(19) IPOS INTELLECTUAL PROPERTY OFFICE OF SINGAPORE

(11) Publication number: **SG 174917 A1**
(43) Publication date: **28.11.2011**
(51) Int. Cl: ;

(12) Patent Application

(21) Application number: **2011069481**
(22) Date of filing: **23.04.2010**
(30) Priority: **US 12/433,288 30.04.2011**

(71) Applicant: **MICROSOFT CORPORATION ONE MICROSOFT WAY REDMOND, WASHINGTON 98052-6399 WH US**

(72) Inventor: **ALLYN, BARRY CHRISTOPHER C/O MICROSOFT CORPORATION LCA - INTERNATIONAL PATENTS ONE MICROSOFT WAY REDMOND, WASHINGTON 98052-6399 US**
RUBLE, B. SCOTT C/O MICROSOFT CORPORATION LCA - INTERNATIONAL PATENTS ONE MICROSOFT WAY REDMOND, WASHINGTON 98052-6399 US

(54) **Title:**
DATA VISUALIZATION PLATFORM PERFORMANCE OPTIMIZATION

(57) **Abstract:**
Data visualization platform optimization may be provided. Applications may provide data values and request creation of a visualization from a data visualization platform (DVP). The DVP may composite a plurality of geometry records associated with a subset of the visualization's data values. The application may render the visualization by iterating through the geometry vectors and translating a subset of the vectors into drawing instructions for output to a display device.

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
4 November 2010 (04.11.2010)

PCT

(10) International Publication Number
WO 2010/126802 A3

(51) International Patent Classification:
G06F 3/14 (2006.01) *G06F 9/06* (2006.01)

(21) International Application Number:
PCT/US2010/032307

(22) International Filing Date:
23 April 2010 (23.04.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/433,288 30 April 2009 (30.04.2009) US

(71) Applicant *(for all designated States except US)*: **MICROSOFT CORPORATION** [US/US]; One Microsoft Way, Redmond, Washington 98052-6399 (US).

(72) Inventors: **ALLYN, Barry Christopher**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **RUBLE, B. Scott**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

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

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(88) Date of publication of the international search report:
3 February 2011

(54) Title: DATA VISUALIZATION PLATFORM PERFORMANCE OPTIMIZATION

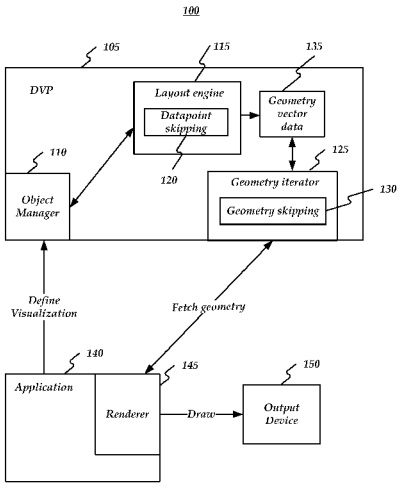

FIG. 1

(57) Abstract: Data visualization platform optimization may be provided. Applications may provide data values and request creation of a visualization from a data visualization platform (DVP). The DVP may composite a plurality of geometry records associated with a subset of the visualization's data values. The application may render the visualization by iterating through the geometry vectors and translating a subset of the vectors into drawing instructions for output to a display device.

WO 2010/126802 A3

DATA VISUALIZATION PLATFORM PERFORMANCE OPTIMIZATION

BACKGROUND

[001] Data visualization platform performance optimizations provide improved performance in the generation of visual objects. In some situations, generating visual objects may be very performance intensive for a computer. For example, generating charts and maps may require a great deal of computational power and/or memory, especially when the object comprises a large number of data points. The conventional strategy is to render every data point, regardless of the size of the resulting object. This may cause problems because certain operations on the object may cause the computer to become sluggish to respond. For example, displaying a large map, selecting a portion of a detailed chart, scrolling, printing, and/or modifying the object may require a great deal of computing power and may cause the computer to respond slowly.

SUMMARY

[002] Data visualization platform performance optimizations may be provided. This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter. Nor is this Summary intended to be used to limit the claimed subject matter's scope.

[003] Data visualization platform optimization may be provided. Applications may provide data values and request creation of a visualization from a data visualization platform (DVP). The DVP may composite a plurality of geometry records associated with a subset of the visualization's data values. The application may render the visualization by iterating through the geometry vectors and translating a subset of the vectors into drawing instructions for output to a display device.

[004] Both the foregoing general description and the following detailed description provide examples and are explanatory only. Accordingly, the foregoing general description and the following detailed description should not be considered to be restrictive. Further, features or variations may be provided in addition to those set forth herein. For example, embodiments may be directed to various feature combinations and sub-combinations described in the detailed description.

BRIEF DESCRIPTION OF THE DRAWINGS

[005] The accompanying drawings, which are incorporated in and constitute a part of this disclosure, illustrate various embodiments of the present invention. In the drawings:

[006] FIG. 1 is a block diagram of an operating environment;

[007] FIG. 2 is a flow chart of a method for optimizing visualization platform performance; and

[008] FIG. 3 is a block diagram of a system including a computing device.

DETAILED DESCRIPTION

[009] The following detailed description refers to the accompanying drawings. Wherever possible, the same reference numbers are used in the drawings and the following description to refer to the same or similar elements. While embodiments of the invention may be described, modifications, adaptations, and other implementations are possible. For example, substitutions, additions, or modifications may be made to the elements illustrated in the drawings, and the methods described herein may be modified by substituting, reordering, or adding stages to the disclosed methods. Accordingly, the following detailed description does not limit the invention. Instead, the proper scope of the invention is defined by the appended claims.

[010] Data visualization platform (DVP) performance optimization may be provided. Consistent with embodiments of the present invention, a DVP may skip datapoints during composition of an object according to a display resolution and may construct a minimal set of geometry vectors in memory. The DVP may also skip composed vectors during rendering of the object. The DVP may further allow scaling of the algorithms for skipping datapoints and/or vectors to expose different quality settings, such as by using an application programming interface (API) allowing control over settings such as maximum memory consumption, time, and/or datapoint counts.

[011] FIG. 1 is a block diagram of an operating environment 100 for providing a visualization platform 105. Visualization platform 105 may comprise an object module 110, a layout engine 115, and a geometry iterator 125. Layout engine 115 may comprise a datapoint skipping algorithm 120. Geometry iterator 125 may comprise a geometry skipping algorithm 130. Geometry iterator 125 and layout engine 115 may create, update, read, retrieve, and/or provide a plurality of geometry vector data 135. Operating environment 100 may further comprise an application 140 comprising a renderer 145 operative to render drawing instructions to an output device 150 such as a monitor, screen, printer, and/or other display device. Visualization platform 105 may comprise an architecture allowing the creation of a data visualization, such as a chart, and may expose an interactive feature on the visualization. The architecture may provide integration with

multiple rendering platforms. When a user selects the exposed feature, the architecture may translate the selection into a common format and modify the data visualization according to layout rules independent of the rendering platform.

[012] Application 140 may define a visualization through object module 110. Object module 110 may call layout engine 115 that may build a collection of geometry records in memory as geometry vector data 135. Application 140 may later need to render the visualization and may call into geometry iterator 125 that may access geometry vector data 135 and return it to application 140. Application 140 may translate each geometry primitive in geometry vector data 135 into drawing instructions for rendering to output device 150.

[013] Visualization platform 105 may comprise a shared core comprising software libraries and/or utilities for providing interactive visualizations. The shared core may be implemented, for example, in C++ or C#, and may be platform independent. The shared core may comprise visualization utilities for providing layouts, shapes and/or geometry, line services, 3-dimensional rendering, animation frame generation, and/or interactive hotspots. Visualization platform 110 may further comprise application programming interfaces (APIs) for interacting with application 140.

[014] Two areas for performance optimization may be during the composition of geometry vector data 135 and translation of geometry vector data 135 into drawing instructions for rendering. For example, during composition of geometry vector data 135, memory usage may increase greatly as the amount of data increases. The amount of memory required may be reduced through the use of data point skipping algorithm 120 so as to composite geometry vectors for a subset of the data. Geometry skipping algorithm 130 may operate to reduce a number of drawing instructions required to be rendered by translating a subset of geometry vector data 135.

[015] FIG. 2 is a flow chart setting forth the general stages involved in a method 200 consistent with an embodiment of the invention for providing data visualization platform optimizations. Method 200 may be implemented using a computing device 300 as described in more detail below with respect to FIG. 3. Ways to implement the stages of method 200 will be described in greater detail below. Method 200 may begin at starting block 205 and proceed to stage 210 where computing device 300 may define a visualization. For example, application 140 may send a request to visualization platform 105 to create a visual object. The request may comprise a data value, a data series, and/or

an object type such as a line chart, a bar chart, a pie chart, or a graph. Visualization platform 105 may use object manager to define a memory location for the visual object.

[016] From stage 210, where computing device 300 defined the visualization, method 200 may advance to stage 220 where computing device 300 may composite at least one geometry vector associated with the visualization. For example, application 140 may define a visualization of a chart comprising 1,000 data points wherein each data point may be represented by an drawn octagonal shape. In conventional systems, visualization platform 105 may composite geometry vectors for each of the 1,000 data points. Consistent with embodiments of the invention, layout engine 115 may use datapoint skipping algorithm 120 to composite geometry vectors for a subset of the 1,000 data points. This may allow the display of a draft and/or preview quality visual object. Further consistent with embodiments of the invention, layout engine 115 may be operative to determine whether data points within the 1,000 data points are close enough to each other so as to be indistinguishable based on characteristics of application 140 and/or output device 150. For example, if a subset of 30 data points within the 1,000 data points all overlap or come within a predefined threshold of each other, such as within 5 pixels, datapoint skipping algorithm 120 may be operative to skip compositing all of the 30 data points and may only composite one of the 30 data points.

[017] Consistent with embodiments of the invention, application 140 may define a column chart comprising 10,000 data points. Layout engine 115 may call a column series class that may walk through the data and composite vectors for boxes for the data points and put them into geometry vector data 135. Layout engine 115 may receive, from application 140 for example, a chart size in pixels and determine how many data points may be displayed in that size of the chart. For example, a chart 300 pixels wide may only be able to display 300 data points without overlapping. Layout engine 135 may thus composite vector data for a 300 point subset of the 10,000 data points. The size of the subset may be based, for example, on the resolution of output device 150.

[018] The size of the subset may also be based, for example, on a determination by layout engine 115 that certain data points will be covered by other data points and layout engine 115 may thus skip compositing vector data for covered data points. Consistent with embodiments of the invention, layout engine 115 may determine that a later data point occupies the exact same pixel as a previous data point and thus the later data point may be skipped. Consistent with further embodiments of the invention, datapoint skipping

algorithm 120 may use a configurable criterion to determine whether the later point is close enough to the previous data point to be skipped. For example, layout engine 115 may set the criterion to skip data points within five pixels of a previous data point. Further consistent with embodiments of the invention, the criterion may be raised as the composition stage proceeds in order to limit the amount of memory consumed by the composition of geometry vector data 135. The criterion may also be received from application 140, such as through a user selection of a user interface throttle control.

[019] Once computing device 300 composites the geometry vectors in stage 220, method 200 may continue to stage 230 where computing device 300 may receive a request to render the visualization. For example, application 140 may request rendering of the visualization according to a user command.

[020] After computing device 300 receives the render request in stage 230, method 200 may proceed to stage 240 where computing device 300 may iterate through composited geometry vectors and translate them into drawing instructions. For example, visualization platform 105 may translate a geometry vector comprising an x-axis of a chart object into a drawing instruction for a line, wherein the drawing instruction comprises associated data such as a starting point for the line, an ending point for the line, a color for the line, and/or a width for the line.

[021] Consistent with embodiments of the invention, visualization platform 105 may receive at least one supported drawing instruction from the application. Visualization platform 105 may be operative to translate the composited geometry vectors into the supported drawing instructions. For example, visualization platform 105 may be operative to linearize a geometry vector comprising an octagon into a drawing instructions comprising a start and end point for each of a set of eight lines used to render an octagonal shape on output device 150. Application 140 may instead or additionally inform visualization platform of a supported drawing instruction comprising a center point and a width for an octagon shape such that visualization platform 105 may translate the geometry vector comprising an octagon into the supported drawing instruction rather than linearizing the octagon shape. Application 140 may thus reduce the number of translated drawing instructions visualization platform 105 may need to provide.

[022] Computing device 300 may utilize geometry skipping algorithm 130 to reduce the number of geometry vectors that need to be translated and rendered. For example, if the visualization comprises a chart of 100 pixels by 200 pixels using octagonal shapes for data

points, each octagon may comprise a shape 10 pixels by 10 pixels or smaller and some line segments may comprise zero lengths. Geometry skipping algorithm 130 may, for example, determine that a particular geometry vector is associated with a criterion for being skipped, such as the vector may be translated into a drawing instruction for a line having the same start and end point. In such an example, geometry skipping algorithm 130 may skip over the particular geometry vector rather than translate it into a drawing instruction and sending it to application 140 for rendering. Geometry skipping algorithm 130 may skip one of the geometry vectors according to many other criteria such as a display size, a data type, a visualization type, a total number of data points, and a total number of geometry vectors.

[023] Consistent with embodiments of the invention, application 140 may fetch the drawing instructions in batches. For example, application 140 may allocate enough memory to receive 50 of the drawing instructions. Visualization platform 105 may then provide 50 drawing instructions and inform application 140 whether more drawing instructions are waiting. Visualization platform 105 may wait for application 140 to request a subsequent batch and repeat the process.

[024] From stage 240, method 200 may advance to stage 250 where computing device 300 may render the visualization to an output device. For example, application 140 may receive the drawing instructions at renderer 145 and draw the visualization on a screen comprising output device 150.

[025] Once computing device 300 renders the visualization in stage 250, method 200 may advance to stage 260 where computing device 300 may determine whether a change has occurred in the display. For example, the displayed visualization may be moved, zoomed, re-sized, and/or obscured wholly or in part. If computing device determines that a change has occurred in the display of the visualization, method 300 may return to stage 250 where computing device 300 may re-render the visualization. Data point skipping may happen during composition at stage 220 when geometry vector data 135 is cached, while geometry skipping may happen each time the visualization is rendered in stage 250. So, for example, each time application 140 has to render the visualization, such as when a window is moved, minimized, maximized, or zoomed, application 140 may receive a new set of drawing instructions from geometry iterator 125. When application 140 zooms out, for example, fewer pixels may be displayed in the visualization and the geometry skipping may be more aggressive, skipping more geometry vectors.

[026] Consistent with embodiments of the invention, data point skipping and geometry skipping may be used separately and/or together and may be controlled through a user exposed configuration preference. For example, a user wishing to zoom into great detail on a chart may disable data point skipping in order to provide the greatest amount of data while enabling geometry skipping to prevent the translation of vectors not visible in the zoomed in state.

[027] Once computing device 300 has determined whether a change has occurred in the display at stage 260 and re-rendered the visualization if needed, method 200 may end at stage 270.

[028] An embodiment consistent with the invention may comprise a system for providing visualization platform optimization. The system may comprise a memory storage and a processing unit coupled to the memory storage. The processing unit may be operative to define a visualization, create a plurality of geometry records, receive a request to display the visualization, iterate through the plurality of geometry vectors, translate at least one of the plurality of geometry records into at least one drawing instruction, and display the visualization on a display device.

[029] Another embodiment consistent with the invention may comprise a system for providing visualization platform optimization. The system may comprise a memory storage and a processing unit coupled to the memory storage. The processing unit may be operative to create a visualization object associated with a plurality of data values, composite at least one first value of the plurality of data values into a geometry vector associated with the visualization object, and skip compositing at least one second value of the plurality of data values.

[030] Yet another embodiment consistent with the invention may comprise a system for providing optimizing a visualization platform. The system may comprise a memory storage and a processing unit coupled to the memory storage. The processing unit may be operative to receive a request from a user application to create a visualization object, composite a plurality of geometry vectors associated with the visualization object, receive a render request from the user application, translate each of a subset of the plurality of geometry vectors into at least one drawing instruction, send the at least one drawing instruction associated with each of the subset of the plurality of geometry vectors to the user application and render the visualization object. The processing unit may be further operative to receive a second render request from the user application in response to a

detected change affecting the rendered visualization object, translate each of a second subset of the plurality of geometry vectors into at least one drawing instruction, send the at least one drawing instruction associated with each of the second subset of the plurality of geometry vectors to the user application, and re-render the visualization object.

[031] FIG. 3 is a block diagram of a system including computing device 300. Consistent with an embodiment of the invention, the aforementioned memory storage and processing unit may be implemented in a computing device, such as computing device 300 of FIG. 3. Any suitable combination of hardware, software, or firmware may be used to implement the memory storage and processing unit. For example, the memory storage and processing unit may be implemented with computing device 300 or any of other computing devices 318, in combination with computing device 300. The aforementioned system, device, and processors are examples and other systems, devices, and processors may comprise the aforementioned memory storage and processing unit, consistent with embodiments of the invention. Furthermore, computing device 300 may comprise an operating environment for system 100 as described above. System 100 may operate in other environments and is not limited to computing device 300.

[032] With reference to FIG. 3, a system consistent with an embodiment of the invention may include a computing device, such as computing device 300. In a basic configuration, computing device 300 may include at least one processing unit 302 and a system memory 304. Depending on the configuration and type of computing device, system memory 304 may comprise, but is not limited to, volatile (e.g., random access memory (RAM)), non-volatile (e.g., read-only memory (ROM)), flash memory, or any combination. System memory 304 may include operating system 305, one or more programming modules 306, and may include visualization platform 110. Operating system 305, for example, may be suitable for controlling computing device 300's operation. In one embodiment, programming modules 306 may include user application 140. Furthermore, embodiments of the invention may be practiced in conjunction with a graphics library, other operating systems, or any other application program and is not limited to any particular application or system. This basic configuration is illustrated in FIG. 3 by those components within a dashed line 308.

[033] Computing device 300 may have additional features or functionality. For example, computing device 300 may also include additional data storage devices (removable and/or non-removable) such as, for example, magnetic disks, optical disks, or tape. Such

additional storage is illustrated in FIG. 3 by a removable storage 309 and a non-removable storage 310. Computer storage media may include volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information, such as computer readable instructions, data structures, program modules, or other data. System memory 304, removable storage 309, and non-removable storage 310 are all computer storage media examples (i.e., memory storage). Computer storage media may include, but is not limited to, RAM, ROM, electrically erasable read-only memory (EEPROM), flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store information and which can be accessed by computing device 300. Any such computer storage media may be part of device 300. Computing device 300 may also have input device(s) 312 such as a keyboard, a mouse, a pen, a sound input device, a touch input device, etc. Output device(s) 314 such as a display, speakers, a printer, etc. may also be included. The aforementioned devices are examples and others may be used.

[034] Computing device 300 may also contain a communication connection 316 that may allow device 300 to communicate with other computing devices 318, such as over a network in a distributed computing environment, for example, an intranet or the Internet. Communication connection 316 is one example of communication media. Communication media may typically be embodied by computer readable instructions, data structures, program modules, or other data in a modulated data signal, such as a carrier wave or other transport mechanism, and includes any information delivery media. The term "modulated data signal" may describe a signal that has one or more characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media may include wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, radio frequency (RF), infrared, and other wireless media. The term computer readable media as used herein may include both storage media and communication media.

[035] As stated above, a number of program modules and data files may be stored in system memory 304, including operating system 305. While executing on processing unit 302, programming modules 306 (e.g., user application 140) may perform processes including, for example, one or more method 200's stages as described above. The aforementioned process is an example, and processing unit 302 may perform other

processes. Other programming modules that may be used in accordance with embodiments of the present invention may include electronic mail and contacts applications, word processing applications, spreadsheet applications, database applications, slide presentation applications, drawing or computer-aided application programs, etc.

[036] Generally, consistent with embodiments of the invention, program modules may include routines, programs, components, data structures, and other types of structures that may perform particular tasks or that may implement particular abstract data types. Moreover, embodiments of the invention may be practiced with other computer system configurations, including hand-held devices, multiprocessor systems, microprocessor-based or programmable consumer electronics, minicomputers, mainframe computers, and the like. Embodiments of the invention may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in both local and remote memory storage devices.

[037] Furthermore, embodiments of the invention may be practiced in an electrical circuit comprising discrete electronic elements, packaged or integrated electronic chips containing logic gates, a circuit utilizing a microprocessor, or on a single chip containing electronic elements or microprocessors. Embodiments of the invention may also be practiced using other technologies capable of performing logical operations such as, for example, AND, OR, and NOT, including but not limited to mechanical, optical, fluidic, and quantum technologies. In addition, embodiments of the invention may be practiced within a general purpose computer or in any other circuits or systems.

[038] Embodiments of the invention, for example, may be implemented as a computer process (method), a computing system, or as an article of manufacture, such as a computer program product or computer readable media. The computer program product may be a computer storage media readable by a computer system and encoding a computer program of instructions for executing a computer process. The computer program product may also be a propagated signal on a carrier readable by a computing system and encoding a computer program of instructions for executing a computer process. Accordingly, the present invention may be embodied in hardware and/or in software (including firmware, resident software, micro-code, etc.). In other words, embodiments of the present invention may take the form of a computer program product on a computer-usable or computer-

readable storage medium having computer-usable or computer-readable program code embodied in the medium for use by or in connection with an instruction execution system. A computer-usable or computer-readable medium may be any medium that can contain, store, communicate, propagate, or transport the program for use by or in connection with the instruction execution system, apparatus, or device.

[039] The computer-usable or computer-readable medium may be, for example but not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, device, or propagation medium. More specific computer-readable medium examples (a non-exhaustive list), the computer-readable medium may include the following: an electrical connection having one or more wires, a portable computer diskette, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), an optical fiber, and a portable compact disc read-only memory (CD-ROM). Note that the computer-usable or computer-readable medium could even be paper or another suitable medium upon which the program is printed, as the program can be electronically captured, via, for instance, optical scanning of the paper or other medium, then compiled, interpreted, or otherwise processed in a suitable manner, if necessary, and then stored in a computer memory.

[040] Embodiments of the present invention, for example, are described above with reference to block diagrams and/or operational illustrations of methods, systems, and computer program products according to embodiments of the invention. The functions/acts noted in the blocks may occur out of the order as shown in any flowchart. For example, two blocks shown in succession may in fact be executed substantially concurrently or the blocks may sometimes be executed in the reverse order, depending upon the functionality/acts involved.

[041] While certain embodiments of the invention have been described, other embodiments may exist. Furthermore, although embodiments of the present invention have been described as being associated with data stored in memory and other storage mediums, data can also be stored on or read from other types of computer-readable media, such as secondary storage devices, like hard disks, floppy disks, or a CD-ROM, a carrier wave from the Internet, or other forms of RAM or ROM. Further, the disclosed methods' stages may be modified in any manner, including by reordering stages and/or inserting or deleting stages, without departing from the invention.

[042] All rights including copyrights in the code included herein are vested in and the property of the Applicant. The Applicant retains and reserves all rights in the code included herein, and grants permission to reproduce the material only in connection with reproduction of the granted patent and for no other purpose.

[043] While the specification includes examples, the invention's scope is indicated by the following claims. Furthermore, while the specification has been described in language specific to structural features and/or methodological acts, the claims are not limited to the features or acts described above. Rather, the specific features and acts described above are disclosed as example for embodiments of the invention.

CLAIMS

1. A method (200) for providing visualization platform (150) optimization, the method (200) comprising:

defining (210) a visualization;

creating (220) a plurality of geometry records;

receiving (230) a request to display the visualization;

iterating (240) through the plurality of geometry vectors (135);

translating at least one of the plurality of geometry records into at least one drawing instruction; and

displaying (250) the visualization on a display device (150).

2. The method (200) of Claim 1, wherein iterating (240) through the plurality of geometry vectors (135) comprises skipping at least one of the plurality of geometry vectors (135).

3. The method (200) of Claim 2, wherein skipping at least one of the plurality of geometry vectors (135) comprises determining whether the at least one of the plurality of geometry vectors (135) is associated with at least one criterion.

4. The method (200) of Claim 3, wherein the at least one criterion comprises at least one of the following: a display size, a data type, a visualization type, a total number of data points to be displayed, a total number of the plurality of geometry vectors (135), and a size of an amount of overlap between display objects.

5. The method (200) of Claim 1, further comprising:

determining whether the at least one of the plurality of geometry records comprises a pixel location associated with at least one second vector of the plurality of geometry records; and

in response to determining that the at least one of the plurality of geometry records comprises a pixel location associated with the at least one second vector of the plurality of geometry records, skipping the at least one second vector of the plurality of geometry vectors (135).

6. The method (200) of Claim 1, further comprising:

sending a plurality of the translated geometry records to an application (140); and

rendering the plurality of the translated geometry records for display by the application (140).

7. The method (200) of Claim 6, further comprising:

receiving a request from the application (140) for a batch of translated geometry records;

sending a subset of the plurality of the translated geometry records to the application (140), wherein the subset of the plurality of the translated geometry records comprises a number of translated geometry records associated with the received batch request; and

waiting to send a remaining subset of the plurality of translated geometry records.

8. The method (200) of Claim 1, wherein translating the at least one of the plurality of geometry records into at least one drawing instruction comprises linearizing at least one of the plurality of geometry vectors (135).

9. The method (200) of Claim 1, further comprising:

receiving at least one supported drawing instruction, wherein translating the at least one of the plurality of geometry records into at least one drawing instruction comprises translating the at least one of the plurality of geometry records into the at least one supported drawing instruction.

10. A system (300) for providing visualization platform (150) optimization, the system (300) comprising:

a memory storage (309, 310); and

a processing unit (302) coupled to the memory storage (309, 310), wherein the processing unit (302) is operative to:

create (210) a visualization object associated with a plurality of data values,

composite (220) at least one first value of the plurality of data values into a geometry vector associated with the visualization object, and

skip compositing at least one second value of the plurality of data values.

11. The system (300) of Claim 10, further operative to composite (220) a subset of the plurality of data values into a plurality of geometry vectors (135) associated with the visualization object.

12. The system (300) of Claim 11, further operative to render (250) the visualization object, wherein being operative to render (250) the visualization object comprises being operative to:

translate at least one of the plurality of geometry vectors (135) into at least one drawing instruction.

13. The system (300) of Claim 12, further operative to translate a subset of the plurality of geometry vectors (135) into a plurality of drawing instructions, wherein the subset of the plurality of geometry vectors (135) is selected according to a geometry skipping algorithm (130).

14. The system (300) of Claim 13, wherein the geometry skipping algorithm (130) comprises an algorithm (130) operative to:

translate at least one first vector of the plurality of geometry vectors (135) into at least one drawing instruction,

iterate (240) to at least one second vector of the plurality of geometry vectors (135);

determine whether the at least one second vector of the plurality of geometry vectors (135) comprises a point on the visualization object proximate to a point on the visualization object associated with the at least one first vector; and

in response to determining that the at least one second vector of the plurality of geometry vectors (135) comprises a point on the visualization object proximate to a point on the visualization object associated with the at least one first vector, skip translating the at least one second vector into at least one drawing instruction.

15. A computer-readable medium which stores a set of instructions which when executed performs a method (200) for optimizing a visualization platform (150), the method (200) executed by the set of instructions comprising:

receiving (210) a request from a user application (140) to create a visualization object, wherein the request comprises a plurality of data values and a visualization type;

compositing (220) a plurality of geometry vectors (135) associated with the visualization object, wherein compositing the plurality of geometry vectors (135) comprises:

compositing at least one geometry vector associated with at least one of the plurality of data values, and

skipping at least one of the plurality of data values according to a configurable granularity;

receiving (230) a render request from the user application (140);

translating each of a subset of the plurality of geometry vectors (135) into at least one drawing instruction, wherein the subset of the plurality of geometry vectors (135) is selected according to a geometry skipping algorithm (130);

sending the at least one drawing instruction associated with each of the subset of the plurality of geometry vectors (135) to the user application (140);

rendering (250) the visualization object, wherein rendering the visualization object comprises rendering the at least one drawing instruction associated with each of the subset of the plurality of geometry vectors (135) to a display device (150);

receiving a second render request from the user application (140) in response to a detected change affecting the rendered visualization object;

translating each of a second subset of the plurality of geometry vectors (135) into at least one drawing instruction;

sending the at least one drawing instruction associated with each of the second subset of the plurality of geometry vectors (135) to the user application (140); and

re-rendering the visualization object.