US 20150023595A1

(19) United States

(12) Patent Application Publication

(10) Pub. No.: US 2015/0023595 A1

(43) Pub. Date: Jan. 22, 2015

Perry et al.

(54) METHOD FOR RENDERING PATHS WITHOUT OUTLINING ARTIFACTS

(71) Applicant: **Mitsubishi Electric Research Laboratories, Inc.**, Cambridge, MA (US)

(72) Inventors: **Ronald N Perry**, Cambridge, MA (US); **Elena J Jakubiak**, Arlington, MA (US)

(21) Appl. No.: **13/942,837**

(22) Filed: **Jul. 16, 2013**

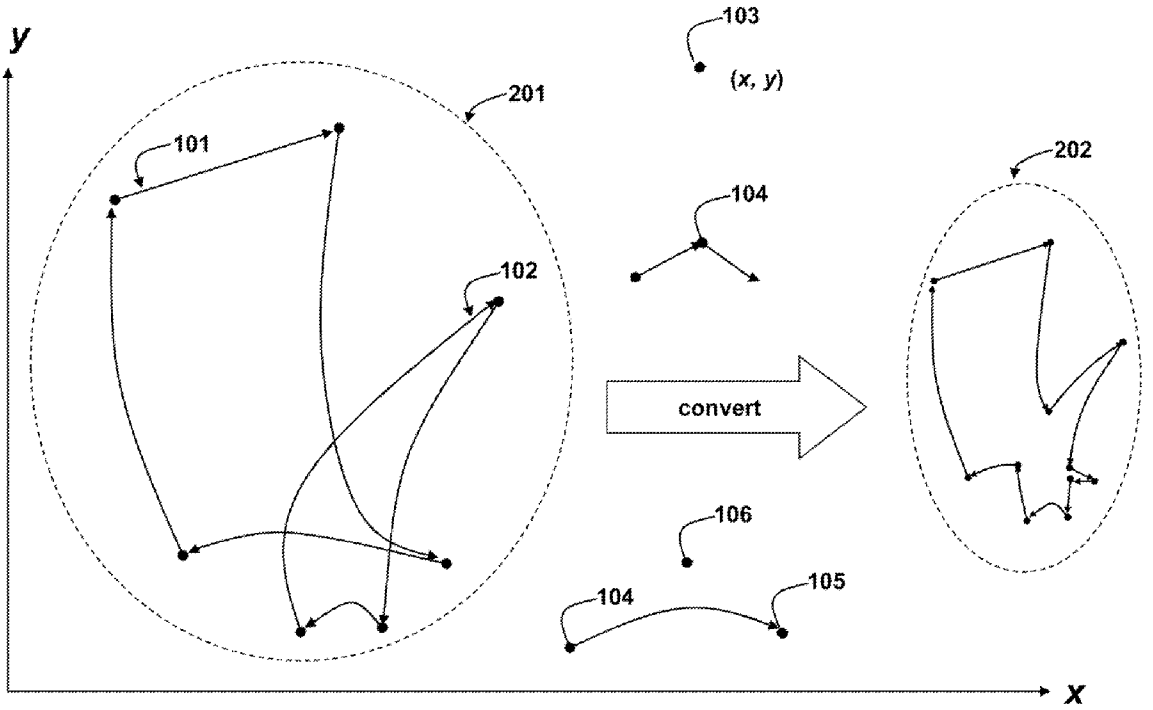
Publication Classification

(51) **Int. Cl.**
G06T 7/00 (2006.01)

(52) **U.S. Cl.**
CPC ***G06T 7/0089*** (2013.01)
USPC **382/173**

(57) **ABSTRACT**

A method for outlining a two-dimensional input path defined according to a nonzero winding rule is described. Degenerate segments and degenerate contours of the input path are removed. Intersections of the input path are determined. Contours of the input path that include intersections are marked. Unmarked interior contours are removed. Intersections are linked. The marked contours are walked to form new contours. Marked contours and degenerate contours are removed. The new contours and the unmarked contours are collected to form an equivalent output path. The segments of the equivalent output path are outlined.



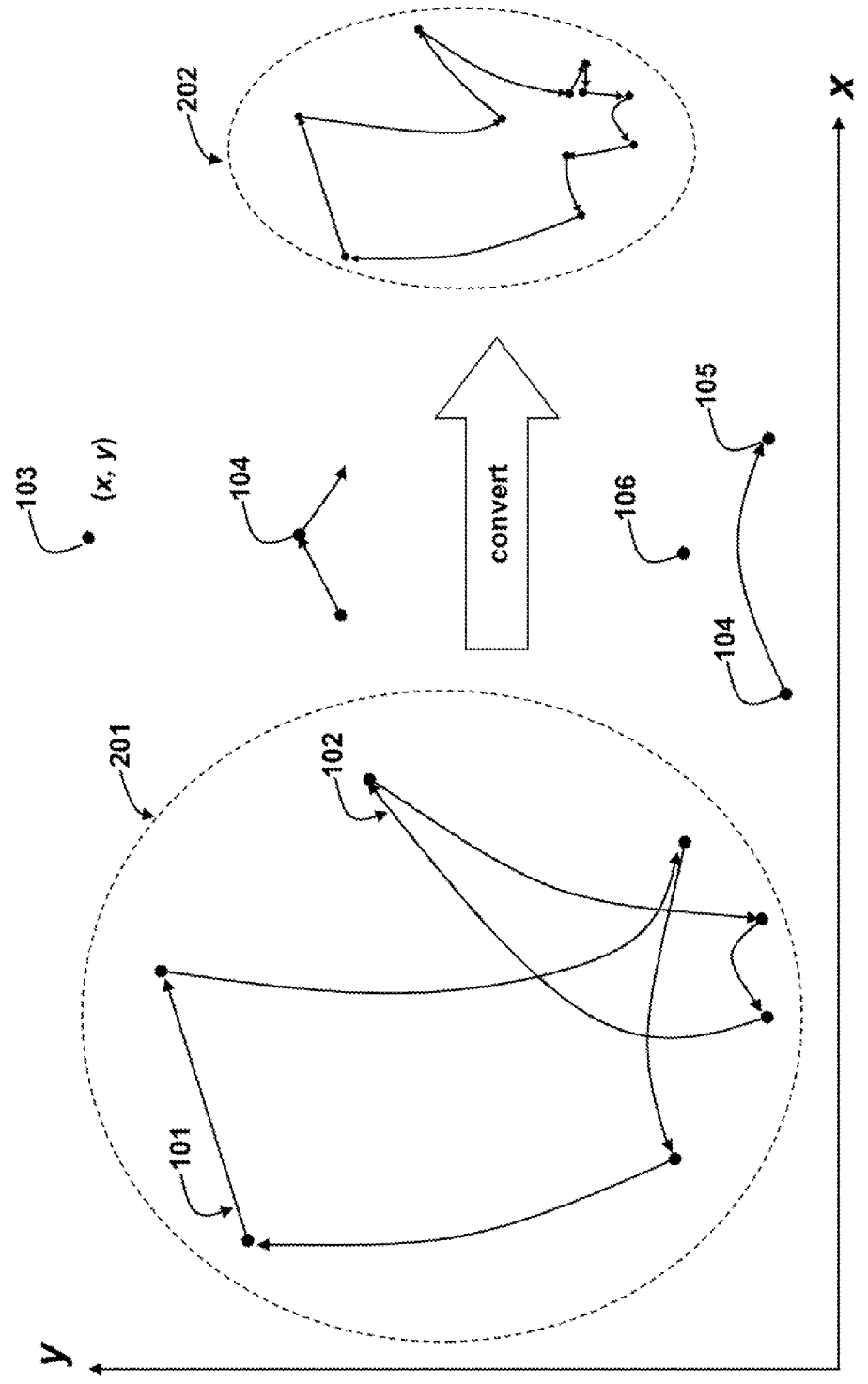


Fig. 1

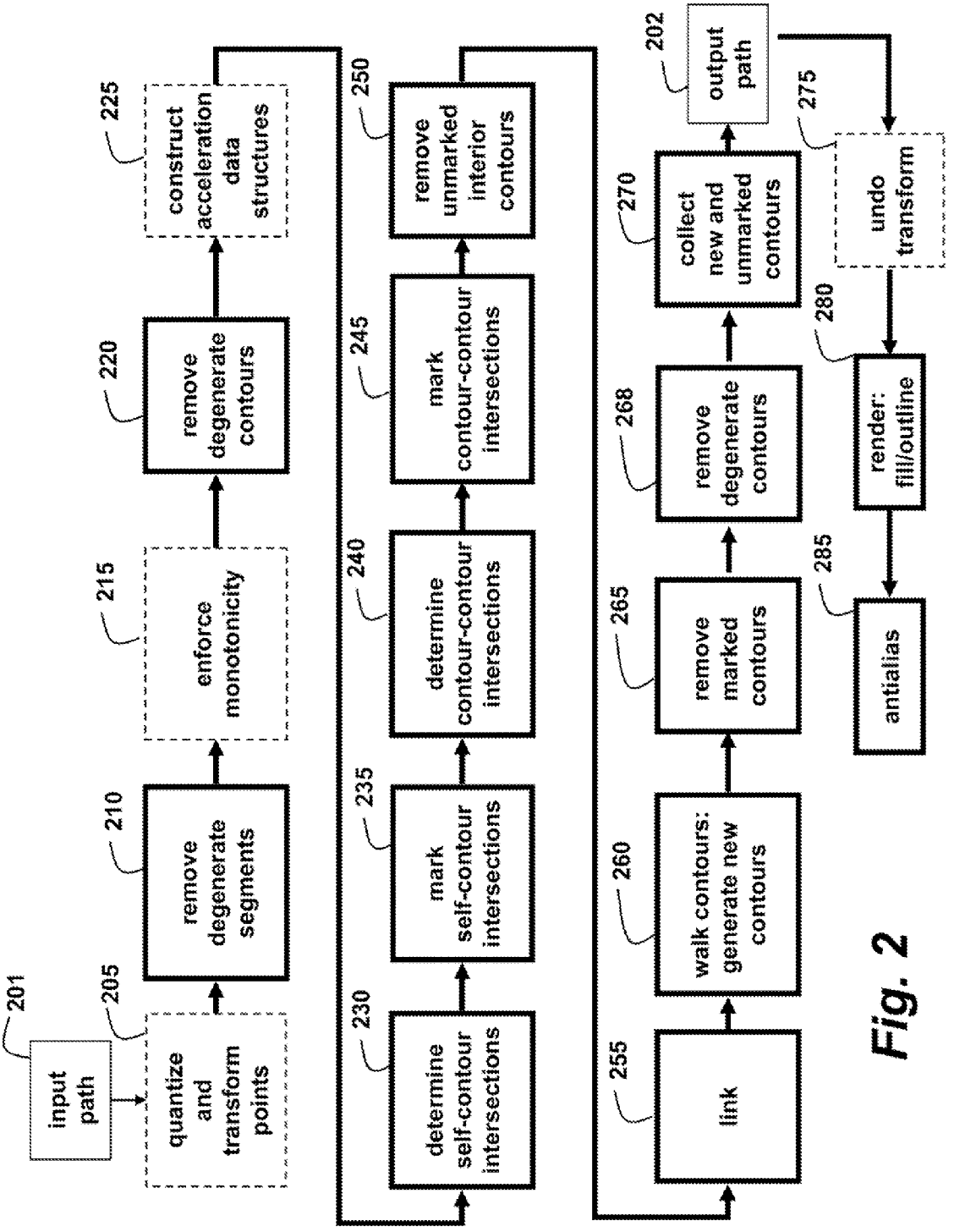

201
input path
205
quantize and transform points
210
remove degenerate segments
215
enforce monotonicity
220
remove degenerate contours
225
construct acceleration data structures
230
determine self-contour intersections
235
mark self-contour intersections
240
determine contour-contour intersections
245
mark contour-contour intersections
250
remove unmarked interior contours
255
link
260
walk contours: generate new contours
265
remove marked contours
268
remove degenerate contours
270
collect new and unmarked contours
202
output path
275
undo transform
280
render: fill/outline
285
antialias

Fig. 2

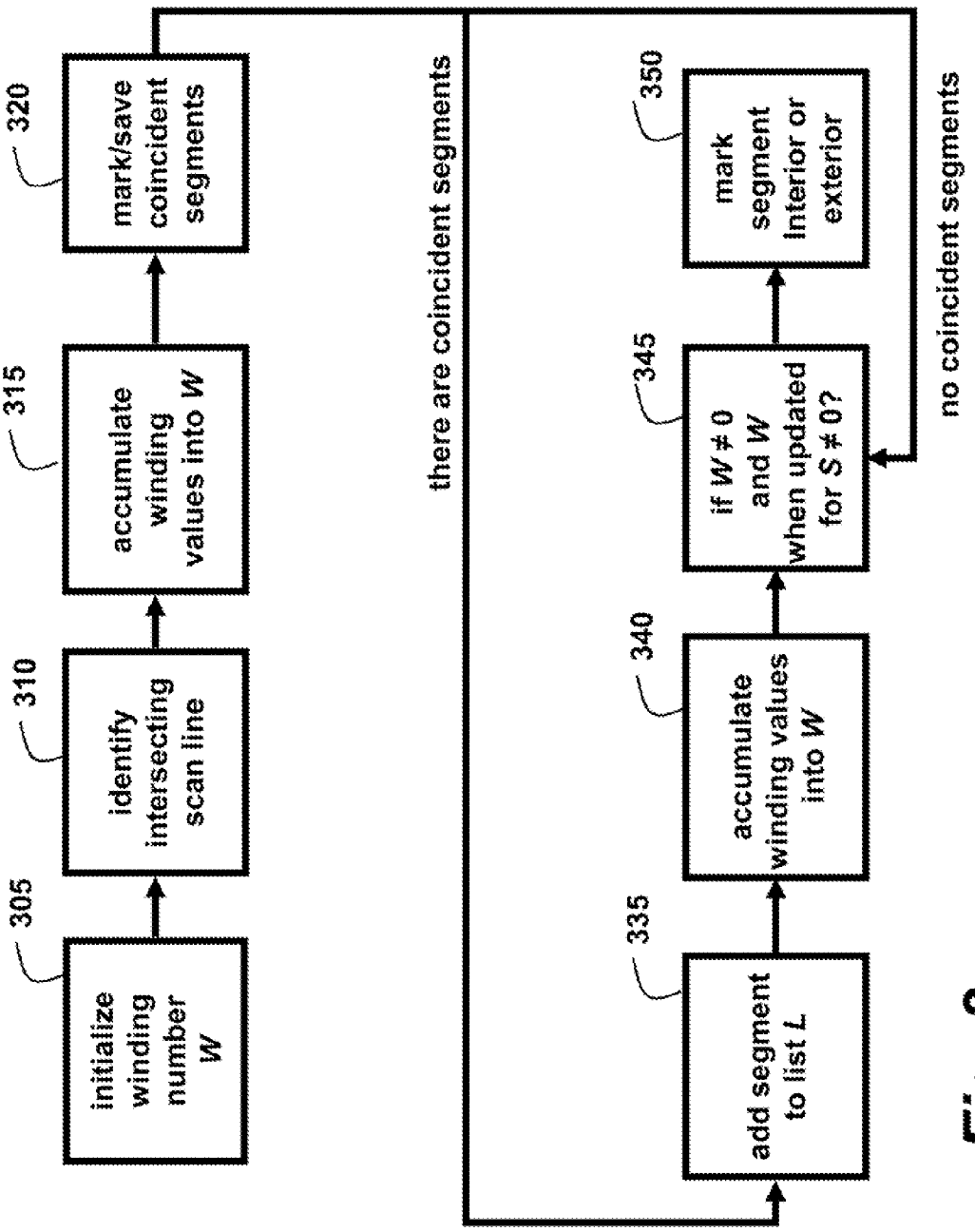

305
initialize winding number W
310
identify intersecting scan line
315
accumulate winding values into W
320
mark/save coincident segments
there are coincident segments
335
add segment to list L
340
accumulate winding values into W
345
if W ≠ 0 and W when updated for S ≠ 0?
350
mark segment interior or exterior
no coincident segments

Fig. 3

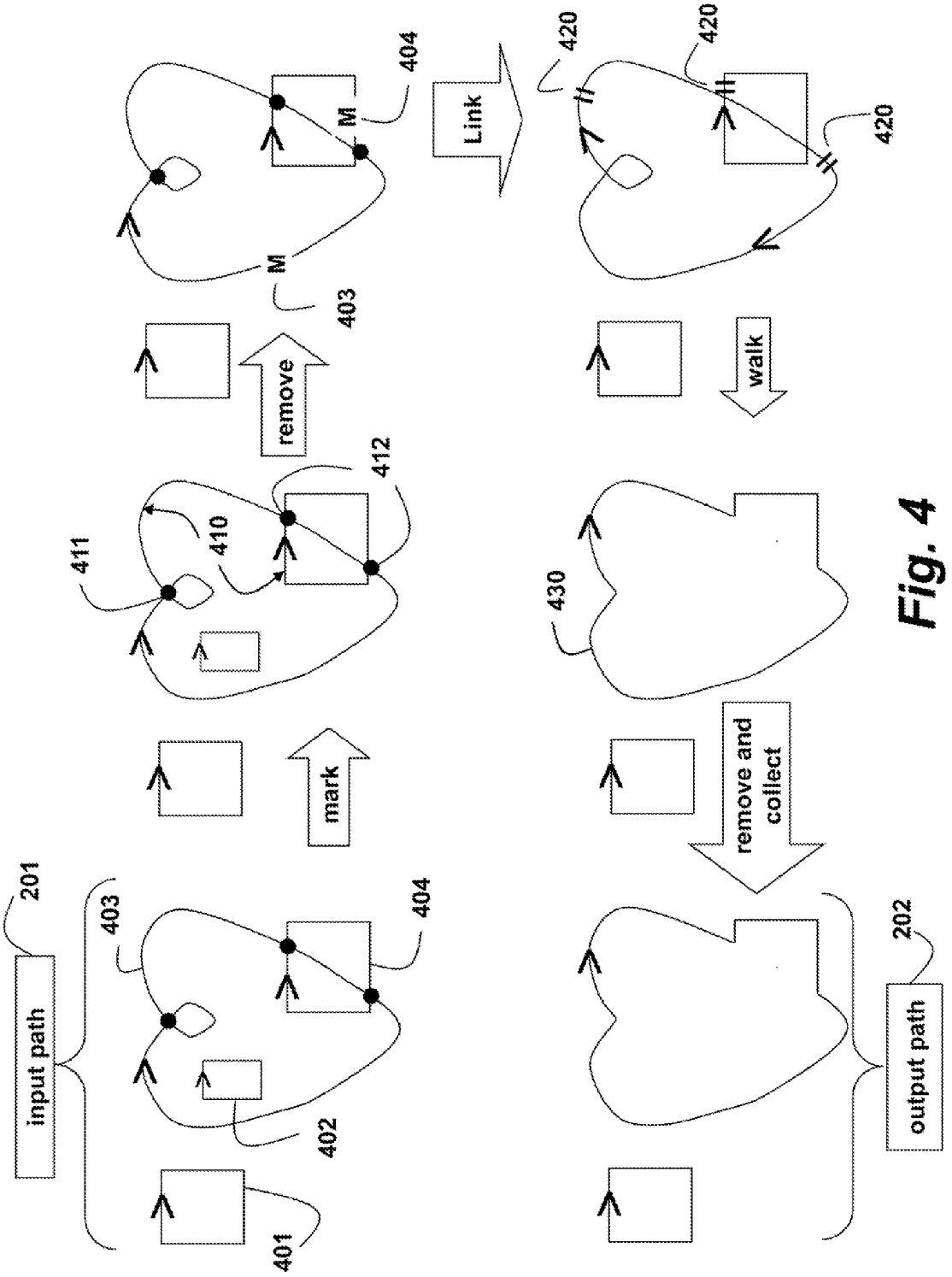

input path
201
401
402
403
404
mark
411
410
412
remove
M
M
403
404
Link
420
420
420
walk
430
remove and collect
output path
202

Fig. 4

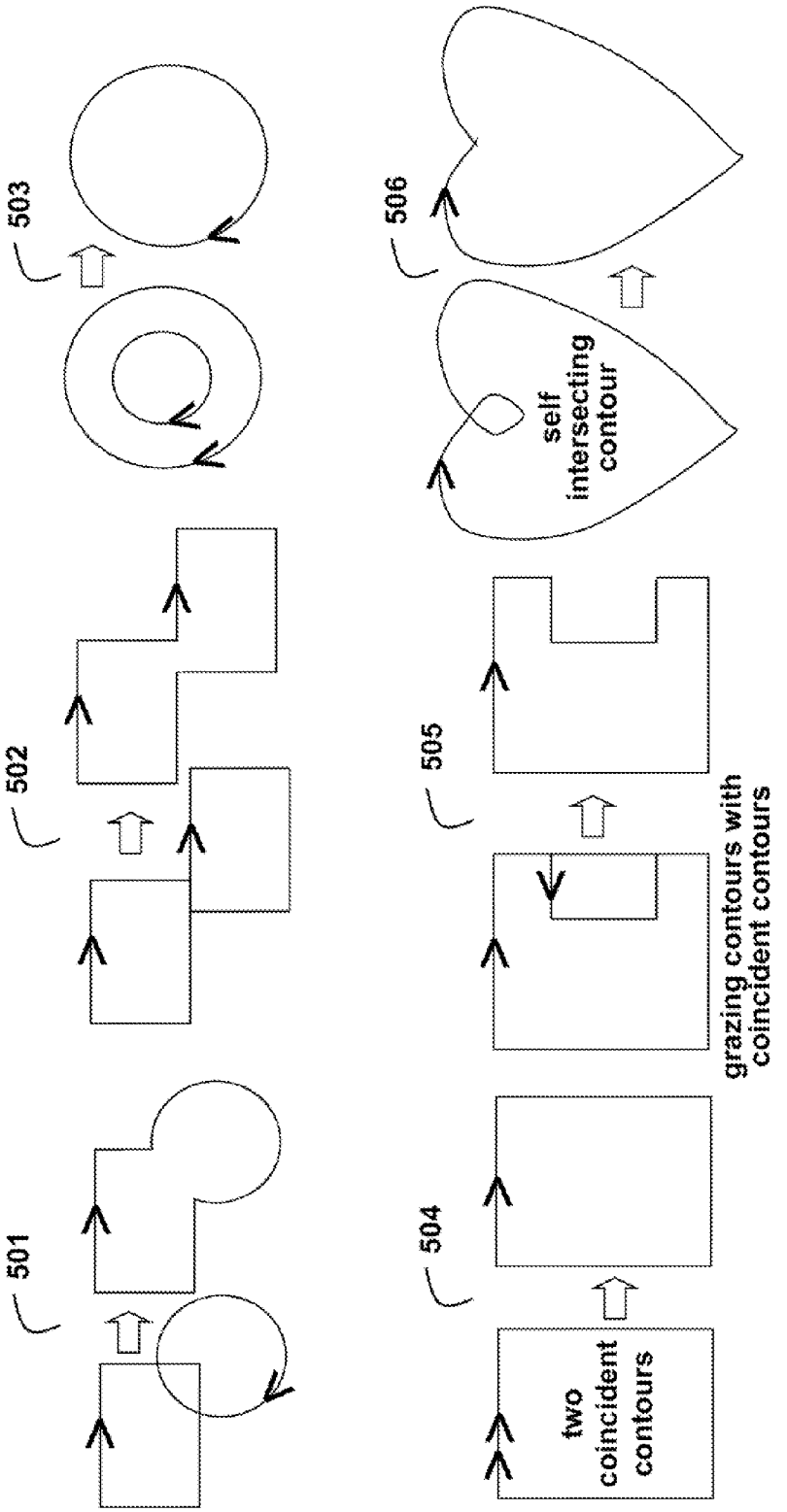

501
502
503
504
505
506
two coincident contours
grazing contours with coincident contours
self intersecting contour

Fig. 5

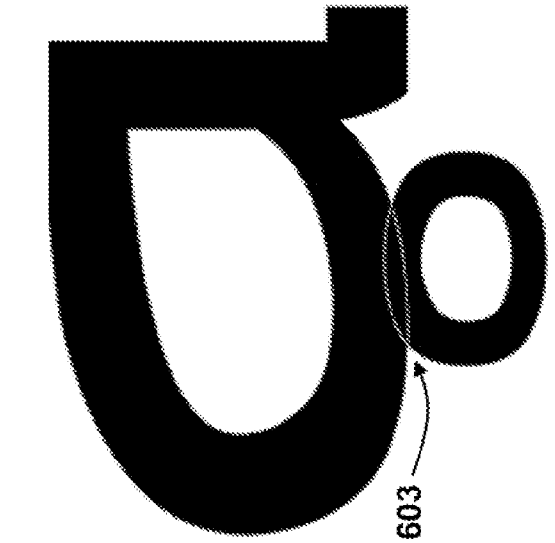


Fig. 6C
prior art

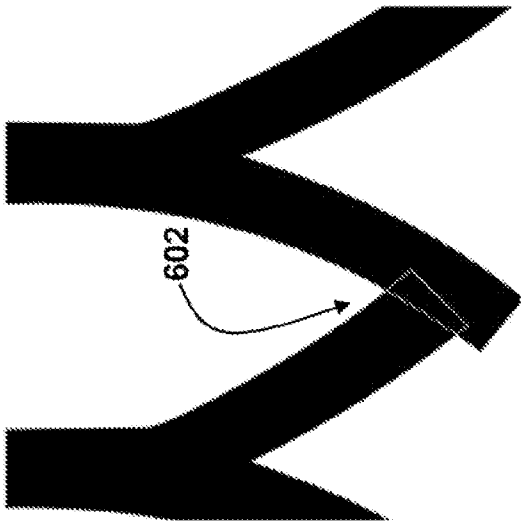


Fig. 6B
prior art

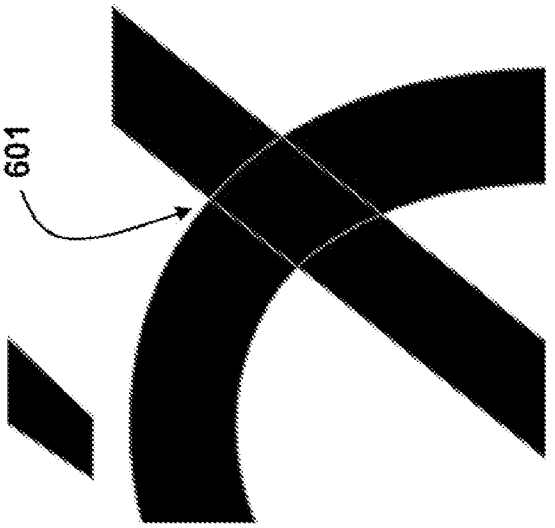


Fig. 6A
prior art

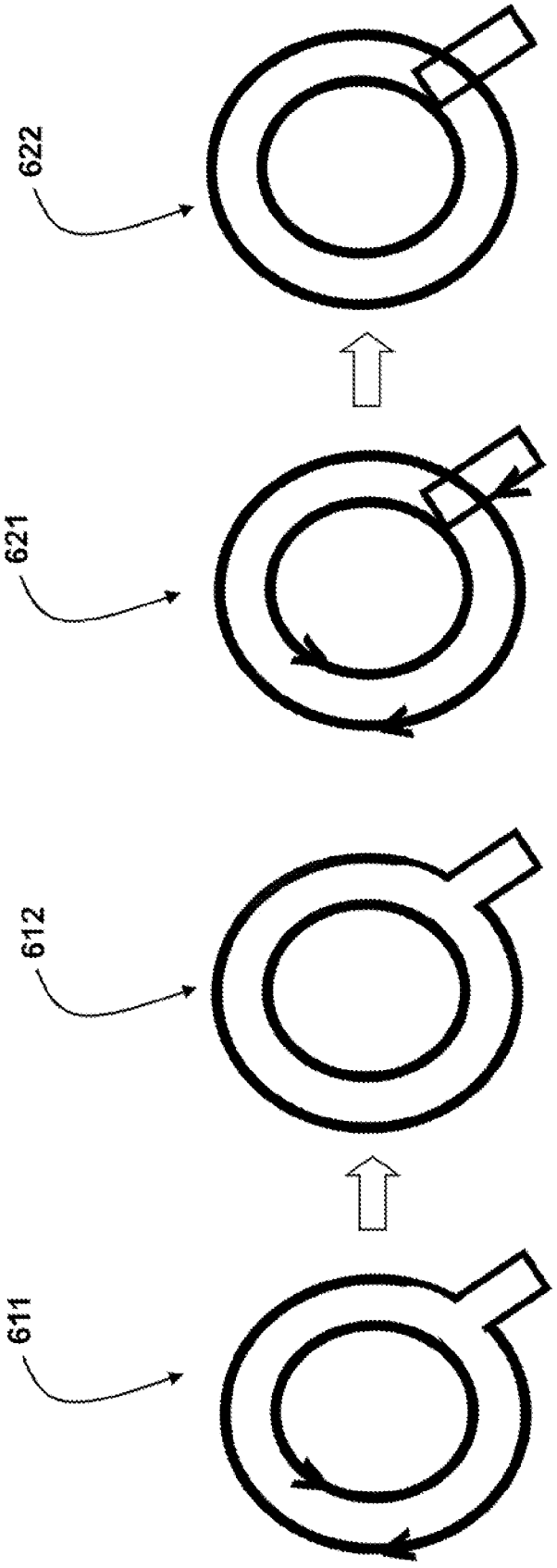


Fig. 6D
prior art

Fig. 6E
prior art

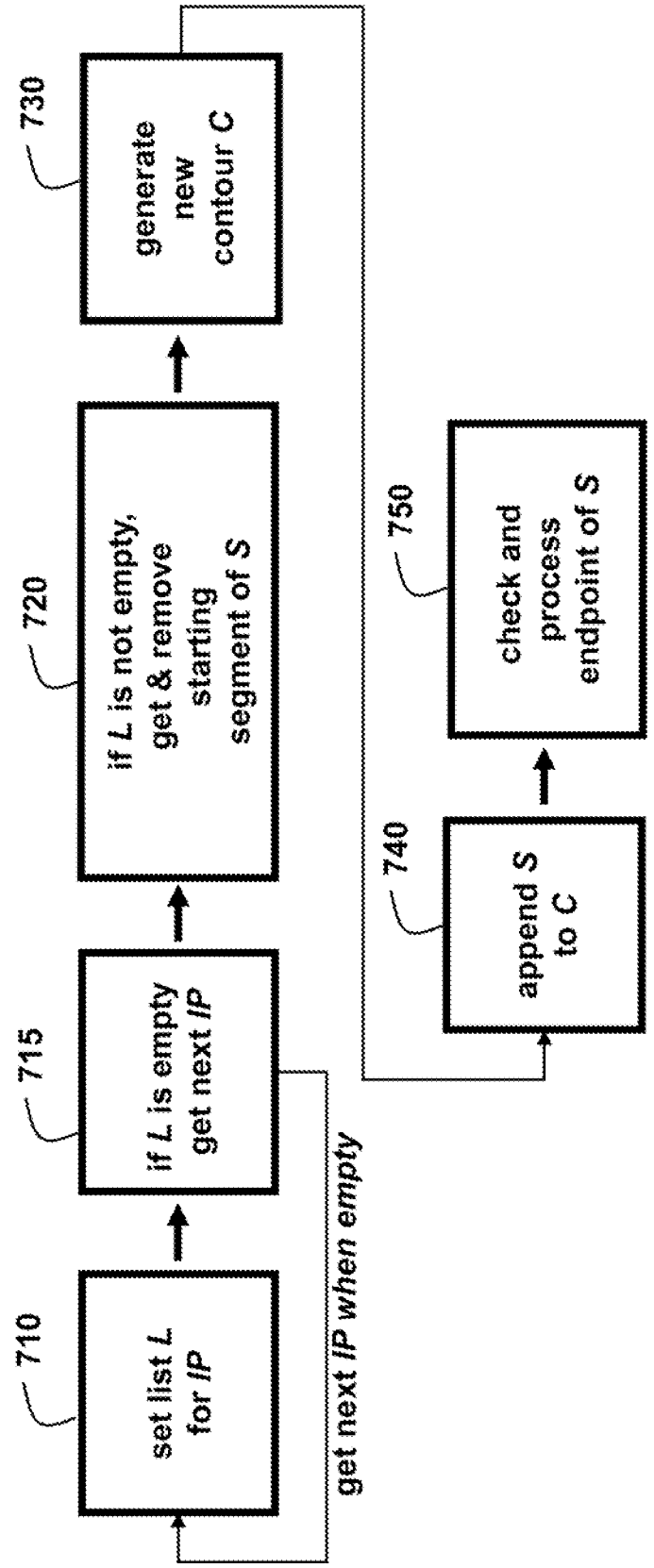

710
set list L for IP
715
if L is empty get next IP
720
if L is not empty, get & remove starting segment of S
730
generate new contour C
740
append S to C
750
check and process endpoint of S
get next IP when empty

Fig. 7

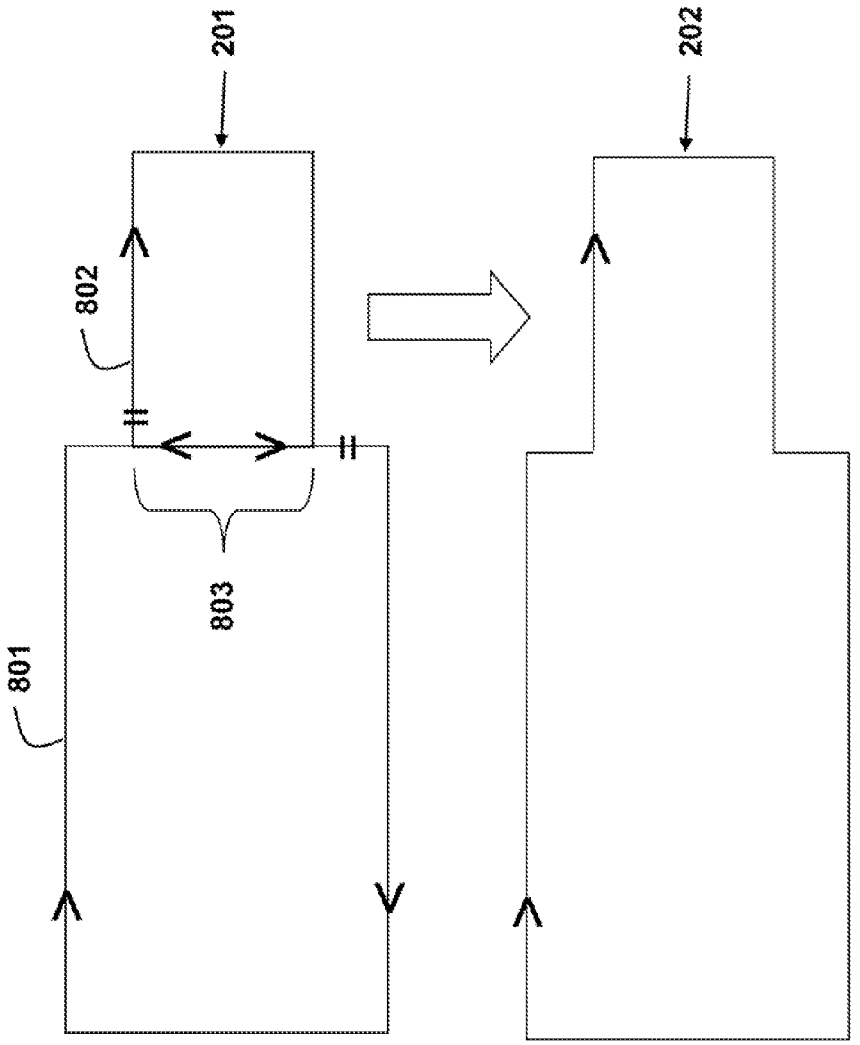


Fig. 8

METHOD FOR RENDERING PATHS WITHOUT OUTLINING ARTIFACTS

FIELD OF THE INVENTION

[0001] This invention relates generally to computer graphics, and more particularly to rendering paths without outlining artifacts.

BACKGROUND OF THE INVENTION

[0002] Paths, Contours, and Glyphs

[0003] In the field of computer graphics, two-dimensional paths are often used to represent shapes of graphical objects that require rendering to a physical device. Examples of such objects include glyphs, structured vector graphics, illustrations, corporate logos, maps, and the like. Although we focus here on digital type, possibly the most common and important two-dimensional object, the following description applies to all types of two-dimensional objects. A collection of glyphs with a consistent design is called a font. Fonts are ubiquitous in computer applications. Fonts can be rendered on many types of physical devices such as computer monitors, telephones, printers, cameras, personal digital assistants (PDAs), global positioning devices, televisions, and the like.

[0004] A glyph is described by a path. Formally, a path includes a set of contours and a fill rule. A contour is a bounded and closed region represented as a sequence of piecewise continuous directed segments. Segments can be linear or curved. Fill rules include a nonzero winding rule and an even-odd parity rule.

[0005] The even odd parity rule determines the "insideness" of a point for a shape defined by a path by drawing a ray from that point to infinity in any direction and counting the number of path segments from the shape that the ray crosses. If this number is odd, then the point is inside; if even, the point is outside.

[0006] The non-zero winding rule is more complex. For a given path C and a given point P: construct a ray, i.e., a straight line, heading out from P in any direction towards infinity. Find all the intersections of C with this ray. Score up the winding number as follows: for every clockwise intersection, i.e., the path passing through the ray from left to right, as viewed from P, subtract 1; for every counter-clockwise intersection, i.e., path passing from right to left, as viewed from P, add 1. If the total winding number is zero, then P is outside C; otherwise, P is inside C.

[0007] Glyphs for computer applications are most frequently designed according to the nonzero winding rule. Glyphs can be filled with, e.g., a solid color, or their paths can be outlined, without filling interior portions, to achieve various visual effects.

[0008] Nonzero Winding Rule

[0009] There are several problems when rendering paths that are to be filled or outlined according to the nonzero winding rule. First, many rendering systems do not support the nonzero winding rule because of its complexity, whereas almost all rendering systems support the even-odd parity rule. Second, the nonzero winding rule is slower to execute than the even-odd parity rule. This can be a problem when rendering on resource constrained devices.

[0010] Third, as shown in FIGS. **6**A, **6**B, and **6**C for partial glyphs, the nonzero winding rule can produce "interior edge haloes" **601-603** for rendering systems that operate in a certain way. The halo artifacts occur because the rendering first fills the shape, and then unconditionally antialiases all the edges, which spoils the filling.

[0011] Other interior artifacts are shown in FIGS. **6**D and **6**E. FIG. **6**D shows a path **611** with two (directed) contours that would be outlined correctly in the prior art as the letter Q **612**. However, if the path **621** is defined by three contours or natural "strokes" as shown in FIG. **6**E, then the interior (non-boundary) edges are not removed during the rendering, and the outline **622** is incorrect.

[0012] Other problems exist when using the nonzero winding rule, even for rendering systems that support the rule. Paths can often contain self-intersections, coincident segments, and other degenerate cases that rendering systems improperly handle because the systems incorrectly fills or outlines the path.

[0013] Therefore, it is desirable to convert a path defined by the nonzero winding rule to an equivalent path that can be rendered with either the even-odd parity rule or the nonzero winding rule. It would be ideal if the equivalent path was simpler, smaller, faster to render, and did not exhibit any incorrect regions or annoying rendering artifacts due to degenerate cases such as those described above.

[0014] Furthermore, it is desirable to enable the correct and accurate determination of a segment of a path as either interior or exterior, even when the path contains self-intersections, coincident segments, and other degenerate cases. The correct and accurate determination for the segment permits path rendering systems, path compression systems, path simplification systems, and the like to function correctly when the path contains degenerate cases.

[0015] U.S. Pat. No. 6,111,587 describes a method that converts a polygon defined by a nonzero winding rule to a polygon defined by an even-odd parity rule. That method operates on closed polygons with linear edges, where each polygon includes a set of labeled contours. The method does not correctly handle degenerate cases such as coincident segments, coincident points, and self-intersections in all of their variations. That method cannot render a simplified polygon defined by the nonzero winding rule as output.

[0016] U.S. Pat. No. 7,321,373 describes a method for performing set operations on two or more arbitrary paths to produce a simple outline path. Like U.S. Pat. No. 6,111,587, that method does not handle all degenerate cases correctly.

SUMMARY OF THE INVENTION

[0017] A method for outlining a two-dimensional input path defined according to a nonzero winding rule is described. Degenerate segments and degenerate contours of the input path are removed. Intersections of the input path are determined. Contours of the input path that include intersections are marked. Unmarked interior contours are removed. Intersections are linked. The marked contours are walked to form new contours. Marked contours and degenerate contours are removed. The new contours and the unmarked contours are collected to form an equivalent output path. The segments of the equivalent output path are outlined.

Differences with the Prior Art

[0018] In general, prior art methods operate on closed polygons with linear edges where each polygon includes a set of known labeled contours. The embodiments operate on a more general representation of shape, i.e., open and closed paths

with nonlinear edges and no predetermined labeling of which contours belong to which shapes.

[0019] Prior art methods convert paths filled by the nonzero winding rule to paths filled by the even-odd parity rule. The embodiments convert to paths that can be filled with either the even-odd parity rule or the nonzero winding rule. The output paths in the embodiments often require less geometry than the input paths and therefore are more efficient to render and to store.

[0020] Prior art methods do not correctly handle coincident edges and coincident points in all cases, e.g., those methods use different steps to determine "outside edges," and do not properly determine "outside edges" in all cases. There, contours "switch" differently due to how the "outside edges" are determined. Also, prior art methods do not always correctly handle degenerate segments, nor do they perform iterative intersection testing in the manner described in the embodiments on contours until convergence to properly handle arithmetic round off errors on underlying integer grid coordinates.

[0021] The prior art methods fail to handle all degenerate cases correctly, and do not convert input nonzero winding rule paths to equivalent output paths that are simpler with less geometry as the paths used by the invention. To facilitate this simplification, the embodiments perform various steps to remove unmarked interior contours, degenerate segments, and degenerate contours.

BRIEF DESCRIPTION OF THE DRAWING

[0022] FIG. **1** is a schematic of example contours and paths used by embodiments of the invention including input and output paths;

[0023] FIG. **2** is a flow diagram of a method for converting a two-dimensional input path defined b a nonzero winding rule to an equivalent 2D output path;

[0024] FIG. **3** is a flow diagram of a procedure for labeling segments according to embodiments of the invention;

[0025] FIG. **4** is a schematic of the method of FIG. **2**;

[0026] FIG. **5** is a schematic of pairs of input and output paths according to embodiments of the invention;

[0027] FIGS. **6**A, **6**B, **6**C, **6**D and **6**E are schematics of prior art renderings of glyphs;

[0028] FIG. **7** is a flow diagram of a walking procedure according to embodiments of the invention; and

[0029] FIG. **8** is a schematic of a procedure for labeling coincident edges according to embodiments of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0030] Some embodiments of the invention provide a method for converting a two-dimensional input path defined according to a nonzero winding rule to an equivalent 2D output path that can be rendered by either the nonzero winding rule or an even-odd parity rule. The embodiments correctly label a segment of a path as interior or exterior. The embodiments can produce renderings without aliasing or outlining artifacts. The embodiments are described using the following definitions.

Definitions

[0031] As shown in FIG. **1**, an input path **201** includes a set of contours **101**-**102**. Contours are represented as a sequence of piecewise continuous directed segments. A degenerate contour is an unbounded or open region **104**, or a region with a zero area or a contour defined by a single point **103**.

[0032] A segment is defined using two or more points with (x, y) coordinates defined on an integer grid. The points include a start point **104** and an end point **105**. Optionally, there can also be one or more control points **106**. For simplicity, this description is limited to linear and Bezier segments defined by two or three points, the extension to any number of points and other curved segments, such as B-splines, will be obvious to those skilled in the art. The segment emanates from the start point and terminates at the end point. The quadratic Bezier segment also includes an off-segment control point. The segment is exterior when it is a portion of a contour that defines a filled region according to the nonzero winding rule. Otherwise, the segment is interior.

[0033] Degenerate segments are transformed during processing to produce non-degenerate segments. Some examples follow. A segment start point cannot be coincident to its endpoint. Such segments are discarded. The start point and end point of a quadratic Bezier curve segment B(t), where t ranges from 0 to 1 inclusively, cannot lie on an "interior" portion of the curve segment, i.e., a portion of the curve segment with a parametric time t such that $0<t<1$. Such segments are replaced with a line segment. The off-curve control point of a quadratic Bezier curve segment cannot be coincident with either the start point or end point of that quadratic Bezier curve segment. Such segments are replaced with a line segment.

[0034] Segments of each contour of the path are partitioned into multiple segments to enforce that every segment is monotonic in both the x and y directions. A segment can be partitioned into two segments by inserting a control point on the segment interior. A quadratic Bezier segment is partitioned by inserting control points according to, e.g., a De Casteljau algorithm. This significantly improves performance when determining self-intersecting contours. It also makes the labeling of a segment as interior or exterior simple and accurate.

[0035] Segments are approximated using linear subdivisions. A line segment is trivially approximated using a single linear subdivision. A quadratic Bezier segment is approximated using one or more linear subdivisions. The control points of the linear subdivisions used to represent the quadratic Bezier segment are rounded to integer locations on the grid. A target rendering size is used to limit the number of linear subdivisions required to represent a quadratic Bezier segment, thereby improving the performance of computing intersections. Linear subdivisions are determined on an as-needed basis and then saved for future processing. The start and end points of a linear subdivision cannot be coincident; such subdivisions are removed.

[0036] A junction contains data associated with a specific intersection. The junction specifies the Cartesian location of the intersection and maintains a list of exterior segments emanating from that intersection, i.e., a list of outbound exterior segments.

[0037] Convert Paths Defined by the Nonzero Winding Rule

[0038] As shown in FIG. **2**, some embodiments of the invention provide a method for converting the input path **201** defined by the nonzero winding rule to the equivalent output path **202** that can be rendered by either the nonzero winding

rule or the even-odd parity rule. The paths are defined in a two-dimensional (2D) coordinate system. The steps are schematically shown in FIG. **4**.

[0039] Quantize and Transform

[0040] First, the method optionally converts, i.e., quantizes and transforms, **205** non-integer control points of the input path to integers, if necessary. For example, TrueType® fonts are defined on an integer coordinate system of sufficient precision called the EM square and therefore do not require quantization or transformation. When conversion is necessary, control points of the input path specified in floating point or fixed point are transformed, e.g., multiplied by 256, and then rounded to determine integers of sufficient precision.

[0041] When this step is skipped, the control points of the input path remain in their original coordinate system. The original coordinate system can be defined on a floating point or a fixed point grid, to name just a few. Arithmetic operations can be performed with a greater precision by skipping the quantizing and transforming, although this may increase processing time and produce flaws in the output path **202**.

[0042] Replace or Remove Degenerate Segments and Optionally Enforce Monotonicity

[0043] Then, after integer conversion, degenerate segments are removed **210**. Some examples of degenerate segments follow. A segment start point cannot be coincident to its endpoint. Such segments are discarded. The start point, and end point of a quadratic Bezier curve segment B(t), where t ranges from 0 to 1 inclusively, cannot lie on an "interior" portion of the curve segment, i.e., a portion of the curve segment with a parametric time t such that $0<t<1$. Such segments are replaced with a line segment. The off-curve control point of a quadratic Bezier curve segment cannot be coincident with either the start point or end point of that quadratic Bezier curve segment. Such segments are replaced with a line segment.

[0044] Optionally, monotonicity of the segments in both the x and y directions are enforced **215**.

[0045] Remove Degenerate Contours

[0046] Degenerate contours are removed **220**. Some examples of degenerate contours include a contour comprised of a single point, an open contour, i.e., a contour which is not "watertight," and a contour with no interior area.

[0047] Construct Data Structures for Accelerating Performance

[0048] Data structures for improving the run-time performance of the method are optionally constructed **225**. The data structures can also be constructed on demand and only when necessary during steps **230** and **240**. Example data structures include bounding boxes, proximity cluster trees, and grids for segments and contours.

[0049] Determine Intersections

[0050] Self-intersecting contours are determined **230**. Self-intersecting contours are partitioned and split at each point P of self-intersection by inserting new control points at P into the sequence of continuous segments preceding and following P. If the optional quantization and transformation step was performed, then intersections are determined at integer coordinates of the grid. Bezier segments are partitioned into linear subdivisions as needed when it is possible the segments could contribute to an intersection. Intersection testing is performed on the linear subdivisions to improve performance and accuracy. The monotonicity of the segments and the acceleration data structures are used to minimize segment-segment intersection tests. Mark **235** each contour of the input path containing a self-intersection. This is followed by determining **240** contour-contour intersections, which are also marked **245**. For the purpose of this description, contours that are not explicitly marked in some way are considered "unmarked."

[0051] Following are additional rules and conventions when determining intersections. To avoid redundant intersections, intersections that occur at $t=0$ are not counted, but intersections that occur at $t=1$ are counted. Two coincident line segments have at most two intersections. A sequence of line segments connected in a contour at a common endpoint does not intersect at that common endpoint. After intersecting contours have been processed, any linear subdivisions generated are no longer used. The intersections are inserted into the contours, e.g., Bezier segments are partitioned and split at their intersections, and the contours of the path are walked as described below.

[0052] The contour intersection steps are repeated until no further intersections occur. This repetition is performed because new intersections can occur as a result of previously determined intersections. This step ensures that the output path **202** is correct and that subsequent rendering and processing is without any artifacts. Note that most input paths require only a single iteration.

[0053] Remove Unmarked Interior Contours

[0054] Unmarked interior contours of the input path **201** are removed **250**. Determine the winding number of any segment, e.g., we typically use the first segment of each unmarked contour of the input path **201** to determine whether the segment is an interior contour and needs to be removed prior to the walking step. Contours of the input path **201** initially hidden may be revealed after the walking step and therefore are removed now.

[0055] Linking and Generating Junctions

[0056] Then, intersections are linked **255** by generating a junction associated with the Cartesian location of each intersection. A junction contains data about a specific intersection and specifies the Cartesian location of the intersection and maintains a list of segments starting at that location. For each unique intersection, i.e., each intersection with unique (x, y) coordinates, a list L of exterior outbound segments is determined. For each inbound segment to the intersection, a pointer to L is maintained. An exterior outbound segment is a segment starting at an intersection, emanating outward from the intersection, and whose winding number indicates it is an exterior edge of the input path **201**. The proper labeling of outbound segments as exterior is an important step. After linking **255**, the segments of L are properly labeled.

[0057] Walking

[0058] After the intersections are linked **255**, each marked contour of the path can be walked **260** to generate new contours by merging. The walking can start at any junction on a marked contour. Segments of marked contours are either copied to the new contours or removed when the segments are interior. The detailed steps of the walking process, as shown in FIG. **7**, are as follows:

[0059] 1. For each intersection point IP, i.e., for each junction, of the marked contours perform these steps:

[0060] 2. Set **710** L to the list of exterior outbound segments of IP.

[0061] 3. If **715** L is empty, then goto step 1 and proceed to the next intersection point.

[0062] 4. If **720** L is not empty, then get and remove the starting segment L. Set S to this segment.

[0063] 5. Generate **730** a new contour C.

[0064] 6. Append **740** segment S to the end of C.

[0065] 7. If **750** the end point of S is IP, then we have traversed back to IP. Goto step 3.

[0066] 8. If **750** the end point of S is not another intersection point, then get the next segment, set this segment to S, and proceed to step 6.

[0067] 9. If **750** the end point of S is another intersection point Q other than IP, then switch contours as follows: get the junction J of Q, get and remove an exterior outbound segment of J, set this exterior outbound segment to S, and proceed to step 6.

[0068] Remove Marked Contours

[0069] Next, marked contours of the input path **201** are removed **265**. Segments of marked contours that contribute to the output path **202** have been copied into the new contours in the previous walking step.

[0070] Degenerate Contours are Removed

[0071] Next, degenerate contours of the remaining contours are removed **268**. Examples of degenerate contours include a contour with a single point, an open contour, and a contour with no interior area.

[0072] Forming the Output Path

[0073] Finally, the new contours and unmarked contours are collected **270** as the output path **202**. The simplified output path can be rendered by either the nonzero winding rule or the even-odd parity rule.

[0074] Invert Quantization and Transformation

[0075] If the input path **201** was optionally quantized and transformed in step **205**, then that process is inverted **275** on the output path **202** to restore the coordinates to their original coordinate system.

[0076] Effects of the Conversion

[0077] FIG. **5** shows example pairs of paths **501-506**, with the input paths **201** on the left, and the output paths **202** on the right. The examples show that the output paths **202** will be correctly rendered with either the nonzero winding rule or the even odd parity rule. Note the presence of degenerate cases, such as coincident segments in the input paths **201**, and their proper handling to produce the correct output paths **202**.

[0078] The merge contours procedure is object-based. The procedure explicitly determines the 2D output path **202** for further processing. Prior art image-based approaches perform this operation during rasterization so that only the pixels exhibit the result, and an explicit 2D output dimensional path is not formed as described herein.

[0079] The input to the merge contours procedure is a single directed 2D path, such as a path representing a glyph. The interior of the path is defined by the nonzero winding rule. The output is an equivalent and simplified 2D path that can be rendered by either the nonzero winding rule or the even-odd parity rule. The simplification can include removal of overlapping, self-intersecting, degenerate, and unnecessary contours.

[0080] Path Conversion: An Example

[0081] FIG. **4** shows the operation of the method for converting an input path **201** schematically. The input path **201** includes an exterior contour **401**, an interior contour **402**, and two intersecting contours **403-404**. The contour **403** is also self-intersecting **411**. The contours **403-404** have two contour-contour intersections **412**. The contours **403-404** with intersections are marked (M). The unmarked interior contour **402** is removed. Next, junctions are associated with the intersections. Outbound segments **420** at each junction are interior or exterior. The exterior outbound segments at each junction are labeled and indicated by "||".

[0082] During the walking, a new contour **430** is generated by merging contours while marked and degenerate contours are removed.

[0083] The steps of the above method, as well as any other procedures or methods described herein, can be performed in a processor connected to a memory and input/output interfaces and devices as known in the art.

[0084] Rendering: Filling or Outlining Paths without Artifacts

[0085] The output path **202** as generated above can be rendered **280** by filling or outlining with either the nonzero winding rule or the even-odd parity rule. Antialiasing **285** can also be applied. In contrast with the prior art, the filled path does not include any interior aliasing artifacts. Similarly, the outlined path does not include any interior edge artifacts.

[0086] Interior and Exterior Segment Labeling

[0087] FIG. **3** shows a procedure to determine whether a segment S is interior or exterior. First, a winding number W is initialized **305** to zero. Next, a scan line that is guaranteed to intersect the segment is identified **310**. If monotonicity is enforced for segments in both x and y directions, then the scan line can be quickly and accurately identified to be either a horizontal or vertical scan line passing through the midpoint of the segment.

[0088] For each contour C, the winding values are accumulated **315** into W for any segments of the contour C that cross the identified scan line before crossing the segment S. The accumulation of winding values into W for coincident segments of the contour C that cross the scan line at the same location as segment S is postponed until after all contours have been processed. The coincident segments have to be treated as a group, as far as the winding number is concerned, because they overlap and cancel each other. The coincident segments are marked and saved **320** in a list L; also see FIG. **8** for coincident segments.

[0089] When there are no coincident segments, we proceed as follows. If **345** W is nonzero and W, when updated to account for S, is nonzero, then the segment S is marked **350** interior, otherwise the segment S is marked exterior.

[0090] When there are coincident segments, we proceed as follows. Add **335** segment S to the list L of saved coincident segments. Pairs of coincident segments in the list L that have opposite directions cancel each other out, and the accumulation of their winding numbers is zero and does not change W. The winding values are accumulated **340** into W for the remaining non-cancelled segments in the list L, excluding segment S. If **345** W is nonzero and W, when updated to account for S, is nonzero, then the segment S is marked **350** interior, otherwise the segment S is marked exterior.

[0091] It is noted that prior art methods do not process coincident segments as described above, and therefore can produce incorrect output paths with interior artifacts.

[0092] FIG. **8** schematically shows the labeling of outbound segments of contours **801-802** of the input path **201** so that the output path **202** is correct. The contours have coincident segments **803**. The exterior outbound segments are indicated by "||". Because of the nature of the steps outlined above, the segments **803** are correctly identified as interior, and hence these segments are removed during the walk. To the best of our knowledge no prior art path rendering deals with this case correctly in all of its variations. In this case, the

coincident segments need to be treated as a group when the winding number is determined.

[0093] Applications and Distinguishing Features

[0094] Rendering methods that fill paths according to the even-odd parity rule can use the merge contours procedure as outlined in FIG. **2** as a preprocessing step to correctly render paths that are designed to be filled according to the nonzero winding rule, e.g., glyphs in TrueType® fonts.

[0095] The merge contours procedure can also be used to eliminate interior edge haloes and outlining artifacts present in various font rendering systems. The even-odd parity rule is less complex and faster to perform than the nonzero winding rule. Consequently, the merge contours procedure can be used to convert the input path to an equivalent output path and then rendered using the even-odd parity rule.

[0096] The merge contours procedure can also be used to perform two-dimensional constructive solid geometry (CSG) operations, such as union, intersection, and difference, on two-dimensional paths.

[0097] The merge contours procedure solves a very difficult computational geometry problem in real-time and has several distinguishing features when compared to the prior art. The merge contours procedure can handle difficult grazing conditions and singularities, e.g., coincident points and segments. The procedure supports nonlinear curved edges, as well as directed edges. Most prior art procedures only work on polygons with linear edges. The merge contours procedure can use integer arithmetic and operate on an integer grid. It is one to two orders of magnitude faster than comparable prior art procedures.

[0098] The merge contours procedure also provides concurrent classification of coincident edges to properly handle grazing conditions, degenerate preprocessing and post processing of segments and contours, and on-demand tessellation of curved segments for determining intersections. The tessellation of curved segments can be based on a target rendering size, thereby optimizing performance for the given target. Monotonicity of segments can be enforced for better performance and accuracy. During the merging, all control points can be represented on an integer coordinate system to improve accuracy and performance. When contour segments are partitioned and split at intersections, the procedure can use iterative intersection testing until convergence to ensure robustness.

[0099] The correct labeling of segments as either interior or exterior permits path rendering systems, path compression systems, path simplification systems, and the like to function correctly when the path contains degenerate cases.

[0100] Although the invention has been described by way of examples of preferred embodiments, it is to be understood that various other adaptations and modifications can be made within the spirit and scope of the invention. Therefore, it is the object of the appended claims to cover all such variations and modifications as come within the true spirit and scope of the invention.

We claim:

1. A method for outlining an input path, wherein the input path is defined according to a nonzero winding rule in a two-dimensional (2D) coordinate system, the input path includes a set of contours, and each contour includes a sequence of segments, comprising the steps of:

removing degenerate segments of the input path;

removing degenerate contours of the input path;

determining intersections of the input path;

marking the contours of the input path that include intersections;

removing unmarked interior contours;

linking the intersections;

walking the marked contours to form new contours;

removing marked contours;

removing degenerate contours; and

collecting the new contours and the unmarked contours into an output path equivalent to the input path; and

outlining the segments of the output path to outline the input path, wherein the steps are performed in a processor.

2. The method of claim **1**, wherein the input path represents a glyph.

3. The method of claim **1**, wherein the input path represents an illustration.

4. The method of claim **1**, wherein the input path represents a structured vector graphic.

5. The method of claim **1**, further comprising:

approximating the curved segments with linear subdivisions to determine intersections.

6. The method of claim **1**, further comprising:

enforcing monotonicity on the segments.

7. The method of claim **1**, further comprising:

associating a junction with each intersection, wherein the junction specifies a Cartesian location of the intersection and maintains a list of exterior segments emanating from the intersection.

8. The method of claim **1**, wherein the degenerate segments include any segment with a start point coincident with an end point of the segment, any segment that is a curve with the start point and the end point on an interior portion of the curve, and any segment with an off-curve control point of the curve that is coincident with either the start point or the end point of that curve, and wherein the degenerate contours include any contour that is an unbounded or an open region, or a region with a zero area, or any contour defined by a single point.

9. The method of claim **1**, wherein some of the segments are coincident.

10. The method of claim **1**, wherein some of the contours are self-intersecting.

11. The method of claim **1**, wherein coordinates defining the segments are specified with integers.

12. The method of claim **1**, wherein coordinates defining the segments are specified with floating point numbers.

13. The method of claim **1**, wherein coordinates defining the segments are specified with fixed point numbers.

14. The method of claim **1**, wherein the determining of the intersections of the input path is performed on an integer grid.

15. The method of claim **1**, wherein the determining of the intersections of the input path is performed on a floating point grid.

16. The method of claim **1**, wherein the determining of the intersections of the input path is performed on a fixed point grid.

17. The method of claim **1**, wherein the determining of the intersections of the input path is repeated until no further intersections are found.

18. The method of claim **1**, wherein the determining of the intersections of the input path uses on demand tessellation of curved segments at a target rendering size to improve performance.

19. The method of claim **1**, wherein the determining of the intersections of the input path uses acceleration data structures to improve performance.

20. The method of claim **19**, wherein the acceleration data structures includes bounding boxes, trees, or grids.

21. The method of claim **1**, wherein the segments of the input path are quantized and transformed to an integer grid prior to processing.

22. The method of claim **21**, wherein the segments of the output path are transformed back to an original coordinate system before the collecting.

* * * * *

(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6169048号
(P6169048)
(45)発行日　平成29年7月26日(2017.7.26)　(24)登録日　平成29年7月7日(2017.7.7)

(51)Int.Cl.　F I
G06T 11/40 (2006.01)　G06T 11/40

請求項の数 26　外国語出願　(全 17 頁)

(21)出願番号	特願2014-121399 (P2014-121399)
(22)出願日	平成26年6月12日 (2014.6.12)
(65)公開番号	特開2015-22760 (P2015-22760A)
(43)公開日	平成27年2月2日 (2015.2.2)
審査請求日	平成29年3月8日 (2017.3.8)
(31)優先権主張番号	13/942,799
(32)優先日	平成25年7月16日 (2013.7.16)
(33)優先権主張国	米国 (US)
(31)優先権主張番号	13/942,808
(32)優先日	平成25年7月16日 (2013.7.16)
(33)優先権主張国	米国 (US)
(31)優先権主張番号	13/942,828
(32)優先日	平成25年7月16日 (2013.7.16)
(33)優先権主張国	米国 (US)

(73)特許権者　000006013
三菱電機株式会社
東京都千代田区丸の内二丁目７番３号
(74)代理人　100110423
弁理士　曾我　道治
(74)代理人　100111648
弁理士　梶並　順
(74)代理人　100122437
弁理士　大宅　一宏
(74)代理人　100147566
弁理士　上田　俊一
(74)代理人　100161171
弁理士　吉田　潤一郎
(74)代理人　100161115
弁理士　飯野　智史

最終頁に続く

(54)【発明の名称】入力パスを変換する方法、入力パスのセグメントを内部又は外部としてラベル付けする方法、入力パスをレンダリングする方法、及び、入力パスの外形を描く方法

(57)【特許請求の範囲】
【請求項１】
オブジェクトベースの入力パスを変換する方法であって、該オブジェクトベースの入力パスは２次元（２Ｄ）座標系における非ゼロ巻数規則に従って定義され、前記オブジェクトベースの入力パスは１組の輪郭を含み、各輪郭は一連のセグメントを含み、該方法は、
前記オブジェクトベースの入力パスの縮退セグメントを除去するステップと、
前記オブジェクトベースの入力パスの縮退輪郭を除去するステップと、
前記オブジェクトベースの入力パスの交差を求めるステップと、
前記交差を含む前記オブジェクトベースの入力パスの前記輪郭をマーク付けするステップと、
マーク付けされていない内部輪郭を除去するステップと、
前記交差をリンク付けするステップと、
前記マーク付けされた輪郭をウォーキングすることにより新たな輪郭を形成するステップと、
前記マーク付けされた輪郭を除去するステップと、
縮退輪郭を除去するステップと、
前記新たな輪郭及びマーク付けされていない輪郭を収集することにより前記オブジェクトベースの入力パスに等価な出力パスを得るステップであって、前記オブジェクトベースの入力パスが、レンダリングされるときに、単一のピクセル内に前記セグメントをラベル付けするように、前記出力パスは、ピクセル・グリッドにレンダリングされるときに、サ

ブ・ピクセルの形状を示す、ステップと
を含み、各前記ステップはプロセッサが実行する、入力パスを変換する方法。
【請求項２】
前記オブジェクトベースの入力パスはグリフを表す、請求項１に記載の方法。
【請求項３】
前記オブジェクトベースの入力パスはイラストを表す、請求項１に記載の方法。
【請求項４】
前記オブジェクトベースの入力パスは構造化ベクトルグラフィックを表す、請求項１に記載の方法。
【請求項５】
いかなるエイリアス又は内部エッジアーティファクトを生成することなく、前記非ゼロ巻数規則および奇数－偶数パリティ規則のいずれか一方によって前記出力パスをレンダリングすることを更に含む、請求項１に記載の方法。
【請求項６】
前記レンダリングは前記輪郭を塗りつぶす、請求項５に記載の方法。
【請求項７】
前記レンダリングは前記輪郭の外形を描く、請求項５に記載の方法。
【請求項８】
線形区画を用いて曲線セグメントを近似することにより交差を求めることを更に含む、請求項１に記載の方法。
【請求項９】
前記セグメントに単調性を強制することを更に含む、請求項１に記載の方法。
【請求項１０】
接続点を各前記交差と関連付けることを更に含み、前記接続点は前記交差のデカルトロケーションを特定し、前記交差から発する外部セグメントのリストを保持する、請求項１に記載の方法。
【請求項１１】
前記縮退セグメントは、
そのセグメントの始点が終点と一致する任意のセグメントと、
曲線から構成され、前記曲線の内部部分に前記始点及び前記終点を有する任意のセグメントと、
前記曲線の前記始点又は前記終点のいずれかと一致する前記曲線のオフカーブ制御点を有する任意のセグメントと
を含み、
前記縮退輪郭は、
非有界領域又は開領域である任意の輪郭、ゼロエリアを有する領域、又は、単一点によって定義される任意の輪郭を含む、
請求項１に記載の方法。
【請求項１２】
前記セグメントのうちのいくつかは一致する、請求項１に記載の方法。
【請求項１３】
前記輪郭のうちのいくつかは自己交差である、請求項１に記載の方法。
【請求項１４】
前記セグメントを定義する座標は整数を用いて指定される、請求項１に記載の方法。
【請求項１５】
前記セグメントを定義する座標は浮動小数点数を用いて指定される、請求項１に記載の方法。
【請求項１６】
前記セグメントを定義する座標は固定小数点数を用いて指定される、請求項１に記載の方法。

【請求項１７】
前記オブジェクトベースの入力パスの前記交差を求めることは、整数グリッドにおいて行われる、請求項１に記載の方法。
【請求項１８】
前記オブジェクトベースの入力パスの前記交差を求めることは、浮動小数点グリッドにおいて行われる、請求項１に記載の方法。
【請求項１９】
前記オブジェクトベースの入力パスの前記交差を求めることは、固定小数点グリッドにおいて行われる、請求項１に記載の方法。
【請求項２０】
前記オブジェクトベースの入力パスの前記交差を求めることは、更なる交差が見つからなくなるまで繰り返される、請求項１に記載の方法。
【請求項２１】
前記オブジェクトベースの入力パスの前記交差を求めることは、目標レンダリングサイズにおいて曲線セグメントのオンデマンドテッセレーションを用いて性能を改善する、請求項１に記載の方法。
【請求項２２】
前記オブジェクトベースの入力パスの前記交差を求めることは、加速データ構造を用いて性能を改善する、請求項１に記載の方法。
【請求項２３】
前記加速データ構造は境界ボックス、ツリー、又はグリッドを含む、請求項２２に記載の方法。
【請求項２４】
前記オブジェクトベースの入力パスの前記セグメントは量子化され、整数グリッドに変形される、請求項１に記載の方法。
【請求項２５】
前記出力パスの前記セグメントは、前記収集することの前に変形され元の座標系に戻される、請求項２４に記載の方法。
【請求項２６】
オブジェクトベースの入力パスのセグメントを内部又は外部としてラベル付けする方法であって、前記オブジェクトベースの入力パスは２次元（２Ｄ）座標系において非ゼロ巻数規則に従って定義され、前記オブジェクトベースの入力パスは１組の輪郭を含み、各輪郭は一連のセグメントを含み、該方法は、
巻数をゼロに初期化するステップと、
前記セグメントに交差する走査線を特定するステップと、
輪郭ごとに、巻値を前記走査線に交わる前記輪郭の任意のセグメントの前記巻数に累積するステップであって、前記セグメントと同じロケーションにおいて前記走査線に交わる前記輪郭の一致セグメントの前記累積は、全ての前記輪郭が処理されるまで延期され、前記一致セグメントはマーク付けされ、前記マーク付けされた一致セグメントはリスト内に保存される、ステップと、
前記リストが空であり、かつ前記巻数がゼロでなく、かつ前記巻数が前記セグメントを計上するように更新されたときにゼロでない場合、前記セグメントを内部としてラベル付けするステップと、
前記リストが空であり、かつ前記巻数がゼロであるか、又は前記巻数が前記セグメントを計上するように更新されたときにゼロである場合、前記セグメントを外部としてラベル付けするステップと、
前記リストが空でなく、かつ、互いに反対の向きを有する前記リストのセグメント対を相殺した後に前記リストが前記セグメントを含むように拡張され、前記リスト内の前記セグメントを除く任意の残りの相殺されていないセグメントについて前記巻値を前記巻数に累積した後に、前記巻数がゼロでなく、かつ、前記セグメントを計上するように更新され

たときに前記巻数がゼロでない場合、前記セグメントを内部としてラベル付けするステップと、

前記リストが空でなく、かつ、互いに反対の向きを有する前記リストのセグメント対を相殺した後に前記リストが前記セグメントを含むように拡張され、前記リスト内の前記セグメントを除く前記残りの相殺されていないセグメントについて前記巻値を前記巻数に累積した後に、前記巻数がゼロであるか、又は、前記セグメントを計上するように更新されたときに前記巻数がゼロである場合、前記セグメントを外部としてラベル付けするステップであって、前記オブジェクトベースの入力パスが、レンダリングされるときに、単一のピクセル内に前記セグメントをラベル付けするように、ラベル付けされた前記セグメントは、ピクセル・グリッドにレンダリングされるときに、サブ・ピクセルの形状を示す、ステップと

を含み、各前記ステップはプロセッサが実行する、入力パスのセグメントを内部又は外部としてラベル付けする方法。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、包括的にはコンピューターグラフィックスに関し、より詳細には、非ゼロ巻数規則によって定義されたパスを、非ゼロ巻数規則又は奇数－偶数パリティ規則のいずれによってもレンダリングすることができる等価なパスに変換することに関する。

【０００２】

本発明は、包括的にはコンピューターグラフィックスに関し、より詳細には、パスのセグメントを内部又は外部としてラベル付けすることに関する。

【背景技術】

【０００３】

パス、輪郭及びグリフ

コンピューターグラフィックスの分野において、物理デバイスにレンダリングする必要があるグラフィックオブジェクトの形状を表すのに、多くの場合に２次元パスが用いられる。そのようなオブジェクトの例には、グリフ、構造化ベクトルグラフィックス、イラスト、企業のロゴ、マップ等が含まれる。ここではデジタルタイプ、場合によっては最も一般的で重要な２次元オブジェクトに焦点を当てるが、以下の説明は全てのタイプの２次元オブジェクトに当てはまる。一貫した設計を有するグリフの集合体はフォントと呼ばれる。フォントは、コンピューターアプリケーションにおいて遍在している。フォントは、コンピューターモニター、電話、プリンター、カメラ、携帯情報端末（ＰＤＡ）、全地球測位デバイス、テレビ等の多くのタイプの物理デバイス上にレンダリングすることができる。

【０００４】

グリフはパスによって示される。形式上、パスは１組の輪郭及び塗りつぶし規則を含む。輪郭は、一連の区分的に連続した有向セグメントとして表される有界の閉領域である。セグメントは線形又は曲線形とすることができる。塗りつぶし規則は非ゼロ巻数規則（nonzero winding rule）及び奇数－偶数パリティ規則（even-odd parity rule）を含む。

【０００５】

奇数－偶数パリティ規則は、パスによって画定される形状に対する点の「内側性（insideness）」を判断し、当該判断は、任意の方向においてその点から無限遠への光線を描画し、この光線が交差する形状からのパスセグメントの数をカウントすることによって行われる。この数が奇数である場合は、その点は内側にあり、偶数である場合は、その点は外側にある。

【０００６】

非ゼロ巻数規則はより複雑である。所与のパスＣ及び所与の点Ｐについて、任意の方向においてＰから無限遠に向かう光線、すなわち直線を構築する。Ｃとこの光線との全ての交差を見つける。以下のように巻数のスコアを計上する。全ての時計回りの交差について

、すなわちＰから見たときに左から右へ光線を通過するパスについて、１を減算し、全ての反時計回りの交差について、すなわち、Ｐから見たときに右から左へ通過するパスについて、１を加算する。総巻数がゼロである場合、ＰはＣの外側にあり、そうでない場合、ＰはＣの内側にある。

【０００７】

コンピューターアプリケーションのグリフは、非ゼロ巻数規則に従って設計されることが最も多い。グリフは、様々な視覚効果を達成するように、例えば無地一色で塗りつぶすことができる、あるいは、内部部分を塗りつぶすことなくそれらのパスの外形を描くことができる。

【０００８】

非ゼロ巻数規則

非ゼロ巻数規則に従って塗りつぶされるか又は外形を描かれることになるパスをレンダリングするとき、いくつかの問題が存在する。第１に、ほとんど全てのレンダリングシステムが奇数－偶数パリティ規則をサポートしているのに対し、多くのレンダリングシステムは、非ゼロ巻線規則が複雑であることに起因して非ゼロ巻線規則をサポートしていない。第２に、非ゼロ巻数規則は奇数－偶数パリティ規則よりも実行が低速である。これは、リソースが制限されたデバイスにおけるレンダリング時に問題となり得る。

【０００９】

第３に、部分グリフについて図６（ａ）、図６（ｂ）及び図６（ｃ）に示すように、非ゼロ巻数規則は、或る特定の方法で動作するレンダリングシステムの「内部エッジハロー（interior edge haloes）」６０１～６０３を生成する可能性がある。このハローアーティファクトは、レンダリングが最初に形状を塗りつぶし、次に無条件に全てのエッジをアンチエイリアシングし、これによって塗りつぶしが損なわれることに起因して生じる。

【００１０】

他の内部アーティファクトが図６（ｄ）及び図６（ｅ）に示されている。図６（ｄ）は、従来技術において文字Ｑ６１２として正しく外形を描かれる２つの（有向）輪郭を有するパス６１１を示している。しかしながら、パス６２１が、図６（ｅ）に示すように３つの輪郭又は自然な「ストローク」によって画定される場合、内部（非境界）エッジはレンダリング中に除去されず、外形６２２は不正確である。

【００１１】

非ゼロ巻数規則を用いるとき、この規則をサポートするレンダリングシステムであっても、他の問題が存在する。パスは多くの場合に自己交差、一致セグメント、及びレンダリングシステムがパスを不正確に塗りつぶすか又はパスの外形を不正確に描くことに起因して不適切に処理する他の縮退事例を含む。

【００１２】

したがって、非ゼロ巻数規則によって定義されるパスを、奇数－偶数パリティ規則又は非ゼロ巻数規則のいずれを用いてもレンダリングすることができる等価なパスに変換することが望ましい。等価なパスがより単純であり、より小さく、レンダリングをより高速に行い、上記したような縮退事例に起因する不正確な領域又は厄介なレンダリングアーティファクトを一切示さないことが理想である。

【００１３】

さらに、パスが自己交差、一致セグメント、及び他の縮退事例を含む場合であっても、パスのセグメントを内部又は外部として正しく正確に判断可能とすることが望ましい。セグメントを正しく正確に判断することによって、パスが縮退事例を含むときに、パスレンダリングシステム、パス圧縮システム、パス単純化システム等が正しく機能することが可能になる。

【００１４】

特許文献１は、非ゼロ巻数規則によって画定される多角形を奇数－偶数パリティ規則によって画定される多角形に変換する方法を記載している。その方法は、線形のエッジを有する閉じた多角形に対し動作し、各多角形は１組のラベル付けされた輪郭を含む。その方

法は、一致セグメント、一致点、自己交差等の縮退事例を、それらの全ての変形において正しく処理するわけではない。その方法は、非ゼロ巻数規則によって画定される単純化された多角形を出力としてレンダリングすることができない。

【００１５】

特許文献２は、２つ以上の任意のパスに対して集合演算を行い、単純な外形パスを生成する方法を記載している。特許文献１のように、その方法は全ての縮退事例を正しく処理するわけではない。

【先行技術文献】

【特許文献】

【００１６】

【特許文献１】米国特許第６，１１１，５８７号

【特許文献２】米国特許第７，３２１，３７３号

【発明の概要】

【００１７】

方法は、非ゼロ巻数規則に従って定義された２次元入力パスを等価な出力パスに変換する。入力パスの縮退セグメント及び縮退輪郭が除去される。入力パスの交差が求められる。交差を含む入力パスの輪郭がマーク付けされる。マーク付けされていない内部輪郭が除去される。交差がリンク付けされる。マーク付けされた輪郭がウォーキングされ、新たな輪郭が形成される。マーク付けされた輪郭及び縮退輪郭が除去される。新たな輪郭及びマーク付けされていない輪郭が収集され、等価な出力パスが形成される。等価な出力パスは、非ゼロ巻数規則又は奇数－偶数パリティ規則のいずれを用いてもレンダリングすることができる。

【００１８】

非ゼロ巻数規則に従って定義された２次元入力パスのセグメントを内部又は外部としてラベル付けする方法が説明される。巻数が初期化される。セグメントに交差する走査線が特定される。入力パスの輪郭ごとに、巻値が、特定された走査線に交わる輪郭の任意のセグメントの巻数に累積される。ここで、セグメントと同じロケーションにおいて特定された走査線に交わる一致セグメントの累積は延期される。これらの一致セグメントはマーク付けされ、リスト内に保存される。内部又は外部としてのセグメントのラベル付けは、リスト及び巻数から判断される。

【００１９】

非ゼロ巻数規則に従って定義された２次元入力パスをレンダリングする方法が説明される。入力パスの縮退セグメント及び縮退輪郭が除去される。入力パスの交差が求められる。交差を含む入力パスの輪郭がマーク付けされる。マーク付けされていない内部輪郭が除去される。交差がリンク付けされる。マーク付けされた輪郭がウォーキングされ、新たな輪郭が形成される。マーク付けされた輪郭及び縮退輪郭が除去される。新たな輪郭及びマーク付けされていない輪郭が収集され、等価な出力パスが形成される。等価な出力パスの輪郭は、非ゼロ巻数規則又は奇数－偶数パリティ規則のいずれかによって塗りつぶされる。等価な出力パスのセグメントはアンチエイリアシングされる。

【００２０】

非ゼロ巻数規則に従って定義された２次元入力パスの外形を描く方法が説明される。入力パスの縮退セグメント及び縮退輪郭が除去される。入力パスの交差が求められる。交差を含む入力パスの輪郭がマーク付けされる。マーク付けされていない内部輪郭が除去される。交差がリンク付けされる。マーク付けされた輪郭がウォーキングされ、新たな輪郭が形成される。マーク付けされた輪郭及び縮退輪郭が除去される。新たな輪郭及びマーク付けされていない輪郭が収集され、等価な出力パスが形成される。等価な出力パスのセグメントが外形を描かれる。

【００２１】

従来技術との違い

概して、従来技術による方法は、線形エッジを有する閉じた多角形に対し動作し、各多

角形は１組の既知のラベル付けされた輪郭を含む。本発明による実施形態は、形状のより一般的な表現、すなわち、非線形エッジを有し、いずれの輪郭がいずれの形状に属するかの所定のラベル付けのない開いたパス及び閉じたパスに対し動作する。

【００２２】

従来技術による方法は、非ゼロ巻数規則によって塗りつぶされたパスを、奇数－偶数パリティ規則によって塗りつぶされたパスに変換する。本発明による実施形態は、奇数－偶数パリティ規則又は非ゼロ巻数規則のいずれによっても塗りつぶされることができるパスに変換する。本発明による実施形態における出力パスは、多くの場合に入力パスよりも必要とする形状が少なく、したがってレンダリング及び保存がより効果的である。

【００２３】

従来技術による方法は、全ての事例における一致エッジ及び一致点を正しく処理することができるわけではない。例えば、これらの方法は様々なステップを用いて「外側エッジ」を求めるが、全ての事例において「外側エッジ」を適切に求めるわけではない。この方法では、「外側エッジ」がどのように求められるかに起因して輪郭が様々な形で「切り替わる」。また、従来技術による方法は、必ずしも縮退セグメントを正しく処理せず、また、本発明による実施形態において説明されるような方式で輪郭に対する反復的交差テストを収束するまで実行せず、基礎をなす整数グリッド座標における算術丸め誤差に適切に対処しない。

【００２４】

従来技術による方法は、全ての縮退事例を正しく処理することができるわけではなく、本発明によって用いられるパスのように、非ゼロ巻数規則による入力パスを、より少ない形状を有するより単純な等価な出力パスに変換しない。この単純化を容易にするために、本発明による実施形態は様々なステップを実行して、マーク付けされていない内部輪郭、縮退セグメント、及び縮退輪郭を除去する。

【図面の簡単な説明】

【００２５】

【図１】本発明の実施形態によって用いられる、入力パス及び出力パスを含む例示的な輪郭及びパスの概略図である。

【図２】非ゼロ巻数規則によって定義される２次元入力パスを等価な２Ｄ出力パスに変換する方法の流れ図である。

【図３】本発明の実施形態によるセグメントをラベル付けする手順の流れ図である。

【図４】図２の方法の概略図である。

【図５】本発明の実施形態による入力パス及び出力パスの対の概略図である。

【図６Ａ】図６（ａ）～（ｃ）は従来技術によるグリフのレンダリングの概略図である。

【図６Ｂ】図６（ｄ）～（ｅ）は従来技術によるグリフのレンダリングの概略図である。

【図７】本発明の実施形態によるウォーキング手順の流れ図である。

【図８】本発明の実施形態による、一致エッジをラベル付けする手順の概略図である。

【発明を実施するための形態】

【００２６】

本発明のいくつかの実施形態は、非ゼロ巻数規則に従って定義された２次元入力パスを、非ゼロ巻数規則又は奇数－偶数パリティ規則のいずれによってもレンダリングすることができる等価な２Ｄ出力パスに変換する方法を提供する。本発明による実施形態は、パスのセグメントを内部又は外部として正しくラベル付けする。本発明による実施形態は、エイリアシングも外形描画アーティファクトも伴わずにレンダリングを生成することができる。本発明による実施形態は以下の定義を用いて説明される。

【００２７】

定義

図１に示すように、入力パス２０１は１組の輪郭１０１及び１０２を含む。輪郭は一連の区分的に連続した有向セグメントとして表される。縮退輪郭は、非有界領域又は開領域１０４、又はゼロエリアを有する領域、又は単一点１０３によって定義される輪郭である

。
【００２８】
整数グリッド上に定義される（ｘ，ｙ）座標を有する２つ以上の点を用いてセグメントが定義される。点は、始点１０４及び終点１０５を含む。オプションで、１つ又は複数の制御点１０６が存在することもできる。単純にするために、この説明は、２つ又は３つの点によって定義される線形セグメント及びベジェセグメントに限定されるが、任意の数の点及びＢスプライン等の他の曲線セグメントへの拡張は当業者に明らかであろう。セグメントは始点から発し、終点で終了する。二次ベジェセグメントは、オフセグメント制御点も含む。セグメントは、非ゼロ巻数規則に従って塗りつぶされた領域を画定する輪郭の一部であるとき、外部である。そうでない場合、セグメントは内部である。
【００２９】
処理中に縮退セグメントが変換され、非縮退セグメントが生成される。以下にいくつかの例を挙げる。セグメント始点はセグメント終点と一致してはならない。そのようなセグメントは破棄される。二次ベジェ曲線セグメントＢ（ｔ）（ただし、ｔは０～１（０及び１を含む）の範囲をとる）の始点及び終点は、曲線セグメントの「内部」部分、すなわち０＜ｔ＜１であるような時間パラメーターｔを有する曲線セグメント部分に位置することができない。そのようなセグメントは、直線セグメントと置き換えられる。二次ベジェ曲線セグメントのオフカーブ制御点はその二次ベジェ曲線セグメントの始点とも終点とも一致してはならない。そのようなセグメントは直線セグメントと置き換えられる。
【００３０】
パスの各輪郭のセグメントを複数のセグメントに分割し、全てのセグメントがｘ方向及びｙ方向の双方において単調であることを強制する。セグメント内部において制御点を挿入することによってセグメントを２つのセグメントに分割することができる。例えばド・カステリョアルゴリズム（De Casteljau algorithm）に従って制御点を挿入することによって二次ベジェセグメントが分割される。これによって、自己交差輪郭を求めるときの性能が大幅に改善する。また、これによって、セグメントを内部又は外部としてラベル付けすることも単純かつ正確になる。
【００３１】
線形区画を用いてセグメントが近似される。１つの直線セグメントは単一の線形区画を用いて自明に近似される。１つの二次ベジェセグメントは１つ又は複数の線形区画を用いて近似される。二次ベジェセグメントを表すのに用いられる線形区画の制御点は、グリッド上の整数ロケーションに丸められる。目標レンダリングサイズを用いて、二次ベジェセグメントを表すのに必要な線形区画の数を制限し、それによって交差を計算する性能を改善する。線形区画は必要に応じて求められ、次に未来の処理のために保存される。線形区画の始点及び終点は一致してはならない。そのような区画は除去される。
【００３２】
接続点は特定の交差と関連付けられたデータを含む。接続点は、交差のデカルトロケーション（Cartesian location）を特定し、その交差から発する外部セグメントのリスト、すなわちアウトバウンド外部セグメントのリストを保持する。
【００３３】
非ゼロ巻数規則によって定義されるパスの変換
図２に示されるように、本発明のいくつかの実施形態は、非ゼロ巻数規則によって定義される入力パス２０１を、非ゼロ巻数規則又は奇数－偶数パリティ規則のいずれによってもレンダリングすることができる等価な出力パス２０２に変換する方法を提供する。パスは２次元（２Ｄ）座標系において定義される。ステップは図４において概略的に示される。
【００３４】
量子化及び変形
第１に、本方法は必要な場合、オプションで入力パスの非整数制御点を整数に変換する、すなわち量子化及び変形する（２０５）。例えば、ＴｒｕｅＴｙｐｅ（登録商標）フォ

ントは、ＥＭスクエアと呼ばれる十分な精度の整数座標系において定義され、したがって量子化も変形も必要としない。変換が必要なとき、浮動小数点又は固定小数点において指定される入力パスの制御点が変形され、例えば２５６を乗算され、次に丸められ、十分な精度の整数が求められる。

【００３５】

このステップをスキップすると、入力パスの制御点はそれらの元の座標系にとどまる。元の座標系は、いくつか例を挙げると、浮動小数点又は固定小数点グリッドにおいて定義することができる。量子化及び変形をスキップすることによって、より高い精度で算術演算を行うことができるが、これによって処理時間が増大し、出力パス２０２において欠陥が生じる場合がある。

【００３６】

縮退セグメントを置き換え又は除去し、オプションで単調性を強制

次に、整数変換後、縮退セグメントが除去される（２１０）。以下に縮退セグメントのいくつかの例を挙げる。セグメント始点はセグメント終点と一致してはならない。そのようなセグメントは破棄される。二次ベジェ曲線セグメントＢ（ｔ）（ただし、ｔは０～１（０及び１を含む）の範囲をとる）の始点及び終点は、曲線セグメントの「内部」部分、すなわち０＜ｔ＜１であるような時間パラメーターｔを有する曲線セグメント部分に位置することができない。そのようなセグメントは、直線セグメントと置き換えられる。二次ベジェ曲線セグメントのオフカーブ制御点はその二次ベジェ曲線セグメントの始点とも終点とも一致してはならない。そのようなセグメントは直線セグメントと置き換えられる。

【００３７】

オプションで、ｘ方向及びｙ方向の双方におけるセグメントの単調性が強制される（２１５）。

【００３８】

縮退輪郭の除去

縮退輪郭が除去される（２２０）。縮退輪郭のいくつかの例は、単一の点、開輪郭、すなわち「水密（watertight）」でない輪郭、及び内部エリアを有しない輪郭で構成される輪郭を含む。

【００３９】

性能を加速するデータ構造の構築

方法の実行性能を改善するデータ構造がオプションで構築される（２２５）。データ構造はステップ２３０及び２４０中にオンデマンドで及び必要なときにのみ構築することもできる。例示的なデータ構造は、境界ボックスと、近接性クラスターツリーと、セグメント及び輪郭のグリッドとを含む。

【００４０】

交差を求める

自己交差輪郭が求められる（２３０）。自己交差輪郭は、Ｐにおいて、Ｐの前後の一連の連続セグメントに新たな制御点を挿入することによって、自己交差の各点Ｐにおいて分割され、分けられる。オプションの量子化及び変形ステップが実行された場合、グリッドの整数座標において交差が求められる。セグメントが交差に寄与し得ることが可能であるとき、ベジェセグメントが必要に応じて線形区画に分割される。線形区画に対し交差試験が行われ、性能及び正確性が改善される。セグメントの単調性及びデータ構造の加速を用いてセグメント間交差テストを最小限にする。自己交差を含む入力パスの各輪郭をマーク付けする（２３５）。これに続いて、輪郭間交差が求められ（２４０）、これらもマーク付される（２４５）。この説明の目的で、何らかの方法で明示的にマーク付けされていない輪郭が「未マーク」とみなされる。

【００４１】

以下は、交差を求めるときの追加の規則及び慣例である。冗長な交差を回避するために、ｔ＝０において生じる交差はカウントされないが、ｔ＝１において生じる交差はカウントされる。２つの一致する直線セグメントは最大で２つの交差を有する。共通の終点にお

いて輪郭内で連結されている一連の直線セグメントは、その共通の終点において交差しない。交差輪郭が処理された後、生成されたいずれの線形区画ももはや用いられない。交差は輪郭に挿入され、例えばベジェセグメントがそれらの交差において分割され、分けられ、以下で説明するようにパスの輪郭がウォーキングされる。

【００４２】

輪郭交差ステップは、更なる交差が生じなくなるまで繰り返される。この繰り返しは、前回求められた交差の結果として新たな交差が生じ得ることに起因して行われる。このステップは、出力パス２０２が正しく、後続のレンダリング及び処理にアーティファクトが一切ないことを確実にする。ほとんどの入力パスが単一の反復しか必要としないことに留意されたい。

【００４３】

マーク付けされていない内部輪郭の除去

入力パス２０１のマーク付けされていない内部輪郭が除去される（２５０）。任意のセグメントの巻数を求め、例えば通常、入力パス２０１の各マーク付けされていない輪郭の第１のセグメントを用いて、セグメントが内部輪郭であり、ウォーキングステップの前に除去される必要があるか否かを判断する。最初に隠されている入力パス２０１の輪郭は、ウォーキングステップの後に明らかにすることができ、したがってここで除去される。

【００４４】

接続点の連結及び生成

次に、各交差のデカルトロケーションと関連付けられた接続点を生成することによって交差がリンク付けされる（２５５）。接続点は特定の交差に関するデータを含み、交差のデカルトロケーションを指定し、そのロケーションにおいて開始するセグメントのリストを保持する。一意の交差ごとに、すなわち一意の（ｘ，ｙ）座標を有する交差ごとに、外部アウトバウンドセグメントのリストＬが求められる。交差へのインバウンドセグメントごとに、Ｌへのポインターが保持される。外部アウトバウンドセグメントは、交差において開始するセグメントであり、交差から外向きに発し、そのセグメントの巻数は、そのセグメントが入力パス２０１の外部エッジであることを示す。アウトバウンドセグメントを外部として適切にラベル付けすることは重要なステップである。リンク付け２５５の後、Ｌのセグメントは適切にラベル付けされる。

【００４５】

ウォーキング

交差がリンク付けされた（２５５）後、パスの各マーク付けされた輪郭をウォーキング（２６０）して、マージによって新たな輪郭を生成することができる。ウォーキングはマーク付けされた輪郭上の任意の接続点において開始することができる。マーク付けされた輪郭のセグメントは、新たな輪郭にコピーされるか、又はセグメントが内部であるときに除去される。図７に示すように、ウォーキングプロセスの詳細なステップは以下の通りである。

１．マーク付けされた輪郭の交点ＩＰごとに、すなわち接続点ごとに、以下のステップを実行する：

２．ＬをＩＰの外部アウトバウンドセグメントのリストに設定する（７１０）。

３．Ｌが空の場合、ステップ１に進み、次の交点に移る（７１５）。

４．Ｌが空でない場合、Ｌの開始セグメントを取得し、除去する。このセグメントにＳを設定する（７２０）。

５．新たな輪郭Ｃを生成する（７３０）。

６．Ｃの最後にセグメントＳを付加する（７４０）。

７．Ｓの終点がＩＰの場合、トラバースしてＩＰまで戻ったことになる。ステップ３に進む（７５０）。

８．Ｓの終点が別の交点でない場合、次のセグメントを取得し、このセグメントをＳに設定し、ステップ６に移る（７５０）。

９．Ｓの終点がＩＰ以外の別の交点Ｑである場合、輪郭を以下のように切り替える：Ｑの

接続点Ｊを取得し、Ｊの外部アウトバウンドセグメントを取得及び除去し、この外部アウトバウンドセグメントをＳに設定し、ステップ６に移る（７５０）。

【００４６】

マーク付けされた輪郭の除去

次に、入力パス２０１のマーク付けされた輪郭が除去される（２６５）。出力パス２０２に寄与するマーク付けされた輪郭のセグメントは前のウォーキングステップにおいて新たな輪郭にコピーされている。

【００４７】

縮退輪郭の除去

次に、残りの輪郭の縮退輪郭が除去される（２６８）。縮退輪郭の例には、単一点を有する輪郭、開輪郭、及び内部エリアを有しない輪郭が含まれる。

【００４８】

出力パスの形成

最後に、新たな輪郭及びマーク付けされていない輪郭が出力パス２０２として収集される（２７０）。単純化された出力パスは、非ゼロ巻数規則又は奇数－偶数パリティ規則のいずれによってもレンダリングすることができる。

【００４９】

量子化及び変形の反転

ステップ２０５において、入力パス２０１がオプションで量子化及び変形された場合、そのプロセスは出力パス２０２において反転され（２７５）、座標が元の座標系に復元される。

【００５０】

変換の効果

図５は、入力パス２０１を左に、出力パス２０２を右に、パス５０１～５０６の例示的な対を示している。例は、出力パス２０２が非ゼロ巻数規則又は奇数－偶数パリティ規則のいずれを用いても正しくレンダリングされることを示している。入力パス２０１における一致セグメント等の縮退事例の存在、及びそれらを適切に処理して正しい出力パス２０２が生成されることに留意されたい。

【００５１】

マージ輪郭手順はオブジェクトベースである。この手順は、更なる処理のための２Ｄ出力パス２０２を明示的に求める。従来技術による画像ベースの手法は、この動作をラスター化中に行うので、ピクセルのみが結果を示し、本明細書において説明されるような明示的な２Ｄ出力次元パスは形成されない。

【００５２】

マージ輪郭手順への入力は、グリフを表すパス等の単一の有向２Ｄパスである。パスの内部は非ゼロ巻数規則によって画定される。出力は、非ゼロ巻数規則又は奇数－偶数パリティ規則のいずれによってもレンダリングすることができる等価で単純化された２Ｄパスである。単純化は、重複する輪郭、自己交差輪郭、縮退輪郭及び不要な輪郭の除去を含むことができる。

【００５３】

パス変換：例

図４は、入力パス２０１を変換する方法の動作を概略的に示している。入力パス２０１は、外部輪郭４０１と、内部輪郭４０２と、２つの交差する輪郭４０３及び４０４とを含む。輪郭４０３は自己交差もしている（４１１）。輪郭４０３及び４０４は２つの輪郭間交差４１２を有する。交差を有する輪郭４０３及び４０４がマーク付けされる（Ｍ）。マーク付けされていない内部輪郭４０２は除去される。次に、接続点が交差と関連付けられる。各接続点におけるアウトバウンドセグメント４２０は内部又は外部である。各接続点における外部アウトバウンドセグメントはラベル付けされて、「｜｜」によって示される。

【００５４】

ウォーキング中、輪郭をマージすることによって新たな輪郭４３０が生成される一方、マーク付けされた輪郭及び縮退輪郭が除去される。

【００５５】

上記の方法及び本明細書において説明する任意の他の手順又は方法のステップは、当該技術分野において既知のメモリ及び入出力インターフェース並びにデバイスに接続されたプロセッサにおいて実行することができる。

【００５６】

レンダリング：アーティファクトのないパスの塗りつぶし又は外形描画

上記で生成されるような出力パス２０２は、非ゼロ巻数規則又は奇数－偶数パリティ規則のいずれかによって塗りつぶすか又は外形を描くことによってレンダリングすることができる（２８０）。アンチエイリアシング２８５も適用することができる。従来技術と対照的に、塗りつぶされたパスは内部エイリアシングアーティファクトを一切含まない。同様に、外形を描かれたパスは内部エッジアーティファクトを一切含まない。

【００５７】

内部及び外部セグメントのラベル付け

図３は、セグメントＳが内部にあるか又は外部にあるかを判断する手順を示している。第１に、巻数Ｗがゼロに初期化される（３０５）。次に、セグメントに交差することが保証される走査線が特定される（３１０）。ｘ方向及びｙ方向の双方においてセグメントの単調性が強制される場合、走査線は、セグメントの中点を通過する水平走査線又は垂直走査線のいずれかであると迅速かつ正確に特定することができる。

【００５８】

輪郭Ｃごとに、セグメントＳに交わる前に特定された走査線に交わる輪郭Ｃの任意のセグメントについて巻値がＷに累積される（３１５）。セグメントＳと同じロケーションにおいて走査線に交わる輪郭Ｃの一致セグメントについて巻値をＷに累積することは、全ての輪郭が処理された後まで延期される。一致セグメントは、巻数に関する限りグループとして扱われなくてはならない。なぜならそれらは重複し、互いを相殺するためである。一致セグメントはマーク付けされ、リストＬ内に保存される（３２０）。一致セグメントについては図８も参照されたい。

【００５９】

一致セグメントが存在しない場合、以下のように進行する。Ｗが非ゼロであり、かつＷがＳを計上するように更新されたときに非ゼロである場合（３４５）、セグメントＳは内部であるとマーク付けされ、そうでない場合、セグメントＳは外部であるとマーク付けされる（３５０）。

【００６０】

一致セグメントが存在するとき、以下のように進行する。保存された一致セグメントのリストＬにセグメントＳを加える（３３５）。リストＬ内の互いに反対方向を有する一致セグメントの対は相殺し合い、それらの巻数の累積はゼロでありＷが変化しない。巻値はセグメントＳを除くリストＬ内の残りの相殺されていないセグメントについてＷに累積される（３４０）。Ｗが非ゼロであり、かつ、Ｓを計上するように更新されたときにＷが非ゼロである場合（３４５）、セグメントＳは内部であるとマーク付けされ、そうでない場合、セグメントＳは外部であるとマーク付けされる（３５０）。

【００６１】

従来技術による方法は上記で説明したような一致セグメントを処理せず、したがって内部アーティファクトを有する不正確な出力パスを生成する可能性があることに留意されたい。

【００６２】

図８は、出力パス２０２が正しくなるような入力パス２０１の輪郭８０１及び８０２のアウトバウンドセグメントのラベル付けを概略的に示している。輪郭は一致セグメント８０３を有する。外部アウトバウンドセグメントは「｜｜」によって示される。上記で概説したステップの特性に起因して、セグメント８０３は内部として正しく識別され、このた

めこれらのセグメントはウォーキング中に除去される。本発明者らの知る限り、従来技術によるパスレンダリングはいずれも、この事例にその全ての変形において正しく対処するわけではない。この事例では、一致セグメントは巻数が求められるときにグループとして扱われる必要がある。

【００６３】

応用形態及び顕著な特徴

奇数－偶数パリティ規則に従ってパスを塗りつぶすレンダリング方法は、図２において概説したような輪郭マージ手順を前処理ステップとして用いて、非ゼロ巻数規則に従って塗りつぶされるように設計されたパス、例えばＴｒｕｅＴｙｐｅ（登録商標）フォントにおけるグリフを正しくレンダリングすることができる。

【００６４】

輪郭マージ手順を用いて、様々なフォントレンダリングシステム内に存在する内部エッジハロー及び外形描画アーティファクトを取り除くこともできる。奇数－偶数パリティ規則は非ゼロ巻数規則よりも複雑でなく、実行が高速である。したがって、輪郭マージ手順を用いて入力パスを等価な出力パスに変換し、次に奇数－偶数パリティ規則を用いてレンダリングすることができる。

【００６５】

輪郭マージ手順は、２次元パスにおける、論理和、論理積、論理差等の２次元空間領域構成法（ＣＳＧ：constructive solid geometry）演算を行うのに用いることもできる。

【００６６】

輪郭マージ手順は、リアルタイムで非常に困難な計算幾何学問題を解き、従来技術と比較すると幾つかの顕著な特徴を有する。輪郭マージ手順は、困難なグレージング条件及び特異性、例えば一致点及び一致セグメントを処理することができる。手順は、非線形曲線エッジ及び有向エッジをサポートする。ほとんどの従来技術による手順は、線形エッジを有する多角形に対してのみ機能する。輪郭マージ手順は、整数計算を用いて整数グリッドに対して演算を行うことができる。これは従来技術による同等の手順よりも１桁～２桁高速である。

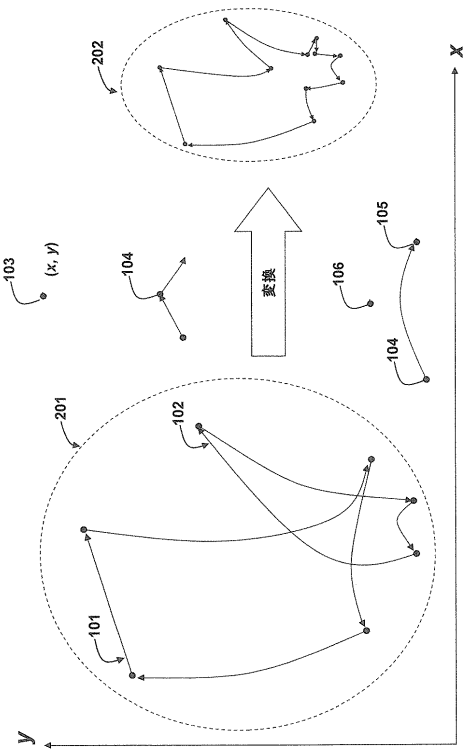
【００６７】

輪郭マージ手順は、グレージング条件、セグメント及び輪郭の縮退前処理及び後処理、並びに交差を求めるための曲線セグメントのオンデマンドテッセレーションを適切に処理するように、一致エッジの同時分類も提供する。曲線セグメントのテッセレーションは目標レンダリングサイズに基づくことができ、したがって所与の目標の性能を最適化することができる。より良好な性能及び正確性のためにセグメントの単調性を強制することができる。マージ中、全ての制御点を整数座標系上に表し、正確性及び性能を改善することができる。輪郭セグメントが交差において分割され、分けられると、手順は収束するまで反復交差テストを用いてロバスト性を確保することができる。

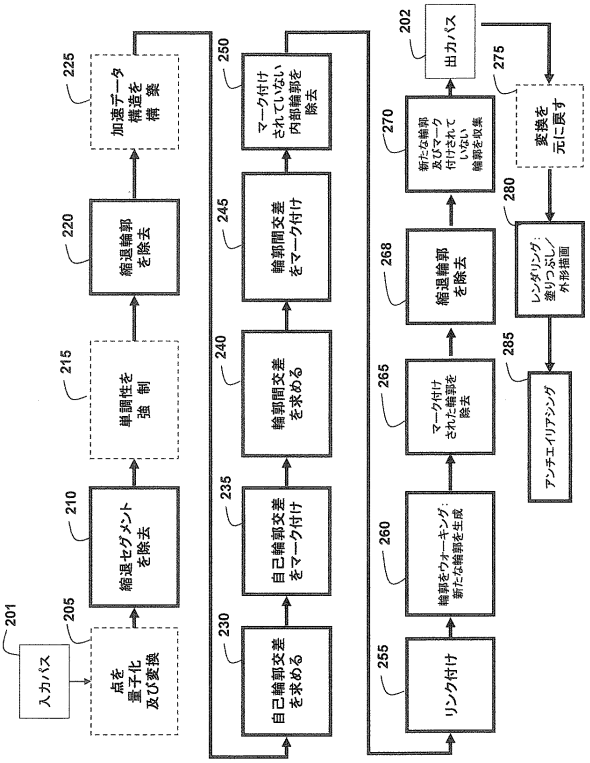
【００６８】

セグメントを内部又は外部として正しくラベル付けすることによって、パスが縮退事例を含むときに、パスレンダリングシステム、パス圧縮システム、パス単純化システム等が正しく機能することが可能になる。

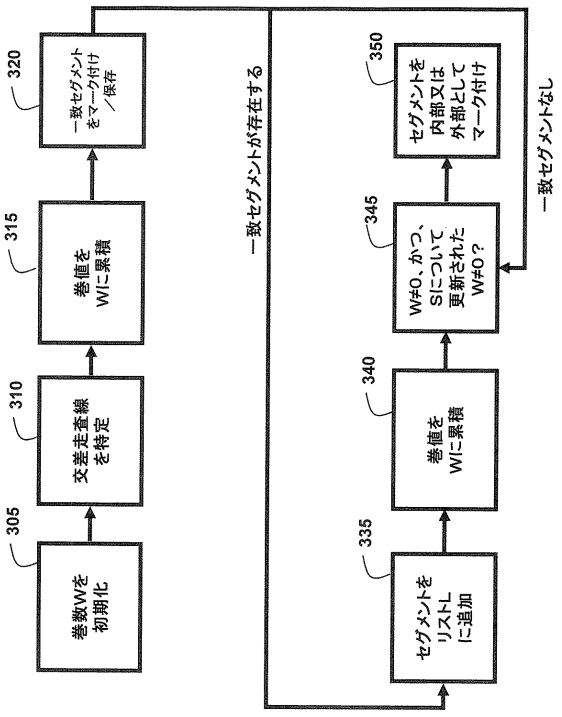
【図１】


202
X
103
(x, y)
104
変換
106
105
104
201
102
101
y

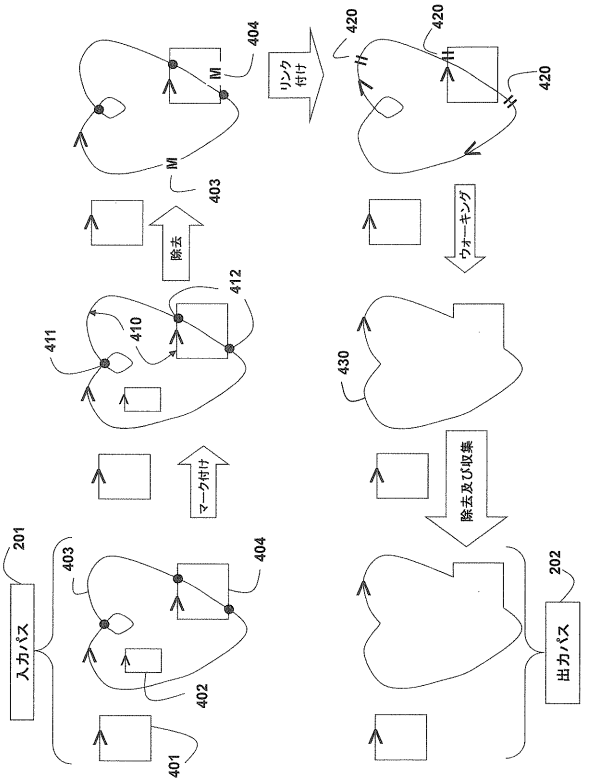
【図２】


201
入力パス
205
点を量子化及び変換
210
縮退セグメントを除去
215
単調性を強制
220
縮退輪郭を除去
225
加速データ構造を構築
230
自己輪郭交差を求める
235
自己輪郭交差をマーク付け
240
輪郭間交差を求める
245
輪郭間交差をマーク付け
250
マーク付けされていない内部輪郭を除去
255
リンク付け
260
輪郭をウォーキング：新たな輪郭を生成
265
マーク付けされた輪郭を除去
268
縮退輪郭を除去
270
新たな輪郭及びマーク付けされていない輪郭を収集
275
変換を元に戻す
280
レンダリング：塗りつぶし／外形描画
285
アンチエイリアシング
202
出力パス

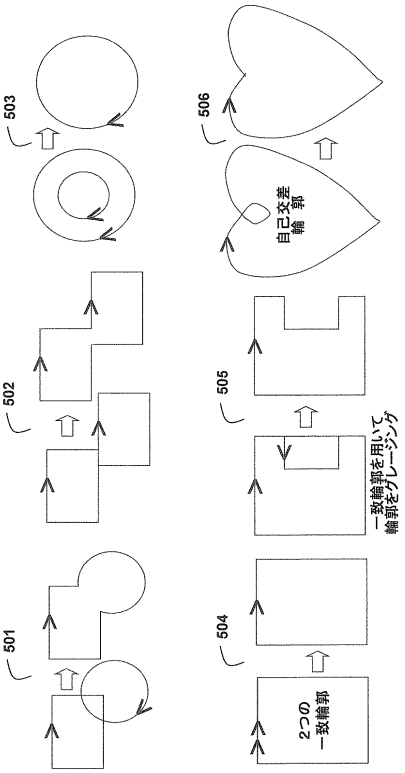
【図３】


305
巻数Wを初期化
310
交差走査線を特定
315
巻値をWに累積
320
一致セグメントをマーク付け／保存
一致セグメントが存在する
335
セグメントをリストに追加
340
巻値をWに累積
345
W≠0、かつ、Siについて更新されたW≠0？
350
セグメントを内部又は外部としてマーク付け
一致セグメントなし

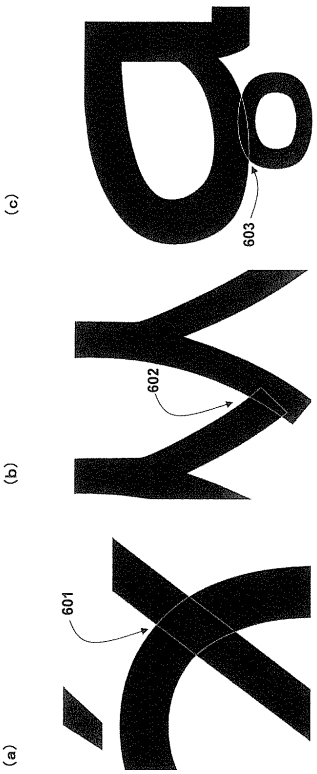
【図４】


201
入力パス
403
404
402
401
マーク付け
除去
リンク付け
ウォーキング
除去及び収集
202
出力パス
420
430
410
411
412
M

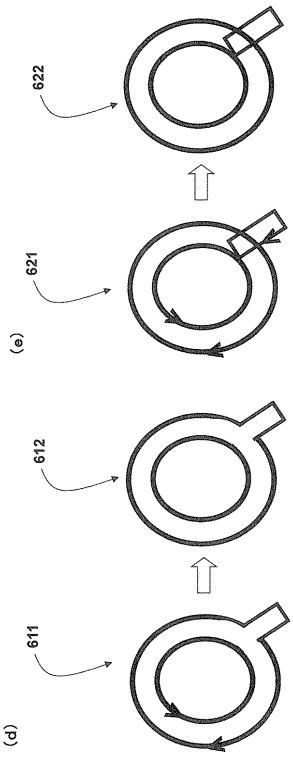
【図５】


503
506
自己交差
輪郭
502
505
一致輪郭を用いて
輪郭をグルーピング
501
504
2つの
一致輪郭

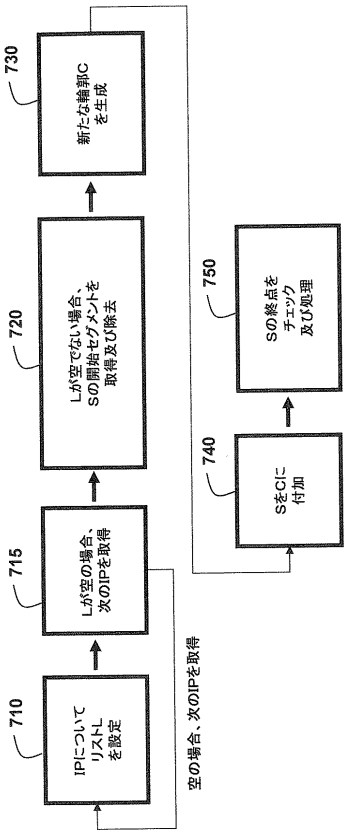
【図６Ａ】


(c)
603
(b)
602
(a)
601

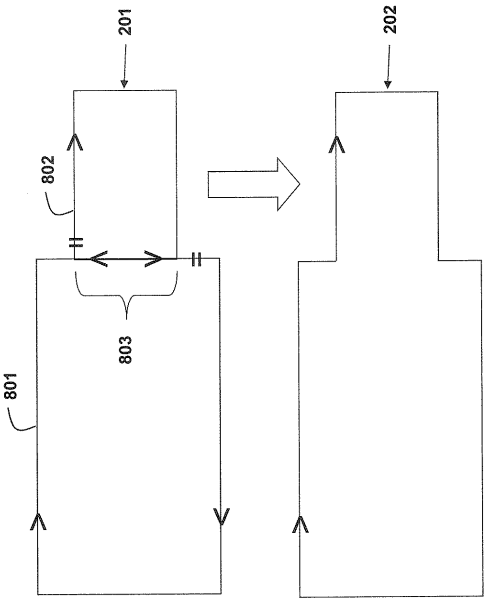
【図６Ｂ】


622
(e)
621
612
(d)
611

【図７】


730
新たな輪郭Cを生成
720
Lが空でない場合、
Sの開始セグメントを
取得及び除去
750
Sの終点を
チェック
及び処理
715
Lが空の場合、
次のIPを取得
740
SをCに
付加
710
IPについて
リストL
を設定
空の場合、次のIPを取得

【図８】



フロントページの続き

(31)優先権主張番号　13/942,837
(32)優先日　平成25年7月16日(2013.7.16)
(33)優先権主張国　米国(US)

早期審査対象出願

(72)発明者　ロナルド・エヌ・ペリー
アメリカ合衆国、マサチューセッツ州、ケンブリッジ、ブロードウエイ　２０１、ミツビシ・エレクトリック・リサーチ・ラボラトリーズ・インコーポレイテッド
(72)発明者　エレナ・ジェイ・ジャクビアク
アメリカ合衆国、マサチューセッツ州、ケンブリッジ、ブロードウエイ　２０１、ミツビシ・エレクトリック・リサーチ・ラボラトリーズ・インコーポレイテッド

審査官　村松　貴士

(56)参考文献　特開平１０－０２１４１５（ＪＰ，Ａ）
特開２００５－１２２３０２（ＪＰ，Ａ）
国際公開第２０１０／１３４１２４（WO，Ａ１）
米国特許第０６１７３０７５（ＵＳ，Ｂ１）
米国特許第０５３１９３４６（ＵＳ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｔ　１１／００　－　１１／８０
Ｇ０９Ｇ　５／００　－　５／４２
Ｇ０６Ｆ　３／０４８　－　３／０４８９