(12) STANDARD PATENT APPLICATION (11) Application No. **AU 2005211665 A1**
(19) AUSTRALIAN PATENT OFFICE

(54) Title
Vectorisation of colour gradients

(51) International Patent Classification(s)
G06T 7/40 (2006.01)

(21) Application No: **2005211665** (22) Date of Filing: **2005.09.23**

(43) Publication Date: **2007.04.19**
(43) Publication Journal Date: **2007.04.19**

(71) Applicant(s)
Canon Kabushiki Kaisha

(72) Inventor(s)
Allen, Matthew James

(74) Agent / Attorney
Spruson & Ferguson, Level 35 St Martins Tower 31 Market Street, Sydney, NSW, 2000

ABSTRACT

VECTORISATION OF COLOUR GRADIENTS

A method is described of generating a vector description of a colour gradient in an image. The image comprises an array of pixels, each pixel having one or more associated colour values. The image has at least one gradient area in which at least one of the colour values varies between neighbouring pixels. In each gradient area, a path is generated (step 210) comprising pixels having substantially the same colour. Linear paths (403) and circular paths (501) are described. One or more control points are determined (step 220, 230) defining a line substantially perpendicular to the path. Representative colour values are calculated at points along the line and a piecewise-linear colour function is generated from the representative colour values (step 240). The one or more control points and the piecewise-linear colour function form a vector description of the colour gradient in the gradient area.

2005211665 23 Sep 2005

S&F Ref: 727433

AUSTRALIA

PATENTS ACT 1990

COMPLETE SPECIFICATION

FOR A STANDARD PATENT

Name and Address of Applicant : Canon Kabushiki Kaisha, of 30-2, Shimomaruko 3-chome, Ohta-ku, Tokyo, 146, Japan

Actual Inventor(s): Matthew James Allen

Address for Service:
Spruson & Ferguson
St Martins Tower Level 35
31 Market Street
Sydney NSW 2000
(CCN 3710000177)

Invention Title: Vectorisation of colour gradients

The following statement is a full description of this invention, including the best method of performing it known to me/us:-

VECTORISATION OF COLOUR GRADIENTS

FIELD OF INVENTION

The current invention relates to computer graphics and in particular to converting colour gradients in images into a higher level vector form.

BACKGROUND

It is often desirable to convert image data into a higher-level form. Such conversion may allow the image data to be stored more efficiently. In addition, the resultant higher-level description may be required for further processing of the image data.

Prior art methods for converting an image from a simple raster bitmap into a higher level form like a vector graphics description involve either leaving the colour data in bitmap form or converting small areas of similar colour to flat filled objects.

In the first case where a bitmap is used for the colour data of the converted image, lots of memory is used in storing the bitmap. In the second case in which small areas are converted to flat filled objects, the main problem is that rendering all the areas of flat colour is computationally expensive. An example of this is shown in Fig. 3, where shape 352 having a gradient, or blend, fill has been converted to a format in which areas of similar colour are replaced by objects with a flat colour fill. The rings of colour labelled 350 are distinct objects with no relationship in the output. Each of the rings 350 has a flat colour.

In both of the described cases, the resultant colour data may be difficult to edit using simple controls (for example, changing the start and/or end points for a linear gradient). In addition, the colour data consumes a non-trivial amount of memory.

SUMMARY OF THE INVENTION

It is an object of the present invention to substantially overcome, or at least ameliorate, one or more disadvantages of existing arrangements.

According to a first aspect of the invention there is provided a method of generating a vector description of a colour gradient in an image, said method comprising the steps of:

receiving the image comprising an array of pixels, each pixel having one or more associated colour values, the image having at least one gradient area in which at least one of the colour values varies between neighbouring pixels;

generating, in each gradient area, a path comprising pixels having substantially the same colour;

determining one or more control points defining a line substantially perpendicular to the path;

calculating representative colour values at points along the line; and

generating a piecewise-linear colour function from the representative colour values,

wherein the one or more control points and the piecewise-linear colour function form a vector description of the colour gradient in the gradient area.

According to a second aspect of the invention there is provided an apparatus for generating a vector description of a colour gradient in an image, said apparatus comprising:

means for receiving the image comprising an array of pixels, each pixel having one or more associated colour values, the image having at least one gradient area in which at least one of the colour values varies between neighbouring pixels;

means for generating, in each gradient area, a path comprising pixels having substantially the same colour;

means for determining one or more control points defining a line substantially perpendicular to the path;

means for calculating representative colour values at points along the line; and

means for generating a piecewise-linear colour function from the representative colour values,

wherein the one or more control points and the piecewise-linear colour function form a vector description of the colour gradient in the gradient area.

According to a further aspect of the invention there is provided a computer program product comprising machine-readable program code recorded on a machine-readable recording medium, for controlling the operation of a data processing machine on which the program code executes to perform a method of generating a vector description of a colour gradient in an image, said method comprising the steps of:

receiving the image comprising an array of pixels, each pixel having one or more associated colour values, the image having at least one gradient area in which at least one of the colour values varies between neighbouring pixels;

generating, in each gradient area, a path comprising pixels having substantially the same colour;

determining one or more control points defining a line substantially perpendicular to the path;

calculating representative colour values at points along the line; and

generating a piecewise-linear colour function from the representative colour values, wherein the one or more control points and the piecewise-linear colour function form a vector description of the colour gradient in the gradient area.

According to a further aspect of the invention there is provided a computer program comprising machine-readable program code for controlling the operation of a data processing apparatus on which the program code executes to perform a method of generating a vector description of a colour gradient in an image, said computer program comprising:

code for receiving the image comprising an array of pixels, each pixel having one or more associated colour values, the image having at least one gradient area in which at least one of the colour values varies between neighbouring pixels;

code for generating, in each gradient area, a path comprising pixels having substantially the same colour;

code for determining one or more control points defining a line substantially perpendicular to the path;

code for calculating representative colour values at points along the line; and

code for generating a piecewise-linear colour function from the representative colour values,

wherein the one or more control points and the piecewise-linear colour function form a vector description of the colour gradient in the gradient area.

BRIEF DESCRIPTION OF THE DRAWINGS

One or more embodiments of the present invention will now be described with reference to the following drawings, in which:

Fig. 1 is a flow chart of a method of converting colour gradients in an image to vector gradients;

Fig. 2 shows an object used to illustrate the path generation step of the method of Fig. 1;

Fig. 3 shows an object used to illustrate a prior art method for approximating a raster bitmap fill;

Fig. 4 shows an object used to illustrate finding the parameters of a linear gradient according to the method of Fig. 1;

Fig. 5 shows an object used to illustrate finding the parameters of a radial gradient according to the method of Fig. 1;

Fig. 6 is a schematic block diagram of a system in which the method of Fig. 1 may be applied; and

Fig. 7 is a schematic block diagram of a general-purpose computer on which the described methods may be practiced.

DETAILED DESCRIPTION INCLUDING BEST MODE

There are many circumstances in which it may be desired to convert a raster bitmap into a higher-level description. One such application is in processing scanned images, as described below with reference to Fig. 6. However, the described methods are not limited to this particular application, but may be used more generally where there is a need to identify gradient information in an input image and to convert the gradient information to a vector description.

Such a need may arise, for example, when processing the output of a renderer. The output may contain too much detail to provide an efficient description for further processing. Generating a vector description may help to accelerate any further processing or to reduce the size of the output description.

Operating environment

Fig. 6 illustrates a system in which the methods described herein may be applied. Scanner 600 is used to convert a document or image to an electronic bitmap, which may be sent to computer 700 for further processing. The bitmap may be converted into a searchable form, for example by applying Optical Character Recognition (OCR). Converting the bitmap to a vector graphics format may provide a more efficient description for storage or further processing.

The scanner 600 includes a plate 602 for supporting and positioning an input document or image. The input is illuminated by the lamp 604, which may be a xenon lamp or a cold cathode fluorescent lamp, for example. A lens assembly 606 conveys images of all or part of the input to a system of optical filters 608. The scanner may have three sets of filters to transmit red, green and blue light respectively. The scanning assembly 610, which may include charge-coupled device (CCD) arrays, scans the output of filters 608. The CCD arrays include light-sensitive diodes that convert photons into electrons, thus forming an electronic description of the image arriving via the lens system 606 and filters 608. In a single-scan arrangement, there is a separate CCD array for each of the red, green and blue components. Alternatively, the input may be scanned repeatedly, with a different filter being placed before a CCD array for each scan to register a different colour component of the image. The image description may be transferred to a computer 700 for storage or processing. Driver software running on the computer 700 controls communication between the computer 700 and the scanner 600. The devices 600, 700 may be linked via a network, which may also communicate with one or more printers for printing copies of the input document.

Description of general-purpose computer

The algorithms and displays presented herein are not inherently related to any particular computer or other apparatus. Various general purpose machines may be used with programs in accordance with the teachings herein. Alternatively, the construction of more specialized apparatus to perform the required method steps may be appropriate. The structure of a conventional general purpose computer will appear from the description below.

In addition, the present invention also implicitly discloses a computer program, in that it would be apparent to the person skilled in the art that the individual steps of the methods described herein are to be put into effect by computer code. The computer program is not intended to be limited to any particular programming language and implementation thereof. It will be appreciated that a variety of programming languages and coding thereof may be used to implement the teachings of the disclosure contained herein.

Such a computer program may be stored on any computer readable medium. The computer readable medium may include storage devices such as magnetic or optical disks, memory chips, or other storage devices suitable for interfacing with a general purpose computer. The computer readable medium may also include a hard-wired medium such as exemplified in the Internet system, or wireless medium such as exemplified in the GSM mobile telephone system. The computer program when loaded and executed on such a general-purpose computer effectively results in an apparatus that implements the steps of the preferred method.

The method of converting colour gradients to a vector description is preferably practiced using a general-purpose computer system 700, such as that shown in Fig. 7

wherein the processes of Fig.1 may be implemented as software, such as an application program executing within the computer system 700. In particular, the method steps are effected by instructions in the software that are carried out by the computer. The instructions may be formed as one or more code modules, each for performing one or more particular tasks. The software may be stored in a computer readable medium, including the storage devices described below. The software is loaded into the computer from the computer readable medium, and then executed by the computer. A computer readable medium having such software or computer program recorded on it is a computer program product. The use of the computer program product in the computer preferably effects an advantageous apparatus for generating a vector description of colour gradients.

The computer system 700 is formed by a computer module 701, input devices such as a keyboard 702 and mouse 703, output devices including a printer 715, a display device 714 and loudspeakers 717. A Modulator-Demodulator (Modem) transceiver device 716 is used by the computer module 701 for communicating to and from a communications network 720, for example connectable via a telephone line 721 or other functional medium. The modem 716 can be used to obtain access to the Internet, and other network systems, such as a Local Area Network (LAN) or a Wide Area Network (WAN), and may be incorporated into the computer module 701 in some implementations.

The computer module 701 typically includes at least one processor unit 705, and a memory unit 706, for example formed from semiconductor random access memory (RAM) and read only memory (ROM). The module 701 also includes an number of input/output (I/O) interfaces including an audio-video interface 707 that couples to the video display 714 and loudspeakers 717, an I/O interface 713 for the keyboard 702 and mouse 703 and optionally a joystick (not illustrated), and an interface 708 for the modem 716 and printer 715. In some implementations, the modem 716 may be incorporated within the computer module 701, for example within the interface 708. A storage device 709 is provided and typically includes a hard disk drive 710 and a floppy disk drive 711. A CD-ROM drive 712 is typically provided as a non-volatile source of data. The components 705 to 713 of the computer module 701, typically communicate via an interconnected bus 704 and in a manner which results in a conventional mode of operation of the computer system 700 known to those in the relevant art.

Typically, the application program is resident on the hard disk drive 710 and read and controlled in its execution by the processor 705. Intermediate storage of the program and any data fetched from the network 720 may be accomplished using the semiconductor memory 706, possibly in concert with the hard disk drive 710. In some instances, the application program may be supplied to the user encoded on a CD-ROM or floppy disk and read via the corresponding drive 712 or 711, or alternatively may be read by the user from the network 720 via the modem device 716. Still further, the software can also be loaded into the computer system 700 from other computer readable media. The term "computer readable medium" as used herein refers to any storage or transmission medium that participates in providing instructions and/or data to the computer system 700 for execution and/or processing. Examples of storage media include floppy disks, magnetic tape, CD-ROM, a hard disk drive, a ROM or integrated circuit, a magneto-optical disk, or a computer readable card such as a PCMCIA card and the like, whether or not such devices are internal or external of the computer module 701. Examples of transmission media include radio or infra-red transmission channels as well as a network connection to another computer or networked device, and the Internet or Intranets including e-mail transmissions and information recorded on Websites and the like.

The method of converting colour gradients in images to a higher-level vector description may alternatively be implemented in dedicated hardware such as one or more integrated circuits. Such dedicated hardware may include graphic processors, digital signal processors, or one or more microprocessors and associated memories.

Converting colour gradients to vector description

Fig. 1 shows a method of generating a vector description of one or more colour gradients in a source image 12. The image 12 is described as an array of pixels arranged on a plurality of scanlines. Each pixel is described by a set of values, which may be the triplet (R,G,B), describing a red, blue and green value respectively for the pixel. The pixel may also have a fourth associated value indicative of opacity. Other colour spaces, for example, may also be used to describe the image 12. The image 12 could also be a greyscale image, in which each pixel is described by a single scalar luminance value.

Method 10 starts with step 100 (the detection phase) which aims to identify the areas in the source image that contain a smooth colour gradient. The detection phase 100 involves two sub-steps. Firstly, sub-step 101 detects colour slopes, using smoothed pixel differences. In sub-step 101, the image 12 is read a scanline at a time, and an array of integers is constructed that contains the differences between each pixel and an adjacent pixel on the same scanline. The difference between two pixels, each containing a red, green and blue component, is calculated using this formula:

$$\text{Diff} = \text{abs}(R_1 - R_2) + \text{abs}(G_1 - G_2) + \text{abs}(B_1 - B_2) \qquad \text{Equation (1)}$$

where (R_1, G_1, B_1) are component values of the first pixel and (R_2, G_2, B_2) are component values of the second pixel. The difference value returned is stored. After the scanline has been fully read, a smoothing function is applied to the array of differences. This is done by averaging a given difference value with *n* neighbouring differences on each side (where in one arrangement, *n* is 5) and then storing the average back into the given difference's location.

Areas of high difference, such as text, have high values after averaging and areas of flat colour have no difference. Areas of slowly changing colour, like a gradient, have a low difference value. So by applying a band pass filter to the difference values, sub-step 101 extracts a set of pixels that are on a colour gradient. Pixels that form areas of flat colour are preferably excluded from the set of possible gradient pixels, by detecting runs of *k* adjacent pixels with zero smoothed difference values, where *k* is a predetermined parameter (in one arrangement *k* is 5). Pixels having a smoothed difference value above a predetermined threshold are also removed from the set of possible gradient pixels.

When scanning in a horizontal direction across the scanline, sub-step 101 cannot detect a vertical linear gradient, as the colour does not vary across the scanline. To work around this, sub-step 101 also repeats the scanning vertically along each pixel column, adding more pixels to the set of pixels found in the 1st horizontal pass. The vertical differences are also calculated and smoothed before scanning for candidate gradient pixels. Thus the output of sub-step 101 is a set of pixels in the image 12 that appear to be on a colour gradient.

Not all the pixels in the colour gradient(s) are found this way so sub-step 102 is used to expand the set of gradient pixels to the edges of the gradient, filling all the gaps left by sub-step 101. Sub-step 102 uses a standard flood fill, or region growing, algorithm starting from each of the pixels output from sub-step 101. The boundary for the flood fill is found by using the difference function (Equation (1)) used in step 101 with a configurable threshold. If the difference between two neighbouring values (i.e. a pixel within the gradient set and a neighbouring pixel) is less than the configurable threshold, the neighbouring pixel is added to the set of gradient pixels.. That is, the flood fill algorithm uses the pixels detected in step 101 as seeds, and determines surrounding pixels that satisfy a similarity criterion. This stops the flood fill from continuing to expand the set of gradient pixels when the difference between a pixel inside the gradient set and a pixel on the edge of the flood fill is above the configurable threshold. Thus the output for sub-step 102 is a set of all pixels in the input image that form a colour gradient.

All steps after sub-step 102 only operate on the set of pixels output from sub-step 102; pixels not in that set are no longer considered. There may be multiple distinct areas of colour gradient in the input image, so sub-step 102 also keeps track of contiguous areas of colour gradient using a numbering scheme. One method of storing the sub-set of pixels accumulated by sub-step 102 is using a mask, which consists of a two-dimensional array of bytes having the same dimensions as the source image, each value in the mask storing the classification of the corresponding pixel in the source image. A value of 0 means that the source pixel is not part of a colour gradient. A non-zero value is used to indicate that the corresponding pixel in the source image is in a colour gradient, and the actual value in the mask can store an index such that contiguous areas of colour gradient have the same mask value. For example, if two colour gradients exist in the source image 12, all the mask values for the first gradient have the value 1 and all the mask values for pixels in the second gradient have the value 2.

After the source image is classified into contiguous areas of colour gradient in the detection phase 100, step 200 (the vectorise phase) of method 10 decides the parameters of the colour gradients and constructs a vector form of those gradients. The detection phase 100 and its sub-steps in some cases may not be required at all depending on the application. Those skilled in the art will recognise that other methods of classifying pixels into contiguous areas of colour gradient may be used in place of the described step 100.

Step 200 (the vectorise phase) is executed on each of the contiguous areas of colour gradient. Step 200 firstly needs to decide whether the gradient in question is a linear gradient or a radial gradient. This is the aim of sub-step 210 (the path generation step), and based on the output from sub-step 210, either sub-step 220 is executed in the case that the gradient is linear, or sub-step 230 is executed in the case that the gradient is radial. After either case 220, 230 sub-step 240 is executed to generate 'colour stops' at points of inflexion.

The path generation sub-step 210 includes four sub-steps, 211, 212, 213 and 214. The path generation sub-step 210 begins by executing sub-step 211, which finds a sub-set of pixels in the colour gradient area that are of very similar values. This is achieved by picking one of the pixels already in the gradient area and finding all similarly coloured pixels in the gradient, starting with exact matches and then increasing the threshold of difference between the starting pixel and the rest of the gradient until a suitable sub-set is found. This is illustrated in Fig. 2, which shows an area 312 containing a radial blend. Reference numeral 310 indicates an annular region of similar pixels. Thus, for example, sub-step 211 may identify the pixels in region 310.

The suitability of the current sub-set is defined by the number of pixels returned by sub-step 212. Sub-step 212 takes the current sub-set found in sub-step 211 and culls the current sub-set down to a smaller set such that each pixel in the smaller set is at least some minimum distance away from and not more than some maximum distance away from any other pixel in the smaller set. An example of this is shown in Fig. 2, where the set of pixels labelled 320 is an exemplary sub-set of the pixels in the annular region 310. Sub-step 212 has the effect of outputting a set of points that runs along a colour contour in the gradient. If not enough pixels are in this set at the end of sub-step 212 then sub-step 211's difference threshold is increased and execution resumes with sub-step 211. If there are enough pixels in the sub-set after sub-step 212 has executed then step 200 is allowed to continue to sub-step 213. In one arrangement the minimum number of pixels is 5, but a minimum of 8 pixels may also be used.

Sub-step 213 takes the sub-set of gradient points as an 'unsorted' array of points with (x,y) co-ordinates and sorts them such that the points form a polyline path. This is done by first scanning the array of points and finding end points, or pixels with only one neighbouring point. A neighbouring point is one that is within a certain distance, being defined by:

$$\text{Dist} = \text{sqrt}(\ (X_1 - X_2)^2 + (Y_1 - Y_2)^2\ )$$

where (X_1, Y_1) is the first point and (X_2, Y_2) is a possible neighbouring point.

One end point is taken out of the 'unsorted' array and put into a new 'sorted' array. Then the nearest remaining point is moved into the sorted array. After that the remaining points are sorted by their distance from the last point placed in the sorted array, and the nearest remaining point is moved from the unsorted array to the end of the sorted array.

If there are no end points identified, for example if an entire ring of similar pixels is present, then any point may be selected as a starting point for the sorting step 213.

Once the sorted point list is complete, sub-step 214 looks at the polyline segments examining the gradient of each line segment. If the line segment gradients are all similar within some configurable tolerance then the polyline is mostly straight, indicating a linear gradient. Otherwise it is assumed that the gradient is radial.

If sub-step 214 indicated that the gradient is linear then linear sub-step 220 (which includes sub-steps 221 and 222) is executed. Sub-step 221 determines the geometric parameters, i.e. the bounds of the linear gradient in gradient space. The bounds consist of two end points delimiting a line that is perpendicular to the colour contours of the gradient area, as illustrated with respect to Fig. 4, which shows a region 405 containing a linear colour blend. The position of the region 405 is defined in the image co-ordinate system with x-axis 407 and y-axis 406.

A contour of the linear blend in region 405 is identified in sub-step 211, and the points 403 along the contour are identified in sub-step 212. Line 402 is orthogonal to the colour contour. The contour and line 402 define co-ordinate axes in ‘gradient space’. Rectangle 401 is a bounding box of region 405, the sides of the bounding box being parallel to the axes of the gradient space. Point 408 is the beginning point of the linear gradient and point 404 is the ending point of the colour gradient..

Colour gradients specify their colour at discrete points, or ‘colour stops’ in unit parameter space between 0 and 1 where 0 is the parameter at the beginning point 408 and 1 is the parameter at the ending point 404. Any number of colour stops may be placed in the range 0 to 1 including the end points 0.0 and 1.0. A colour stop defines the colour at a given point in this unit, or normalised, parameter space. Colour varies at a constant rate (linear fashion) between two colour stops. To decide the location of the start and end point 408, 404 of the linear gradient, sub-step 221 needs to scan through all the pixels in the gradient area and transform the position of each pixel into gradient space. Gradient space is defined with one axis (the x-axis) along the colour contour lines and the other axis (the y-axis) along the contour's normal line marked as 402 in Fig 4. The normal line's angle is calculated by taking the average gradient of the polyline segments created from the points labelled 403 and calculating the perpendicular angle as arctan(1 / average_gradient). The angle α between the gradient normal 402 and the image's y-axis 406 is calculated and used to rotate each gradient pixel's co-ordinates such that the colour contours are horizontal (i.e. the normal 402 is vertical). The maximum and minimum of all the rotated co-ordinates' x and y values are calculated and these form a rectangular bounding box 401 for the gradient area as defined in gradient space. Two points 408, 404 on the left edge of the gradient's bounding box 401, at the bottom-left corner and the top-left corner respectively, are transformed from gradient space back into the image's co-ordinate space by reversing the rotation applied earlier and these points 408, 404 form the start and end points of the linear gradient, thus completing sub-step 221.

Sub-step 222 constructs a colour ramp array with a configurable number of buckets (in one arrangement 1024 buckets are used), each of which contains three accumulated sums of Red, Green and Blue components respectively and a count of pixels. This array of buckets will be referred to as the colour ramp array. The purpose of the colour ramp array is to collect an average colour value at a corresponding points on a notional vertical axis 402 in the gradient unit parameter space.

Sub-step 222 scans through all the pixels in the gradient area and maps the co-ordinates of each pixel into the gradient unit parameter space by using the gradient space bounding box 401 calculated in sub-step 221. Given the angle between the image space vertical (y) axis 406 and the gradient space axis 402 normal to the colour contours is α and the co-ordinates of the pixel in the gradient area are (X_1, Y_1) then the point's coordinates in gradient space are given by the following image to linear-gradient-space transform:

$$X_G = \cos(\alpha).X_I - \sin(\alpha).Y_I$$

$$Y_G = \sin(\alpha).X_I + \cos(\alpha).Y_I$$

Given the bounding box 401 in gradient space calculated in sub-step 221 is delimited by (X_{min}, Y_{min}, X_{max}, Y_{max}) the pixel's position P in the gradient unit parameter space is:

$$P = (Y_G - Y_{min}) / (Y_{max} - Y_{min})$$

Where $P >= 0.0$ and $P <= 1.0$

So by calculating P for every pixel in the gradient area, sub-step 222 determines a corresponding bucket for the pixel in the colour ramp array. The sub-step 222 of adding the pixel to the colour ramp array simply adds the Red, Green and Blue components to the respective accumulators in the bucket and increments the bucket's pixel count.

Execution then passes to sub-step 240. In the case that sub-step 210 decides that the gradient is a radial gradient, sub-step 230 is executed instead of sub-step 220.

Sub-step 230's purpose is to decide the geometric parameters of the radial gradient. These parameters consist of a centre point and a radius. The radial sub-step 230 includes sub-steps 231, 232 and 233. The determination of the radial parameters is illustrated with reference to Fig. 5, which shows a region 505 containing a radial blend.

Substep 231 calculates the centre of the gradient by starting with the polyline calculated in sub-step 213, which forms a line along a colour contour in the gradient. In this case the contour is a circular arc or a full circle. A subset of the points defining the polyline (e.g. the points marked 501 in Fig. 5), forming less than half a circle, is taken by comparing gradients of each line segment in the polyline until the angle has changed through more than 90° and less than 180°. A line 502 is formed between the two end points of this subset of points 501 and the mid point of the line 502 is calculated. A new line 503 is formed from the midpoint of line 502 along the normal to line 502 and extended away from the existing points 501 in the polyline subset. This line 503 should come close to intersecting with the centre of the circle defining the radial blend, however the exact co-ordinates of the centre are not yet known. By iteratively advancing a point (some examples of this point's movement are marked 504 in Fig. 5) along this normal line 503 and calculating the distances from point 504 to the polyline's points until such distances are almost equal, sub-step 231 can find a fairly accurate approximation of centre of the radial gradient. This method is only one such possible method and those skilled in the art will understand that other methods could be used as sub-step 231.

Once the centre is known, sub-step 232 scans through all the pixels in the gradient area, calculating their distance from the centre, and retaining a maximum. This maximum becomes the radius of the radial gradient.

Given the centre and radius, sub-step 233 proceeds to generate the colour ramp array. This is similar to sub-step 222 for linear gradients. Each pixel in the gradient area is scanned again and the distance between the pixel's co-ordinates and the centre is calculated, and that distance value is transformed into a gradient unit parameter space value between 0 and 1 using the radius. Given a gradient pixel co-ordinate (X_1, Y_1) the distance to the centre (X_c, Y_c) is:

$$\text{Distance} = \text{sqrt}(\ (X_c - X_1)^2 + (Y_c - Y_1)^2\ )$$

The distance is mapped into the gradient unit parameter space using the image to radial gradient space transform:

P = Distance / Radius.

This position is mapped into an array of colour buckets, similar to sub-step 222. The colour components of the gradient pixel are added to the bucket's accumulators and the bucket's count is incremented.

Once sub-step 220 or 230 has completed execution passes on to sub-step 240 (the stop generation sub-step), which takes the ramp array of colour buckets from sub-step 222 or 233 and builds a list of colour stops in the gradient unit parameter space (0 -> 1).

Sub-step 241 takes the array of colour buckets and marks any entries that have too small a pixel count (indicating that the corresponding bucket does not contain enough source image pixels to get a good average) to be ignored in all further processing. A configurable lower limit on the pixel count is preferably set to 3 pixels. Thus, the unused entries are ignored for subsequent processing. At this point each of the colour component accumulators is divided by the number of pixels contributing to the component to get an average value for that component across all the pixels that contributed to that bucket.

The remaining colour buckets tend to have some noise in the colour components due to stray pixels, scanning artefacts and the effects of anti-aliasing. Sub-step 242 takes the colour ramp array and runs a smoothing function over the entries to remove particularly bad spikes in the data. The smoothing operation can be implemented by any number of suitable algorithms. One such algorithm is to look at each colour bucket component and compare the current bucket to the previous and next buckets. If the value of the current bucket is far enough outside the value of the neighbouring buckets then an average of the neighbouring buckets is used instead.

Sub-step 243 then takes the smoothed colour ramp array and creates a list of colour stops that describes the colour ramp at discrete points in the gradient unit parameter space such that, when rendered, the colour stops, with linear interpolation in between, form a piecewise linear functional approximation of the full colour gradient. This sub-step scans the colour ramp array from start to finish and emits stops when the gradient of any of the components changes significantly, i.e. whenever an inflexion point is identified. This calculation is done by moving a "current" pointer through the colour ramp array as well as maintaining a "last" pointer. The last pointer starts at the beginning of the array and is updated to the current position when a colour stop is emitted. The colour ramp can be visualized in a two dimensional plane, where the x-axis is the gradient unit parameter space (where the first colour ramp entry is at x=0 and the last colour ramp entry is at x=1), and the y-axis has the colour component's value for that colour ramp bucket. Each component traces out a rough line across this plane. Sub-step 243 converts the line for each colour component, made of discrete samples, into a vector polyline. The current pointer is advanced from x=0, checking that all the component values of each bucket between the current pointer and the last pointer fall on a straight line, or near enough to within a configurable tolerance. This is done for each colour component individually each time the current pointer is advanced. If any of the colour components between the 2 pointers are outside the line then a colour stop is emitted and the last pointer is moved to the location of the new colour stop. This sub-step 243 is continued until the end of the colour ramp is reached.

The vectorisation phase 200 ends with the output of vector gradient information 14, which includes either two end points for a linear gradient or a centre point and a radius for a radial gradient. In addition, the vector gradient information 14 includes a list of colour stops, each having a normalised position coordinate and a colour value.

200521l665 23 Sep 2005

The benefits of the described arrangements include lower memory usage, faster rendering time and being able to quickly change the gradient parameter of the output vector data.

Industrial Applicability

It is apparent from the above that the arrangements described are applicable to the computer and data processing industries.

The foregoing describes only some embodiments of the present invention, and modifications and/or changes can be made thereto without departing from the scope and spirit of the invention, the embodiments being illustrative and not restrictive.

In the context of this specification, the word “comprising” means “including principally but not necessarily solely” or “having” or “including”, and not “consisting only of”. Variations of the word "comprising", such as “comprise” and “comprises” have correspondingly varied meanings.

The claims defining the invention are as follows:

1. A method of generating a vector description of a colour gradient in an image, said method comprising the steps of:

receiving the image comprising an array of pixels, each pixel having one or more associated colour values, the image having at least one gradient area in which at least one of the colour values varies between neighbouring pixels;

generating, in each gradient area, a path comprising pixels having substantially the same colour;

determining one or more control points defining a line substantially perpendicular to the path;

calculating representative colour values at points along the line; and

generating a piecewise-linear colour function from the representative colour values,

wherein the one or more control points and the piecewise-linear colour function form a vector description of the colour gradient in the gradient area.

2. A method according to claim 1, wherein the path is substantially linear and the control points are the extremities of the perpendicular line within the gradient area.

3. A method according to claim 1, wherein the path is an arc of a circle and the control points are a centre of the circle and a maximum length of a radial line from the centre to a boundary of the gradient area.

2005211665 23 Sep 2005

4. A method according to claim 2 wherein the representative colour value at a point along the line is an average of the corresponding colour values of the pixels in the gradient area lying on a contour that is parallel to the path and intersects the perpendicular line at the point.

5. A method according to claim 3 wherein the representative colour value at a point along the line is an average of the corresponding colour values of the pixels in the gradient area lying on a contour that is the same distance from the centre as the point.

6. A method according to claim 1 wherein the piecewise-linear colour function is defined by one or more stops and wherein said step of generating the colour function considers the points in sequence and, for a currently-considered point, said generating step comprises the substeps of:

predicting one or more colour values at the currently-considered point based on a linear model of colour values at previously-considered points; and

adding a stop to the colour function if a difference between the one or more predicted colour values and corresponding representative colour values exceeds a predetermined threshold,

wherein the stop comprises a position and colour of the currently-considered point.

7. A method according to claim 6 further comprising the step of smoothing the representative colour values prior to generating the piecewise-linear function.

8. A method according to claim 1 wherein said step of generating the path comprises the substeps of:

picking a seed pixel in the gradient area;

identifying pixels in the gradient area that have a similar colour to the seed pixel according to a similarity measure dependent on the associated colour values and a predetermined threshold;

finding a subset of identified pixels such that pixels in the subset are at least a predetermined minimum distance from other pixels in the subset and are less than a predetermined maximum distance from other pixels in the subset; and

sorting the subset of pixels to form the path.

9. A method according to claim 8 comprising the further step of changing the predetermined threshold and repeating said identifying, finding and sorting substeps if there are insufficient pixels in the path.

10. A method according to claim 1 wherein the path lies on an arc of a circle and wherein said step of determining control points comprises the substeps of:

identifying a subset of pixels in the path, the subset bounded by two end pixels and covering less than half the circle;

calculating a candidate radius that is a perpendicular bisector of a line joining the two end points; and

determining a candidate centre point lying on the candidate radius and substantially equidistant from the pixels in the subset, wherein the candidate centre point is one of the control points forming the vector description.

11. A method according to claim 1, further comprising the step of detecting the at least one gradient area.

12. A method according to claim 11, wherein said detecting step comprises the substeps of:

calculating colour differences between neighbouring pixels in the image;

identifying seed pixels based on the colour differences; and

applying a flood fill to the seed pixels, wherein the at least one gradient area is a result of the flood fill.

13. An apparatus for generating a vector description of a colour gradient in an image, said apparatus comprising:

means for receiving the image comprising an array of pixels, each pixel having one or more associated colour values, the image having at least one gradient area in which at least one of the colour values varies between neighbouring pixels;

means for generating, in each gradient area, a path comprising pixels having substantially the same colour;

means for determining one or more control points defining a line substantially perpendicular to the path;

means for calculating representative colour values at points along the line; and

means for generating a piecewise-linear colour function from the representative colour values, wherein the one or more control points and the piecewise-linear colour function form a vector description of the colour gradient in the gradient area.

14. A computer program product comprising machine-readable program code recorded on a machine-readable recording medium, for controlling the operation of a data processing machine on which the program code executes to perform a method of generating a vector description of a colour gradient in an image, said method comprising the steps of:

receiving the image comprising an array of pixels, each pixel having one or more associated colour values, the image having at least one gradient area in which at least one of the colour values varies between neighbouring pixels;

generating, in each gradient area, a path comprising pixels having substantially the same colour;

determining one or more control points defining a line substantially perpendicular to the path;

calculating representative colour values at points along the line; and

generating a piecewise-linear colour function from the representative colour values,

wherein the one or more control points and the piecewise-linear colour function form a vector description of the colour gradient in the gradient area.

15. A computer program comprising machine-readable program code for controlling the operation of a data processing apparatus on which the program code executes to

perform a method of generating a vector description of a colour gradient in an image, said computer program comprising:

code for receiving the image comprising an array of pixels, each pixel having one or more associated colour values, the image having at least one gradient area in which at least one of the colour values varies between neighbouring pixels;

code for generating, in each gradient area, a path comprising pixels having substantially the same colour;

code for determining one or more control points defining a line substantially perpendicular to the path;

code for calculating representative colour values at points along the line; and

code for generating a piecewise-linear colour function from the representative colour values,

wherein the one or more control points and the piecewise-linear colour function form a vector description of the colour gradient in the gradient area.

16. A method of generating a vector description of a colour gradient in an image substantially as described herein with reference to any one of the embodiments as illustrated in the accompanying drawings.

17. An apparatus for generating a vector description of a colour gradient in an image substantially as described herein with reference to any one of the embodiments as illustrated in the accompanying drawings.

18. A computer program product substantially as described herein with reference to any one of the embodiments as illustrated in the accompanying drawings.

19. A computer program substantially as described herein with reference to any one of the embodiments as illustrated in the accompanying drawings.

DATED this Twenty-Third Day of September 2005
CANON KABUSHIKI KAISHA
Patent Attorneys for the Applicant
SPRUSON&FERGUSON

2005211665 23 Sep 2005

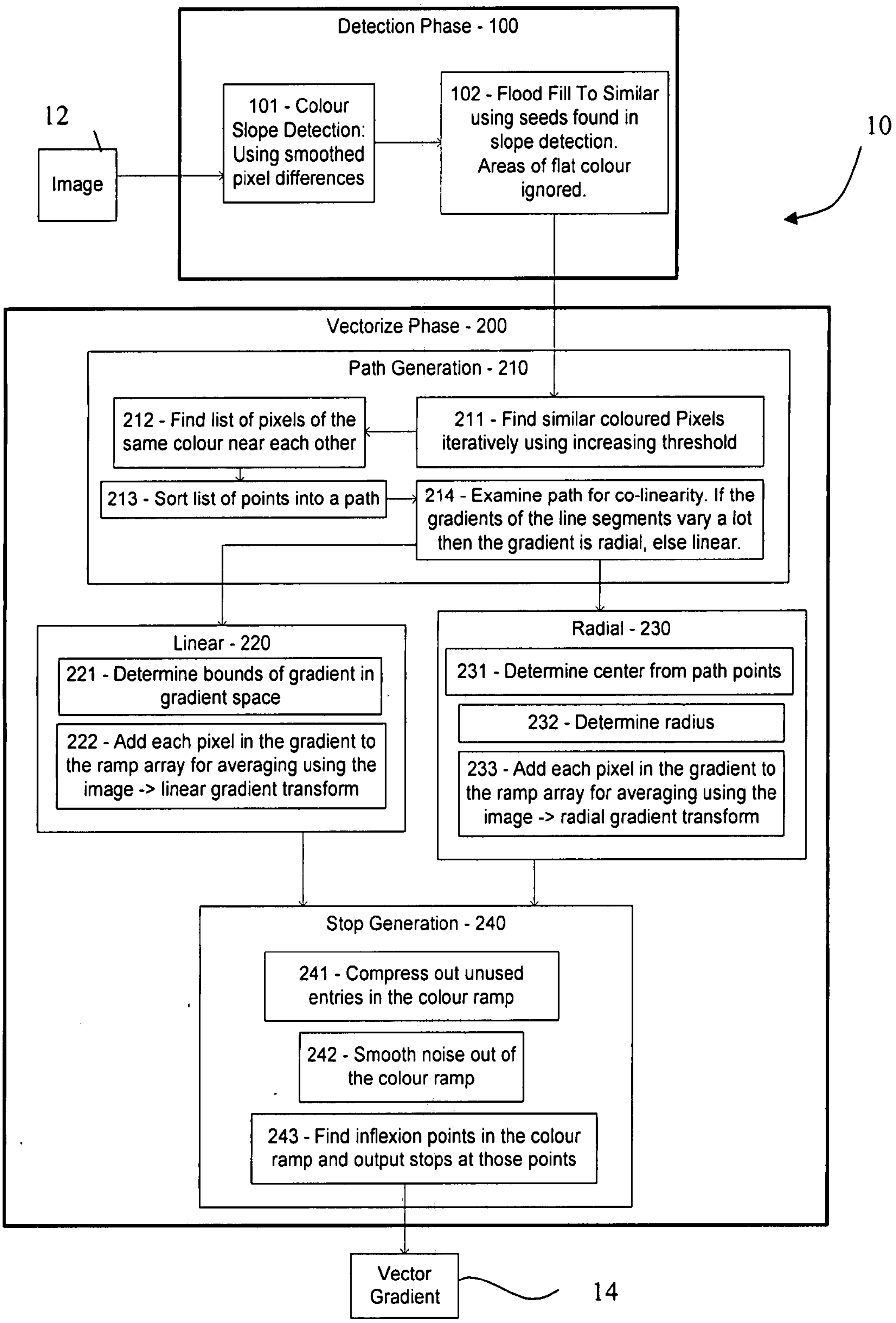
Detection Phase - 100
101 - Colour Slope Detection: Using smoothed pixel differences
102 - Flood Fill To Similar using seeds found in slope detection. Areas of flat colour ignored.
12
Image
10
Vectorize Phase - 200
Path Generation - 210
212 - Find list of pixels of the same colour near each other
211 - Find similar coloured Pixels iteratively using increasing threshold
213 - Sort list of points into a path
214 - Examine path for co-linearity. If the gradients of the line segments vary a lot then the gradient is radial, else linear.
Linear - 220
221 - Determine bounds of gradient in gradient space
222 - Add each pixel in the gradient to the ramp array for averaging using the image -> linear gradient transform
Radial - 230
231 - Determine center from path points
232 - Determine radius
233 - Add each pixel in the gradient to the ramp array for averaging using the image -> radial gradient transform
Stop Generation - 240
241 - Compress out unused entries in the colour ramp
242 - Smooth noise out of the colour ramp
243 - Find inflexion points in the colour ramp and output stops at those points
Vector Gradient
14

Fig. 1

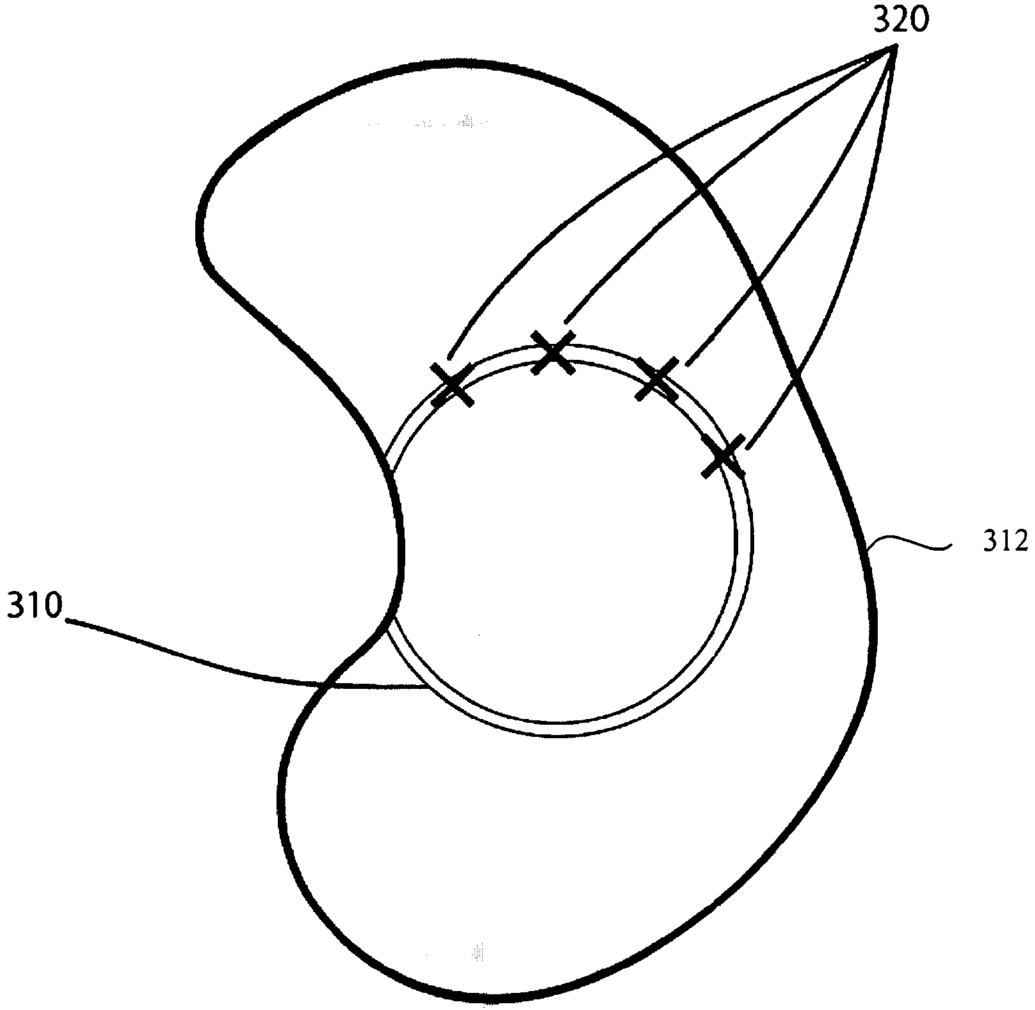

Fig. 2

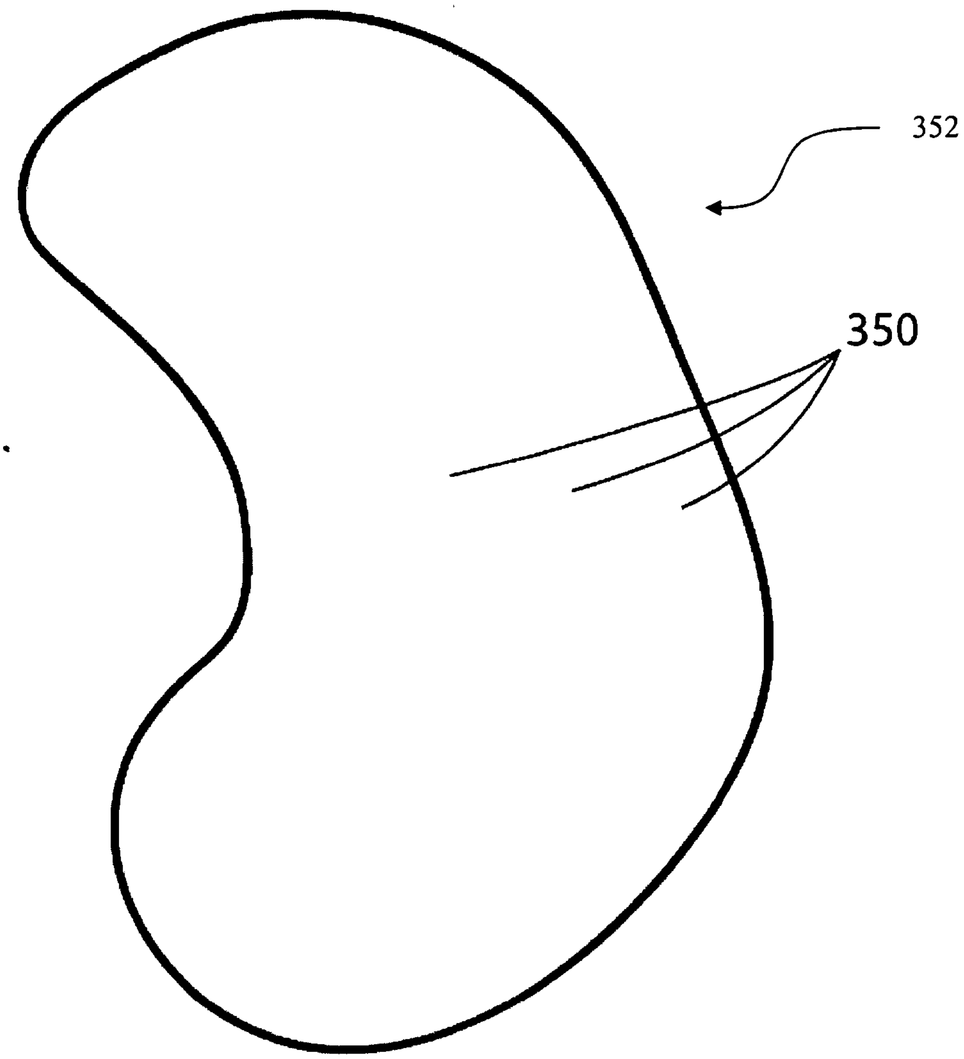

Fig. 3

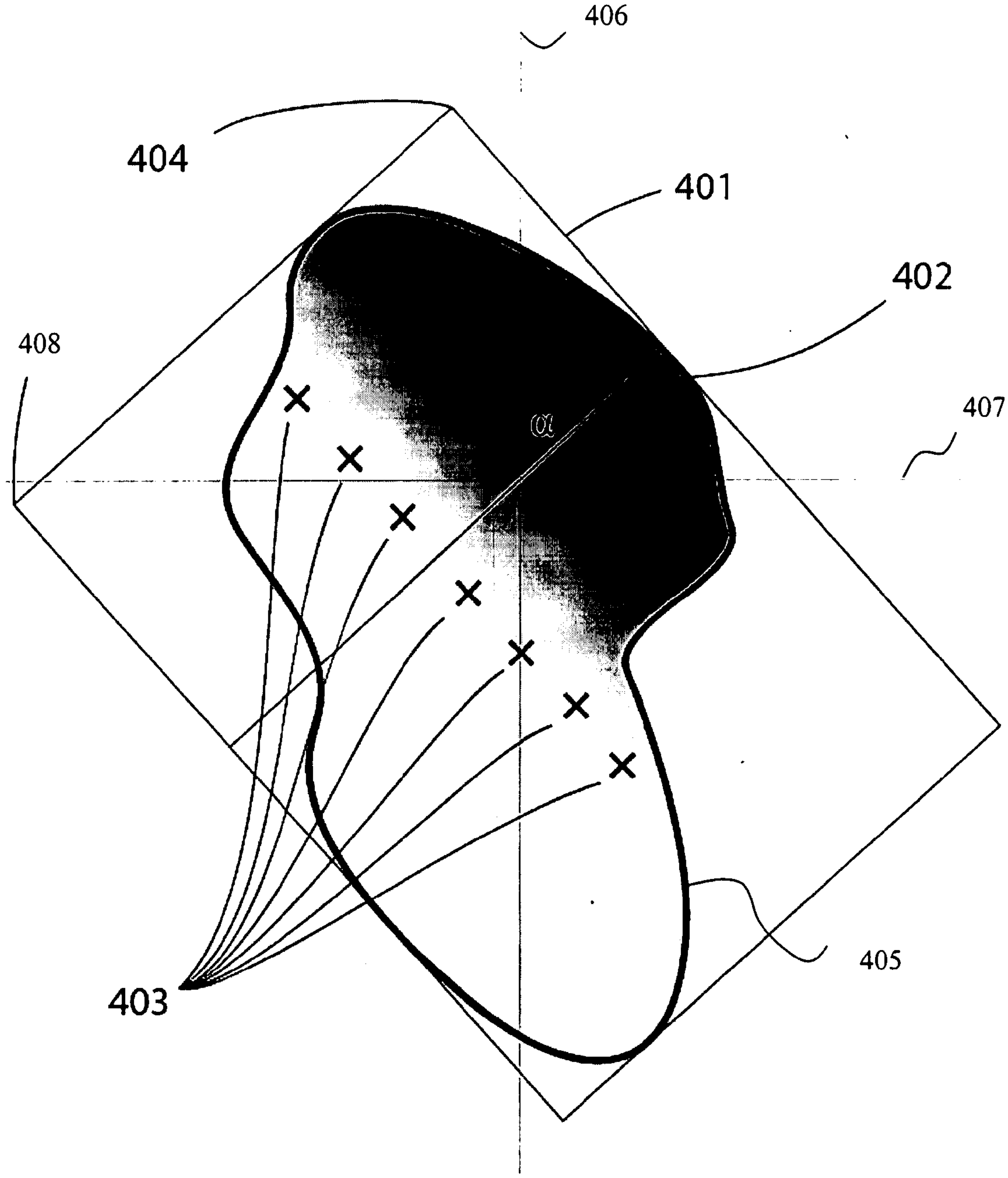

Fig. 4

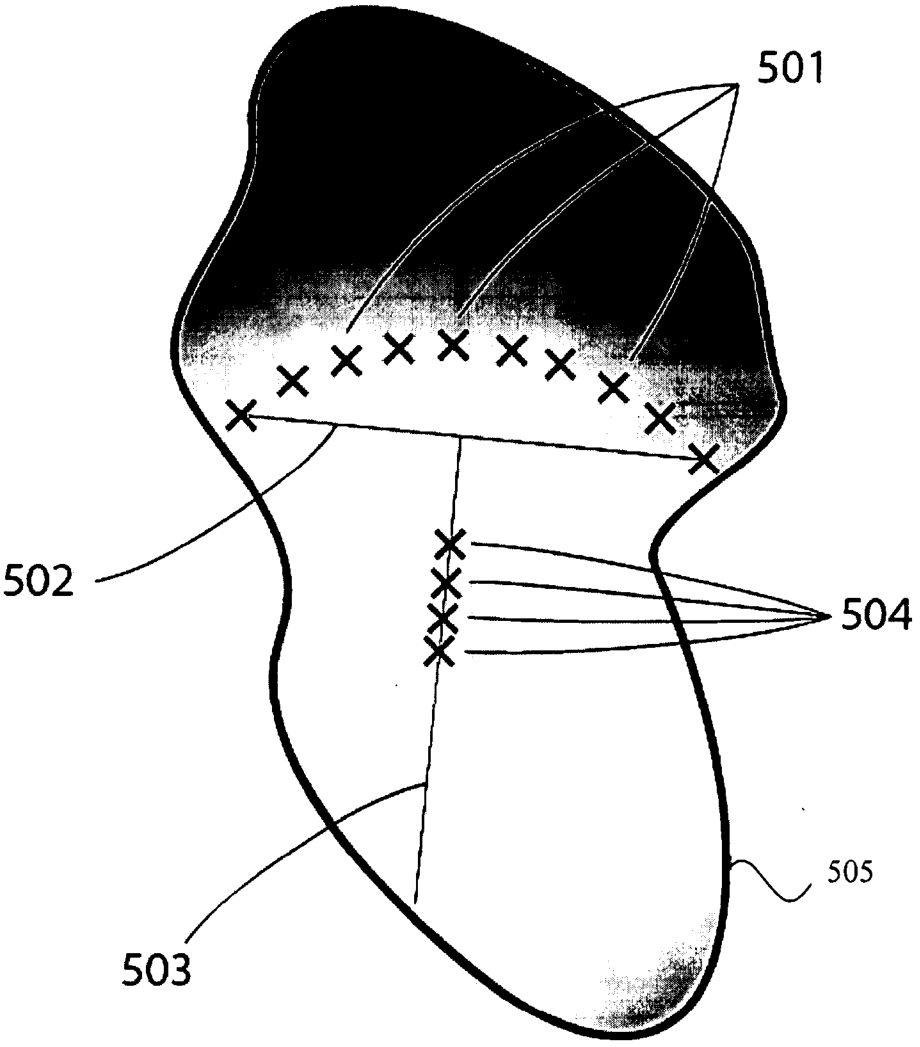

Fig. 5

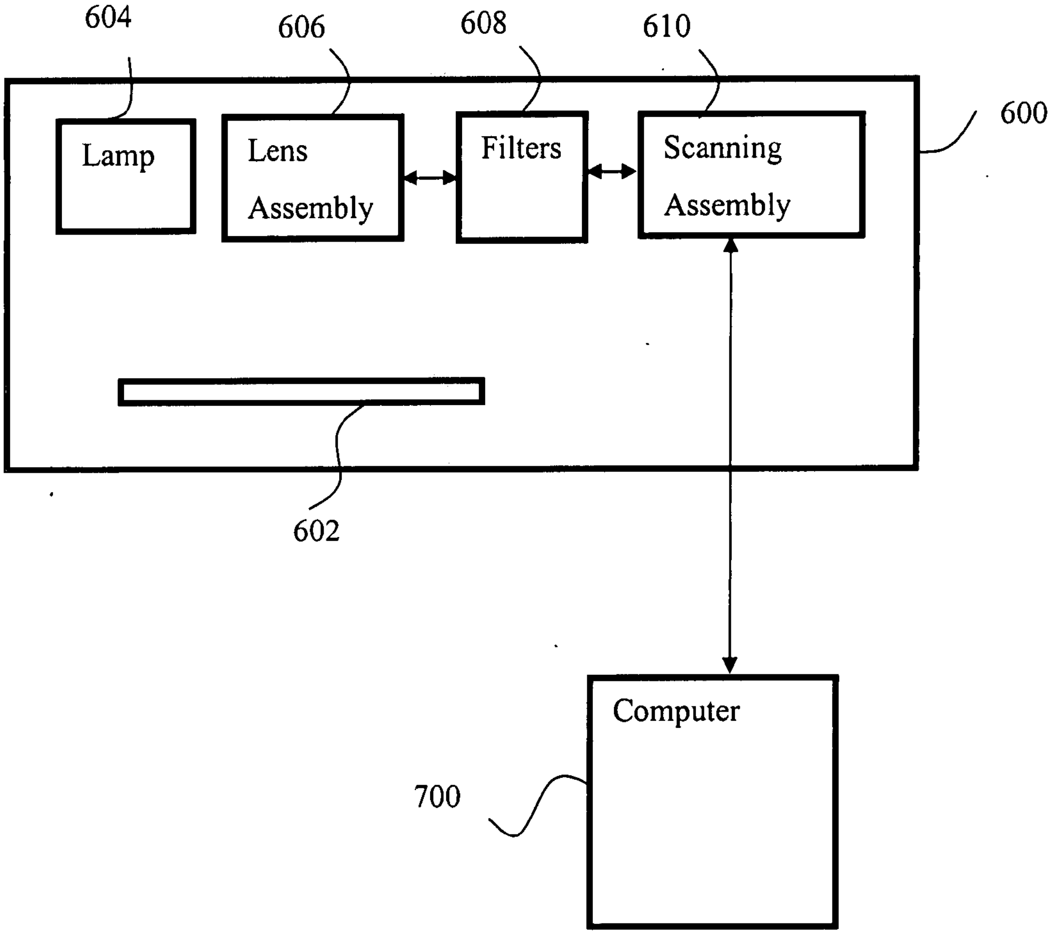
604
606
608
610
600
Lamp
Lens Assembly
Filters
Scanning Assembly
602
Computer
700

Fig. 6

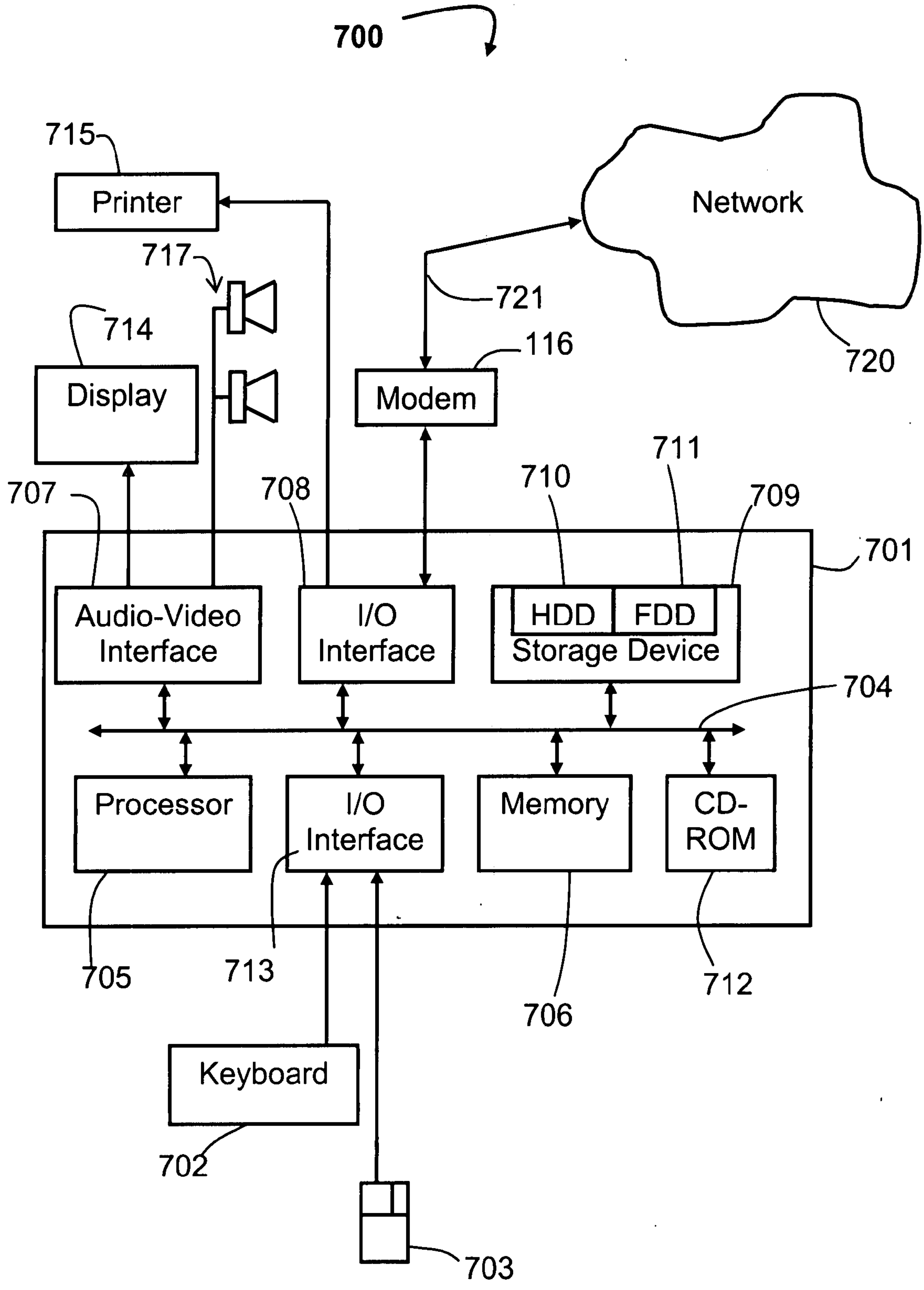
700
715
Printer
717
714
Display
721
Network
720
Modem
116
707
708
710
711
709
701
Audio-Video Interface
I/O Interface
HDD
FDD
Storage Device
704
Processor
I/O Interface
Memory
CD-ROM
705
713
706
712
Keyboard
702
703

FIG. 7