004-06-07), pages 113-120, XP001210029, DOI: 10.1145/987657.987676 ISBN: 978-1-58113-887-0 abstract page 115, paragraph 3 - page 118, paragraph 4.G ----- -/--	1-20

[X] 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

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
1 July 2016	12/07/2016

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Zamuner, Umberto

2

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	HERTZMANN A ED - ASSOCIATION FOR COMPUTING MACHINERY: "PAINTERLY RENDERING WITH CURVED BRUSH STROKES OF MULTIPLE SIZES", PROCEEDINGS OF THE 1998 EUROGRAPHICS / SIGGRAPH WORKSHOP ON GRAPHICS HARDWARE. LISBON, AUG. 31 - SEPT. 1, 1998; [EUROGRAPHICS / SIGGRAPH WORKSHOP ON GRAPHICS HARDWARE], NEW YORK, NY : ACM, US, vol. WORKSHOP 2, 1 January 1998 (1998-01-01), pages 453-461, XP000994967, ISBN: 978-1-58113-097-3 abstract page 454, paragraph 1.2 - page 455, right-hand column, last line page 458, paragraph 3.1 - paragraph 3.2 -----	1-20
Y	J.P. COLLOMOSSE ET AL: "Painterly rendering using image salience", EUROGRAPHICS UK CONFERENCE, 2002. PROCEEDINGS. THE 20TH, 1 January 2002 (2002-01-01), pages 122-128, XP055284896, DOI: 10.1109/EGUK.2002.1011281 ISBN: 978-0-7695-1518-2 abstract page 122, paragraph 1 page 123, paragraph 3 - page 127, paragraph 5 -----	1,4,7,9, 11,13, 16,20
Y	LITWINOWICZ P ED - ASSOCIATION FOR COMPUTING MACHINERY: "PROCESSING IMAGES AND VIDEO FOR AN IMPRESSIONIST EFFECT", COMPUTER GRAPHICS PROCEEDINGS. SIGGRAPH 97. LOS ANGELES, AUG. 3 - 8, 1997; [COMPUTER GRAPHICS PROCEEDINGS. SIGGRAPH], READING, ADDISON WESLEY, US, 3 August 1997 (1997-08-03), pages 407-414, XP000765841, ISBN: 978-0-201-32220-0 abstract page 408, paragraph 3 - page 410, paragraph 3.B page 412, left-hand column, line 1 - right-hand column, line 5 ----- -/--	1,4,9, 11,13,16

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	JAN ERIC KYPRIANIDIS ET AL: "State of the Art: A Taxonomy of Artistic Stylization Techniques for Images and Video", IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS, IEEE SERVICE CENTER, LOS ALAMITOS, CA, US, vol. 19, no. 5, 1 May 2013 (2013-05-01), pages 866-885, XP011497433, ISSN: 1077-2626, DOI: 10.1109/TVCG.2012.160 abstract page 867, paragraph 2.1 - page 868 page 870, paragraph 3 - page 871, paragraph 3.3 page 872, paragraph 4.1 - page 873, paragraph 4.2.1 -----	1-20

2