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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
30 May 2014 (30.05.2014)

(10) International Publication Number
WO 2014/081420 A1

(51) International Patent Classification:
G06F 9/44 (2006.01)

(21) International Application Number:
PCT/US2012/066159

(22) International Filing Date:
20 November 2012 (20.11.2012)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
13/681,424 20 November 2012 (20.11.2012) US

(72) Inventor; and
(71) Applicant : EKPAR, Frank Edughom [NG/US]; Massachusetts (US).

(72) Inventor: EKPAR, Frank Edughom; Matsunaga 1-17-26-A107, Ikki-machi, Aizuwakamatsu City, Fukushima 965-0001 (JP).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, 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, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

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

Declarations under Rule 4.17:

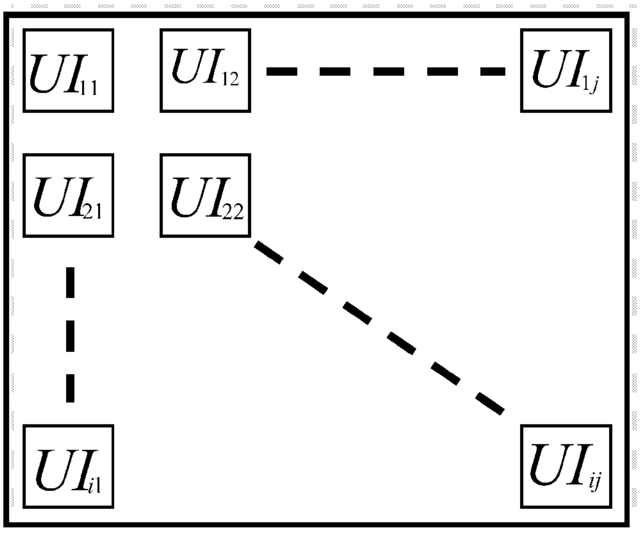
— *of inventorship (Rule 4.17(iv))*

Published:

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

(54) Title: ARBITRARY DIMENSIONAL USER INTERFACES

FIG. 1



(57) **Abstract**: The present invention discloses an arbitrary dimensional graphical user interface permitting the display or presentation of one or more instances of the user interface on one or more arbitrarily-sized and arbitrarily-shaped surfaces with each instance comprising one or more arbitrary dimensional background elements each of which is divided into one or more arbitrarily-sized and arbitrarily-shaped arbitrary dimensional partitions, wherein each partition may contain one or more user interface elements and is associated with one or more sets of rules that define rendering, positioning, element placement and other relevant attributes and behaviors, wherein said rules can be specified in such a way as to enable said arbitrary dimensional background to assume any desired arbitrary shape and to facilitate expansion to any desired arbitrary size without distortion or loss in quality. In this context, an instance of a graphical user interface is any display or presentation of any representation of any aspect of the user interface. Furthermore, the present invention discloses automatic, semi-automatic or manual processing systems for arbitrary dimensional graphical user interfaces based on the principles of the present invention.

ARBITRARY DIMENSIONAL USER INTERFACES

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates generally to the field of graphical user interfaces. In particular, the invention relates to a graphical user interface system permitting the creation of arbitrary dimensional $(\alpha - \dim \mathit{ensional})$ graphical user interfaces that can have any shape and that can dynamically be expanded to any size without distortion or loss in quality and that comprise systems for automatic, semi-automatic or manual processing of the user interface.

Description of the Prior Art

Contemporary graphical user interfaces are limited in that when they allow arbitrary shapes, they are generally not expandable and when they are expandable, they generally do not permit the use of arbitrary shapes. Furthermore, these graphical user interfaces are generally limited to flat 2-dimensional or at best simulated 3-dimensional structures. Popular graphical user interface systems from software developers such as Microsoft and Apple and consequently the computer systems and computer-implemented methods offered by these popular and other less popular vendors and their products and services suffer from these limitations.

Most significantly, in spite of the well-known fact that numerous critically important issues in science and engineering, medicine, law enforcement, economics and finance, and many other fields of human endeavor require the management of arbitrary dimensional data, the user interfaces permitted by the prior art lack the ability to manage arbitrary dimensional data directly or efficiently. In fact, contemporary graphical user interfaces typically lack even the very concept of arbitrary dimensionality – failing to provide any practical way or means of managing arbitrary dimensional data.

Liu et al. (US2005/0172239 A1) teach a graphical user interface that could be stretched or resized. The characteristics or nature of designated areas (such as “border” or “resize” regions) could be used to provide hints (such as a change in the shape of the cursor) to the user that resizing or stretching is possible or occurring at a particular position. However, Liu et al. fail to teach the use of adaptive or selective rendering of the areas of the graphical user interface to achieve the said resizing or stretching. In fact, Liu et al. fail to teach any specific way to achieve the resizing or stretching at all.

Furthermore, Liu et al. teach "creating one or more first regions..." and "creating one or more second regions" for the user interface. Thus, Liu et al. teach the use of at least two regions - at least one first region and at least one second region.

N.M. et al. (US 2003/0041099 which has matured into US 7,165,225) teach a graphical user interface (GUI) based on GUI objects but fail to teach any specific way of resizing the user interface.

Hamlet et al. teach the use of "optimized vector image data" to enable the display of a graphical user interface in any shape and at any size with minimal or no loss of original image quality, Hamlet et al. fail to provide the option of utilizing the nature (possibly related to the appearance or texture) of the original image. In fact, Hamlet et al. teach away from the use of the appearance of the original image. However, according to the principles of the present invention, the adaptive or selective application of appropriate rules to selected regions of the original image based on the nature (possibly related to the texture or appearance of same) can achieve “infinite resolution” and permit the user interface to assume any desired shape and size without easily noticeable distortion or loss of quality. Additionally, it should be noted that according to the teachings of Hamlet et al., the “optimized vector image data” is generally created separately and could also be stored separately from the graphical user interface to which it is applied and that changes in certain attributes (such as size) of the interface may necessitate re-computation of vector data. In contrast, the present invention provides the option of directly using the original image and adaptively applying appropriate rules to selected regions of the image with suitable characteristics to facilitate user interfaces that can assume any size and shape. Thus, the options provided by the present invention obviate the need to generate, access or otherwise compute or re-compute vector data, leading to savings in resources and allowing for faster and more responsive and richer user interfaces than permitted by the prior art.

Callaghan et al. (US2005/0108364 A1) teach a graphical user interface that employs scalable vector graphics (SVG) for rendering – including scaling, resizing or stretching.

Similarly, Kaasila et al. (7,222,306 B2) teach a graphical user interface that utilizes a plurality of scale factors for scaling, resizing or stretching. It should be noted that although the scale factors utilized by Kaasila et al. can be selected from a list of available scale factors or generated in response to user interaction with the user interface, none of the prior art (N.M: US2003/0041099, Hamlet et al.: US 6,606,103, Liu et al.: US2005/0172239 A1, Callaghan et al.:US2005/0108364 A1, Kaasila et al.: 7,222,306 B2) teaches a user interface wherein resizing is based on the nature of selected regions.

Similarly, none of the prior art (N.M: US2003/0041099, Hamlet et al.: US 6,606,103, Liu et al.: US2005/0172239 A1, Callaghan et al.: US2005/0108364 A1, Kaasila et al.: 7,222,306 B2) teaches a user interface wherein one or more design characteristics of one or more partitions are guided by one or more intended presentation characteristics of the affected partition.

Although Guo et al. (US 2006/0104511 A1) teach a user interface in which intended display or presentation characteristics of the user interface could be used to edit the user interface – thus allowing intended display or presentation characteristics to guide or inform design characteristics, the guidance being utilized via the editing of the user interface to better conform with the intended display or presentation characteristics, Guo et al. fail to teach the adaptive or selective application of appropriate rules to selected regions of the original image based on the nature (possibly related to the texture or appearance of same) to achieve "infinite resolution" and permit the user interface to assume any desired shape and size without easily noticeable distortion or loss of quality either in the representations presented or displayed or in the original background elements themselves.

One of ordinary skill in the art would readily appreciate that characteristics of user interface elements such as texture, shape, size, and so on, can be determined or chosen or selected or specified or modified or edited at the time or during the process whereby the user interface elements are designed or created or modified or edited. Thus, these characteristics are obviously design characteristics and consequently the term **"design characteristics"** does not require additional explanation in the specification. Any characteristic of any user interface element such as texture, shape, size, and so on, that can be determined or chosen or selected or specified or modified at the time or during the process whereby the user interface element is designed or created or modified or edited is obviously a design characteristic.

Similarly, one of ordinary skill in the art would readily appreciate that characteristics of user interface elements such as texture, shape, size, and so on, can be determined or chosen or selected or specified or modified or edited at the time or during the process whereby the user interface elements are presented or displayed in any suitable representation based on the intent of the user or designer or creator. Thus, these characteristics are obviously intended presentation characteristics and consequently the term **“intended presentation characteristics”** does not require additional explanation in the specification. Any characteristic of any user interface element such as texture, shape, size, and so on, that can be determined or chosen or selected or specified or modified at the time or during the process whereby the user interface element is presented or displayed in any suitable representation based on the intent of the user or designer or creator is obviously an intended presentation characteristic.

The set of rules that each partition (or generally user interface element) is individually associated with and that defines attributes and behaviors of the partition can be selected or specified or formulated in a manner that guides or informs design decisions pertaining to the partition (or generally user interface element) at the time or during the process whereby the partition (or generally user interface element) is designed or created or modified or edited. For example, a partition that is intended to be displayed or presented in representations that permit the ability to resize the partition arbitrarily could be designed as simple partitions with a uniform texture for the elements comprising the partition. Similarly, a partition that is intended to be displayed or presented in representations that do not permit the ability to resize the partition arbitrarily but that limit the rendering or display or presentation to the original size of the partition could be designed as complex partitions with an arbitrarily complex texture for the elements comprising the partition since there is little chance of distortion of the partition as a result of stretching during resizing.

As clearly explained in the specification, the converse is also true. That is, the rules can be specified based on the characteristics of the relevant partition or user interface element as currently designed or created.

The present inventor discloses a versatile graphical user interface comprising one or more N-dimensional background elements each of which is divided into one or more arbitrarily-shaped N-dimensional partitions, wherein each partition may contain one or more user interface elements and is associated with one or more sets of rules that can be based on the nature of the partitions and that define rendering, positioning, element placement and other relevant attributes and behaviors, wherein said rules can be specified in such a way as to enable said N-dimensional background to assume any desired arbitrary shape and to facilitate expansion to any desired arbitrary size without distortion or loss in quality in US 2005/0225572 A1. According to US 2005/0225572 A1, N can be 1, 2, 3, 4 - for 3 spatial dimensions and 1 temporal dimension for instance, 5, or any number of dimensions. Although the invention disclosed in US 2005/0225572 A1 remedies many of the limitations of the prior art and leads to the creation of much more versatile, more dynamic and richer user interfaces than are possible with the prior art, N, the number of dimensions is not truly arbitrary but is associated with specific embodiments of the invention. For example, the invention disclosed in US 2005/0225572 A1 enables embodiments where N can be 1, 2, 3 or 4 but does not define embodiments in which N is greater than 4 or 5 or higher.

As noted previously, it is a well-known fact that numerous critically important issues in science and engineering, medicine, law enforcement, economics and finance, and many other fields of human endeavor require the management of arbitrary dimensional data but the user interfaces permitted by the prior art lack the ability to manage arbitrary dimensional data directly or efficiently and as a matter of fact contemporary graphical user interfaces typically lack even the very concept of arbitrary dimensionality and thus fail to provide any practical way or means of managing arbitrary dimensional data.

SUMMARY OF THE INVENTION

It is an object of the present invention to overcome the limitations of the prior art set forth above by providing an arbitrary dimensional $(\alpha - \dim ensional)$ graphical user interface permitting the display or presentation of one or more instances of the user interface on one or more arbitrarily-sized and arbitrarily-shaped surfaces with each instance comprising one or more arbitrary dimensional background elements each of which is divided into one or more arbitrarily-sized and arbitrarily-shaped arbitrary dimensional partitions, wherein each partition may contain one or more user interface elements and is associated with one or more sets of rules that define rendering, positioning, element placement and other relevant attributes and behaviors, wherein said rules can be specified in such a way as to enable said arbitrary dimensional background to assume any desired arbitrary shape and to facilitate expansion to any desired arbitrary size without distortion or loss in quality. Furthermore, the present invention discloses automatic, semi-automatic or manual processing systems for arbitrary dimensional graphical user interfaces based on the principles of the present invention.

Graphical user interfaces enabled by the present invention can have an arbitrary number of dimensions and can thus be used to resolve numerous critically important issues in science and engineering, medicine, law enforcement, economics and finance, and many other fields of human endeavor require the management of arbitrary dimensional data.

The unlimited (within normal resource constraints) number of dimensions permitted by the present invention allow new and useful relationships in data to be discovered that could not be discovered or that could only be discovered by exerting an inordinate amount of effort or expending an inordinate amount of resources using the prior art.

Entirely new classes of applications that were hitherto impossible to realize or that could only be realized through the expenditure of inordinate amounts of effort and resources could easily and directly be realized on the basis of the principles of the present invention.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates a representative concept for the preferred embodiment of the present invention.

FIG. 2 shows how instances of the user interfaces enabled by the present invention could be disposed on one or more surfaces.

FIG. 3 demonstrates tri-linear interpolation for a user interface instance bounded by eight neighboring user interface instances in a 3-dimensional spatial configuration.

FIG. 4 shows the partitioning of a background image according to the prior art.

FIG. 5 depicts a flowchart for the processing of the user interface for a preferred embodiment of the present invention.

FIG. 6 illustrates the tier-1 image representation used by the preferred embodiment of the present invention for the management of very large data sets.

FIG. 7 shows the partitioning or segmentation of the original image frame to form tier-2 of the image representation used by the preferred embodiment of the present invention for the management of very large data sets.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Generally, a computer system such as a personal computer system, workstation, server, tablet computer system, handheld or mobile computer system and any other suitable system could be used to embody the present invention. Other suitable devices and systems providing means for or allowing the steps of the present invention to be carried out could be used. When a computer system is used, user interaction with the user interface could be via a mouse or any other suitable means. Well-known alternative means of interacting with user interfaces include gesture recognition systems, touch-based systems such as touch-screens and associated systems, brain-computer interfaces, speech recognition systems, and so on. These could all be employed in interacting with suitable representations of any aspect of the user interfaces enabled by the principles of the present invention. Data for the interface could be stored or generally managed in computer memory and software running on the computer system could be used to allow editing and presentation of the user interface. Suitably configured network systems such as the Internet could also be used to store or generally manage data associated with the user interfaces. The user interface could be presented or rendered on a computer monitor or screen or any other suitable display or presentation system. Suitable computer network systems could be used to implement and/or present aspects of the user interface.

In the context of the present invention, an instance of a graphical user interface is any display or presentation of any representation of any aspect of the user interface. Consequently, an instance of a graphical user interface could include, without limitation, any of the following elements or any combination of the following elements: representations of background elements, representations of partitions extracted from or defined in association with background elements, view or display areas for data managed using the user interface, user interface elements such as background elements, partitions, buttons, scrollbars, sliders, windows comprising any number or combination of user interface elements, and so on.

A background element could be a single background image or a collection of images considered as a background image. Elements within the background image could be pixels or groups of pixels or fractions of pixels or groups of fractions of pixels (for sub-pixel precision) in the case of a digital image.

Partitions could be any selected, demarcated, defined or labeled aspect or region of a background element. Thus partitions could comprise labeled pixels or labeled groups of pixels contained either in a single image or distributed over a collection of images. Parts of a background image could actually be cut out into a separate image to represent a partition – in which case the partition could be considered a literal partition. It should be obvious to one of ordinary skill in the art that partitions could be defined conceptually over any number of background elements including, but not limited to, pixels or groups of pixels or individual images without actually cutting out the elements into separate parts but by simply maintaining the data representing the partitions within the user interface. Such partitions could be considered conceptual partitions.

Any representation of any data displayed or rendered or presented using the user interface could be considered a part of the user interface and thus a component of an instance of the user interface. Data could be represented as digital images comprising pixels or groups of pixels. Other suitable representations of data could be utilized as required by any given application of the present invention. Interaction with representations of data could be facilitated via interaction with the user interface as is well-known to one of ordinary skill in the art. Such interaction could involve clicking buttons, manipulating sliders, resizing or manipulating partitions, background elements or any suitable representations thereof.

Referring now to FIG. 1, an illustration of a representative concept for the preferred embodiment of the present invention, instances of the user interface indicated generally as UI_{11}, UI_{12}, UI_{21}, UI_{22}, UI_{1j}, $UI_{i1}, \ldots, UI_{ij}$ are disposed on an arbitrarily-shaped and arbitrarily-sized surface. The surface is depicted by the bounding box for the user interface instances in FIG. 1.

The individual user interface instances are shown in a manner reminiscent of entries in a 2-dimensional (2D) matrix. According to the principles of the present invention, the surface represented by the 2D matrix in FIG. 1 can be assigned an arbitrary size and may assume an arbitrary shape and is not limited to a 2-dimensional or 2D plane. Suffixes **i** and **j** (as in UI_{ij}) refer to row and column numbers within the 2D matrix. Hence, UI_{11} is the instance in row 1, column 1; UI_{12} is the instance in row 1, column 2; UI_{21} is the instance in row 2, column 1; UI_{22} is the instance in row 2, column 2; UI_{1j} is the instance in row 1, column j, where j is an arbitrary number, UI_{i1} is the instance in row i, column 1, where i is an arbitrary number and UI_{ij} is the instance in row i, column j, where i and j are arbitrary numbers.

Entries in columns refer to specific instances of the user interface. Please note that the underlying user interface elements need not be distinct for each individual instance. For example, one or more instances may share the same user interface elements but may be used to present or display different views or perspectives or representations of data presented or displayed using the user interface. Thus, one or more entries in columns could represent different representations of data while sharing some or all underlying user interface elements.

Rows within the 2D matrix represent individual or separate dimensions. Consequently, row 1 could be used to represent a new or additional dimension for the user interface instances contained in the columns within row 1, namely, UI_{11}, UI_{12}, UI_{13},..., UI_{1j-1}, UI_{1j}.

Consequently, irrespective of the actual number of dimensions contained within a specific user interface instance such as UI_{11} within row 1, the inclusion of UI_{11} as a column (column 1 in this case) within row 1 increases the number of dimensions for the specific user interface instance and its derivative representations by 1. Hence if UI_{11} was originally a conventional 2D user interface, then the inclusion of UI_{11} as a column within row 1 increases the number of dimensions for UI_{11} and its derivative representations by 1 and thus promotes UI_{11} and its derivatives to a 3D user interface. A derivative instance in this context refers to any new instance of the user interface (such as UI_{12}) that shares some or all underlying user interface elements and that could be used to present or display a different view or perspective or representation of data presented or displayed using the user interface. Note that the concept of derivation is not limited to the sharing of one or more user interface elements but could be defined in terms of associations between instances. That is, a new instance could be considered a derivative of an existing instance if the two instances are associated with the same dimension (for example by being disposed within the same row of the representative matrix) within the user interface irrespective of whether the instances actually share any common elements or not. So entries within row 1 could be used to represent or display or present derivatives of UI_{11} and thus promote UI_{11} and derivatives of UI_{11} to a 3D user interface in the case where UI_{11} (and its derivatives) were originally conventional 2D user interface instances.

Adding a new dimension to the user interface simply involves adding a new row to the 2D matrix representing the user interface and representing, displaying, presenting or rendering derivative instances within the new row. Similarly, removing or disabling an unused or unwanted dimension from the user interface involves simply removing or disabling the affected row within the representative 2D matrix and disabling the representation, display, presentation or rendering of instances within the affected row.

Consequently, the 2D matrix representation of the user interface depicted in FIG. 1 and described in the foregoing is sufficient to permit the user interface to manage arbitrary dimensional data without limitation. The user simply adds a new row to the 2D matrix in order to add a new or additional dimension to the user interface.

Note that according to the principles of the present invention, the individual user interface instances are themselves arbitrary dimensional. That is, each instance could be managed using a representative 2D matrix as described for the entire user interface.

One of ordinary skill in the art will readily appreciate that the actual representations of the instances – which could be in the form of digital images or vector graphics on computer systems or any other suitable representation – could be rendered or displayed or presented in a wide variety of ways or using a wide variety of means. For example, each instance could be displayed as a separate window within the desktop of a typical window-based operating system such as those available from Microsoft (Microsoft Windows, Windows Mobile, and so on), Apple (Apple OS, iPhone OS, iPad OS, and so on), Google (Android, and so on) or any other window-based operating system. Given a sufficiently large or high resolution display device or display surface, it is possible to display each instance as a region within a single window within a typical window-based operating system desktop or similar surface. The exact configuration or arrangement of instances within such a window or the exact configuration or arrangement of windows (in the case where separate windows are assigned to separate instances) could be determined in accordance with user preferences or other relevant factors such as resource constraints. Suitable systems for managing the representations of the instances could also be adopted. For example, window swapping techniques allowing one or more instances to be viewed at a time or in a specific situation or providing means of selecting which specific instance to view at a specific time or in a specific situation could be adopted in resource-constrained environments. Alternatively, each instance could be displayed on a separate monitor or display device where the monitors or display devices are spread over any chosen geographical territory in any chosen spatial configuration or arrangement. One simple spatial arrangement could involve disposing the monitors or display devices in a manner reminiscent of a 2D matrix to closely match the 2D matrix representation of the user interface.

It should be apparent to one of ordinary skill in the art that the arbitrarily-sized and arbitrarily-shaped surfaces on which the user interface instances could be disposed could be 1-dimensional (constraining the instances to a linear configuration), 2-dimensional (constraining the instances to a planar or 2D configuration), 3-dimensional (constraining the instances to the familiar physical 3D spatial environment or configuration), 4-dimensional (for example comprising a spatial 3D configuration and a linear or 1D time dimension as in time-varying spatial 3D configurations or arrangements) or arbitrary dimensional by adopting the principles of the present invention and possibly utilizing the 2D matrix paradigm introduced by the present invention. Furthermore, it should be understood that the surfaces need not be literal or physical surfaces but could represent computer memory, rendering surfaces – such as device contexts in Microsoft Windows Software Development Kit parlance – or any other suitable representations of surfaces on which instances could be disposed.

Any of the well-known viewport and window management techniques including, but not limited to, scrolling, panning, and so on, could be employed where appropriate to facilitate the display or presentation of the instances.

The conceptualization of the user interface of the present invention as a 2D matrix of arbitrary dimensional instances permitting new or additional dimensions to be added simply by adding new rows to the representative 2D matrix and permitting unused or unwanted dimensions to be removed or disabled simply by removing or disabling rows within the representative 2D matrix makes user interfaces enabled by the present invention amenable to straightforward mathematical characterization and analysis.

Using the 2D matrix paradigm, useful characteristics associated with the user interface such as performance metrics could easily be generated and managed using well known and widely used mathematical tools. For example, each instance could be configured to collect and store - or transmit for further analysis – user data associated with any selected aspect of the user experience and this data could be represented and analyzed using the 2D matrix notation – the representation of the user interface itself as a 2D matrix of instances making the collection, analysis and utilization of the data easier or more efficient.

Performance metrics analysis could be facilitated by carrying out automatic, semi-automatic or manual collection and analysis of relevant user experience data. Automatic data collection could be implemented by any means or method that allows the tracking of relevant user actions such as mouse clicks, button activation, hits and misses on buttons, sliders or any other tracked user interface elements. Such tracking of using actions could readily be programmed into the user interface by one of ordinary skill in the art using readily available programming languages such as C, C++, JAVA, Python, HTML, HTML5, VRML, and so on in combination with suitable programing tools such as software development kits for any chosen computer system or application environment. Other means of tracking the user experience such as the use of suitably configured brain-computer interfaces or similar bio-technology systems or any other suitable systems to determine the level of excitement or frustration caused by the use of specific instances of the user interface or the amount of effort required by specific groups of users to master the user interface to a specific level of proficiency could also be utilized in the automatic collection of user experience data.

Semi-automatic data collection could involve the use of any of the automatic data collection methods described earlier (or any other suitable automatic data collection method) augmented by manual inspection and/or correction of errors in the automatically collected data.

Manual data collection could involve the explicit or implicit (where appropriate) use of user surveys, polls or queries to document or track the user experience.

Once user data associated with any specific instances has been collected, known matrix analysis methods or any other suitable mathematical tools could be used to compute and characterize performance metrics with a view to improving the user experience and building better – more responsive and more efficient – user interfaces.

Any other aspect of the user interface – such as design or presentation issues - could also be analyzed using matrix notation aided by the conceptualization of the user interface of the present invention as a 2D matrix of arbitrary dimensional instances.

While it is very clear from the foregoing that the conceptualization of the user interface of the present invention as a 2D matrix of arbitrary dimensional instances is sufficient to permit any arbitrary dimensional application of the user interface, some users may prefer the more familiar 3D spatial configuration. Accordingly, FIG. 2 illustrates how instances of the user interfaces enabled by the present invention could be disposed on one or more arbitrarily-sized and arbitrarily-shaped surfaces along the lines of the familiar 3D spatial configuration.

In FIG. 2, three separate surfaces are shown as an example. Any number of surfaces could be used in practice. Each surface is arbitrarily-sized and arbitrarily-shaped and based on the principles of the present invention, is arbitrary dimensional as already explained for the 2D matrix notation. The instances disposed on the surfaces are labeled $\{UI_{11}\}_1$, $\{UI_{1j}\}_1$, $\{UI_{i1}\}_1,\ldots,\{UI_{ij}\}_1$ for the first surface; $\{UI_{11}\}_2,\{UI_{1j}\}_2,\{UI_{i1}\}_2,\ldots,$ $\{UI_{ij}\}_2$ for the second surface (note that $\{UI_{i1}\}_2$ and $\{UI_{ij}\}_2$ are not visible in FIG. 2) ; and $\{UI_{11}\}_l$, $\{UI_{1j}\}_l$, $\{UI_{i1}\}_l,\ldots,\{UI_{ij}\}_l$ for the last surface numbered l in FIG. 2 where **l** is the surface number and can be chosen arbitrarily to satisfy the demands of a given application. In this conceptualization the surfaces labeled 1, 2,…, **l** (where l is any number) constitute a new or additional dimension for the user interface.

By considering the collection of surfaces numbered 1, 2,…, **l** as a new or additional dimension (separate from the familiar rows – numbered 1, 2, …, i and columns – numbered 1, 2, …, j of the 2D representation already described which together constitute an unlimited number of dimensions) for the representative matrix, the matrix could be extended to a 3-dimensional or 3D form. Just as explained for the 2D matrix formulation, techniques for the mathematical characterization and analysis of 3D matrices could be applied to this 3D matrix formulation for the representative matrix.

One of ordinary skill in the art would understand that any specific dimension could be extracted from a specific row of the 2D matrix form and interpreted as an additional or third matrix dimension (which must be distinguished from an actual user interface dimension) to form a 3D matrix representation. Furthermore, this process could be repeated to create arbitrary dimensional matrix representations. Conversely, a specific matrix dimension in any matrix formulation with three or higher dimensions could be extracted from the higher dimensional matrix and interpreted as a new or additional row for the 2D matrix form and in so doing reduce the number of dimensions for that specific matrix representation. Generally, arbitrary dimensional matrices could be used to represent the user interface in the manner described and these matrices could be transformed from one dimensionality to the other in a process involving dimension expansion or reduction to facilitate mathematical characterization and analysis in any preferred formulation. As already explained, the 2D matrix form is sufficient for any number of actual dimensions for the user interface. The use of higher dimensional matrices is for convenience in mathematical characterization and analysis only and is not required for adequate representation of the arbitrary dimensional user interfaces of the present invention.

A useful metaphor for the intuitive understanding of the 3D matrix representation as illustrated in FIG. 2 is the book and page metaphor. In this interpretation, each surface could be considered a page in a book representing the user interface. Each individual user interface instance is disposed on or contained in one of the arbitrarily-sized and arbitrarily-shaped pages of the book. The pages of the book could be flipped to reveal additional pages or more generally to navigate the book by moving from one page to the other. This page flipping or page navigation could be implemented via the use of appropriate buttons or any other user interface elements on any of the instances that could be activated to initiate a page flip and reveal additional instances disposed on pages separate from the current page. Hence, a user could step through the user interface – gaining access to instances disposed on successive pages – by utilizing page flipping commands or any other suitably configured elements on any of the instances within the current page. Display or presentation of instances could be carried out as previously described for the 2D matrix representation. Appropriate transition effects could be added to enhance user perception of the page flipping process.

SYNTHESIZING OR CREATING NEW USER INTERFACE INSTANCES

New instances of the user interface could be synthesized or created by utilizing existing instances of the user interface. Interpolation techniques requiring neighborhood relationships or associations between the existing instance or instances and the new or synthesized instances could be employed. Such interpolation techniques include, but are not limited to, bilinear interpolation (especially for 2D matrix representations), cubic interpolation, spline interpolation, tri-linear interpolation (especially for 3D matrix formulations) and so on.

The synthesis of new instances could be viewed as a means of improving the resolution of the user interface by permitting the creation or synthesis or prediction of new instances that may improve the value of the user interface.

One of ordinary skill in the art would appreciate that interpolation is by no means the only way to improve resolution or to synthesize, create or predict new instances. New instances could be synthesized without reference to any existing instances. Physical constraints or other relevant factors within a given application of the present invention could allow new instances to be predicted, created or synthesized without reference to existing instances.

Furthermore, synthesis of new instances on the basis of one or more existing instances could be accomplished using any other suitable means apart from the interpolation techniques mentioned in the foregoing.

When utilizing a 3D matrix formulation for the representation of the user interface, tri-linear interpolation could be used to synthesize or compute new instances of the user interface by applying the data from neighboring instances.

FIG. 3 depicts tri-linear interpolation for a user interface instance bounded by eight neighboring user interface instances in a 3-dimensional spatial configuration and illustrates how tri-linear interpolation could be implemented. For example, to compute data for the point labeled T(x, y, z) located at 3D coordinates (x, y, z) from an arbitrarily chosen origin and for which data is not directly available, data for the neighboring points labeled V000, V100, V101, V001, V010, V110, V111 and V011 could be utilized as follows and tri-linear interpolation applied.

$$T(x, y, z) = (V000*(1-x) + V100*x) * (1-y) + ((V010*(1-x)+V110*x)*y)*(1-z)+V001*(1-x)+V101*x)*(1-y)+((V011*(1-x)+V111*x)*y)*z$$

Note that L0, L1, L3, L2 and B0, B1 are intersection points between adjacent faces of the cube formed by the neighboring points (namely V000, V100, V101, V001, V010, V110, V111 and V011) for which data is available and the target point – T(x, y, z) - for which no data is available.

It should be obvious to one of ordinary skill in the art that the correct interpretation of the entities represented by the data points labeled V000, V100, V101, V001, V010, V110, V111 and V011 for which data is available depends on the specific application of the present invention. For instance, data within an instance could be represented as an image with elements comprising pixels or fractions of pixels (for sub-pixel precision) with the typical red, blue, green, alpha (RGBA) quad values for the associated red, green and blue color channels and alpha transparency channel and associated 3D coordinates x, y, z, and an optional time component that could be used in the case of video or time-varying image data. In this case the foregoing equation for tri-linear interpolation could be interpreted as a pixel-wise operator permitting the synthesis of new pixels for new instances by utilizing the values of existing pixels for existing instances.

BACKGROUND PARTITIONING

Referring now to FIG. 4, an illustration of a preferred embodiment of the present invention, the arbitrarily-sized and arbitrarily-shaped background is indicated generally as B. In FIG. 4, P_1, P_2, P_3, P_4, P_5, P_6, P_7,..., P_k are arbitrary dimensional partitions and k can be any number. For simplicity, the background and partitions in FIG. 4 are depicted in 2-dimensional or 2D form. Furthermore, the partitions are contiguous. In practice, however, the background and partitions are arbitrary dimensional according to the principles of the present invention and the partitions need not be contiguous. Furthermore, the partitions need not be literal – in which case a background comprising a 2-dimensional or 2D image would need to be broken up into a plurality of images to support a plurality of partitions – but could be logical or conceptual only – in which case said background image could remain monolithic and the partitions represented in any suitable data format that corresponds to the conceptual form. Each partition may contain any number of user interface elements. Clearly, the partition itself is a user interface element. Thus, a partition could be deemed to contain no user interface element if the partition does not contain any other additional user interface element apart from the partition itself. According to the principles of the present invention, each partition is arbitrary dimensional and has an arbitrary shape and an arbitrary size and is associated with a set of rules that define rendering, positioning, element placement and other relevant behaviors and attributes. (Generally, attributes and behaviors or characteristics or aspects of the user interface are chosen on the basis of usefulness or relevance in a given embodiment.) These rules can be specified in such a way that the arbitrary dimensional background-based graphical user interface can assume any arbitrary desired shape and can be expanded to any arbitrary desired size without distortion or loss in quality. For instance, if the background comprises a single, arbitrarily-shaped digital image and the user interface built from said background is to be rendered on a computer screen, then said background can be divided into a number of partitions based on the nature of the background and the rendering of each partition can in turn be carried out on the basis of the nature of the partition. A partition defined on a uniformly textured region of the background can be stretched without noticeable distortion or loss in quality. In contrast, a partition defined on a non-uniform region of the background may be rendered in its original size and shape to prevent distortion and loss in quality. By creating a number of partitions based on the nature of the background and selectively assigning appropriate sets of rules for rendering, positioning, component or element placement and other behaviors and attributes of each partition, the entire background can be made to assume an arbitrary shape and an arbitrary size without distortion or loss in quality. Consequently, user interfaces based on the principles of the present invention are more versatile, more dynamic and allow a much richer user experience than is possible with the prior art. Most significantly, user interfaces based on the principles of the present invention are arbitrary dimensional and can be used to manage arbitrary dimensional data without the limitations inherent in the prior art.

FIG. 5 shows a flowchart for automatic processing of the user interface. The user interface could comprise a background image that is to be partitioned. Partitioning could be based on the nature of the image. For example, the texture of the image could be used.

In the step indicated generally as 110 (START) in FIG. 5, the image could be prepared for processing. Such preparation could involve storing the image in memory and providing means of accessing the data representing the image. It could also comprise – in the case of a network-based system – the streaming or transmission of the data representing the image for further manipulation. If required, preparation could also involve pre-processing steps such as filtering and de-noising of the image or the application of any combination of any required pre-processing steps as is well-known in the art.

In the step labeled 120 (IDENTIFY PARTITIONS), an automatic process for the identification of distinct partitions or regions within the image could be carried out. The identification could be based on the texture of the image. Image segmentation techniques (including those popular in the literature) could be used in this step. Any other suitable process for automatically identifying partitions could also be used. The process could be completely automated – in which case the results of an automatic image processing step such as image segmentation are used as the basis for partitioning the image. Furthermore, a semi-automatic process could be used – in which case the automatic partition identification process could be augmented (via user or designer inspection) to manually correct any misidentifications or to more closely conform to user taste. More generally, the step of identifying partitions in the user interface comprises assigning a label to each element in the image such that elements with the same label share common characteristics. In the case of a digital image, each such element would be a pixel. For simplicity, the texture of the image could be chosen as the characteristic on which the labeling of elements is based. It should be noted that any other suitable characteristic (including, but not limited to, shape) could be chosen as the basis of the partitioning. Typical labels could be SIMPLE (for elements with a simple or uniform texture), COMPLEX (for elements with a relatively more complex texture), HORZONTAL (for elements with a texture that appears horizontal) and VERTICAL (for elements with a texture that might appear vertical). Other suitable labels could be used. For simplicity, the boundaries of identified partitions could be expanded to abut neighboring partitions when necessary in order to ensure that all identified partitions taken together cover the entire background without gaps.

According to the principles of the present invention, the partitioning process is not limited to the automatic and/or semi-automatic processes described in the foregoing. Partitioning could be carried out manually on the basis of a visual inspection of the user interface. Manual identification of partitions could be accomplished on a computer system via suitable instructions (possibly embodied in application software) that permit the designer or user to identify and/or label partitions. This could be accomplished by clicking and dragging a computer mouse over the background image to demarcate or identify and/or label partitions. The labels (for example SIMPLE, COMPLEX, HORIZONTAL, VERTICAL, and so on) mentioned for automatic and/or semi-automatic partition identification could also be applied to manual partition identification.

In step 130 (ASSIGN RULES TO PARTITIONS), each partition identified in step 120 could be associated with a set of rules defining the characteristics of the partition. For example, based on the texture of the identified partition, a specific partition could be designated for vertical tiling during rendering or presentation. By way of example, consider the situation in which the rendering or presentation of the user interface is the characteristic that is to be defined. Any partition labeled SIMPLE could be assigned a rendering or presentation rule that effectively causes the partition to be stretched to fit its destination. In the case of a two-dimensional digital image, this could be accomplished via simple two-dimensional (horizontal and vertical) pixel replication as is well known in the field. In contrast, a partition labeled COMPLEX could be assigned a rendering or presentation rule that effectively causes the partition to be rendered at its actual size – in which case any destination region allocated to the partition could be constrained to the same size and shape as the original partition. A partition labeled HORIZONTAL could be assigned a rendering or presentation rule that effectively causes the partition to be tiled horizontally while a partition labeled VERTICAL could be assigned a rendering or presentation rule that effectively causes be partition to be tiled vertically during rendering or presentation on a destination surface or device.

One of ordinary skill in the art would appreciate that it is possible to synthesize a mapping or table associating rules or sets of rules for rendering (or any other chosen characteristic) with a label or sets of labels identifying partitions within the interface.

Finally, assigned rules can be applied in step 140 to the associated partitions in the storage, presentation and/or more generally further manipulation of the user interface.

Additionally, identification of partitions and/or assignment of rules to identified partitions could also be based on a selected user or designer profile. Such a profile could be built up automatically on the basis of prior user interaction with the user interface, user preferences or other relevant data gathered about the user or designer. For example, a specific user or designer could prefer that SIMPLE partitions be tiled vertically while another could prefer that such partitions be simply stretched via pixel replication or an equivalent process. Via appropriate program code or computer software, the user or designer could be permitted to edit such a profile or build a new profile from scratch. Thus, user or designer profiles could be built explicitly on the basis of user or designer input. Another option is to use a semi-automatic approach in which a user or designer profile could first be built automatically (possibly on the basis of the known behavior and/or preferences of a wide spectrum of users or designers or via some other suitable means) and the automatically synthesized profile subjected to optional editing by users or designers.

The background, partitions, associated user interface elements, user or designer profiles and any required configuration information – and more generally any aspect of the user interface - could be managed as elements in a universal file format. Such a universal file format would specify a header identifying the file type and containing information as to the number, types, locations and sizes of the elements it contains. Each element in the file is in turn described by a header specifying the type of the element, its size and any relevant data or attributes and the types, locations and sizes of any additional elements it contains. By making use of self-describing elements in the manner explained in the foregoing, the universal file format would be able to store an arbitrary element having an arbitrary number and types of other such elements embedded in it.

MEDICAL / SCIENTIFIC APPLICATION

Some of the advantages of the principles of the present invention over the prior art will be demonstrated via the application of the principles of the present invention to the management of data in the medical / scientific field.

The data comprised 8-dimensional or 8D representations of miscellaneous colonies of bacteria that were studied under a microscope to monitor their development under a specific set of conditions. Some species in the colonies exhibited luminescence or emitted detectable light when exposed to a specific chemical or reagent. 3-dimensional or 3D video micrographs effectively comprising 4-dimensional or 4D image data were captured under controlled conditions. Each video was of the same duration. Thus, the primary data for the example is the 4D image data (referred to as **4D_IMAGE** for the purpose of this illustration) that the 3D video micrographs represent. One of the parameters used in the capture of the data was the wavelength of the light used. Three separate wavelengths representing the Red, Blue and Green colors on the color spectrum were used. This constituted the fifth dimension (referred to as **WAVELENGTH** for the purpose of this illustration) for the data. Under the

WAVELENGTH dimension, the original population of the colonies at the beginning of each recording could be represented. The sixth dimension for the data was the environment (labeled **ENVIRONMENT** for the purpose of this illustration) under which the bacteria were monitored. Two separate environments were utilized. In one environment, the bacteria were exposed to a hostile fungus that caused a certain portion of the colony to die off. In the other environment, no hostile fungus was present and conditions were favorable for the development of the colonies. The **ENVIRONMENT** dimension could be adapted to represent the percentage change in the population at the end of the recording as compared with the original population at the beginning of the recording. As mentioned earlier, some species of bacteria in the colonies exhibited photoluminescence in the present of a certain chemical or reagent. The presence or absence of this photoluminescence-causing chemical or reagent and the measurable effect produced (labeled **PHOTOLUMINESCENCE** for the purpose of this illustration) constituted the seventh dimension for the data. Video was captured for a week every other day of the week beginning on a Monday. Hence, data was available for Monday, Wednesday, Friday and Sunday. There was no data for Tuesday, Thursday and Saturday. The day of the week (labeled **DAY** for the purpose of this illustration) on which data was captured represented the eighth dimension for the data.

3D matrix representation has been adopted for the management of the user interface instances for this 8D medical / scientific application of the principles of the present invention.

The first four dimensions are tied up with the 3D video data – constituting a 4D image data set (referred to as **4D_IMAGE** for the purpose of this illustration) with pixel elements.

For the **fifth dimension** (the first row or **row 1** of the 2D face of the 3D representative matrix), the wavelength of the light at which the 3D video data was captured has been utilized. This dimension is labeled **WAVELENGTH** for the purpose of this illustration and permits three separate values, namely **RED** (for the red light wavelength), **GREEN** (for the green light wavelength) and **BLUE** (for the blue light wavelength). Instances associated with the **WAVELENGTH** variable could be configured to display the 3D video data as captured under **RED**, **GREEN** or **BLUE** light in the appropriate columns.

The **sixth dimension** (the second row or **row 2** of the 2D face of the 3D representative matrix) is tied up with the environment (labeled **ENVIRONMENT** for the purpose of this illustration) under which the bacterial colonies develop. **FAVORABLE** is a value of the **ENVIRONMENT** parameter indicating a favorable environment marked by the absence of hostile fungi while **HOSTILE** is a value of the **ENVIRONMENT** parameter indicating a hostile environment marked by the presence of hostile fungi. Instances associated with the **ENVIRONMENT** parameter could be configured to display only those bacterial populations that remain alive under **HOSTILE** or **FAVORABLE** environmental conditions as well as information – possibly in the form of text or textual elements – indicating the percentage of population of remaining bacteria compared with the original population of the colony. Note that for each instance or column entry within the sixth dimension (the second row or **row 2** of the 2D face of the 3D representative matrix), the **ENVIRONMENT** information unique to the sixth dimension could be combined with the associated wavelength information from the fifth dimension. Thus, column entries within the second row on any surface on which instances are disposed could display the 3D video in **RED**, **GREEN** or **BLUE** light (depending on the column number – for example **RED** for column 1, **GREEN** for column 2 and **BLUE** for column 3) filtered to display only those bacterial populations that remain alive in the presence of a **HOSTILE** or **FAVORABLE** environment depending on the specific value of the **ENVIRONMENT** parameter specified for the affected instance.

Luminescence (labeled **PHOTOLUMINESCENCE** for the purpose of this illustration) in the presence of a certain chemical or reagent constitutes the subject of the seventh dimension (the third row or **row 3** of the 2D face of the 3D representative matrix) and instances associated with this parameter could be configured to display the 3D video with a filter permitting only the population of luminescent bacterial species to be displayed and possibly with information – in the form of text or textual elements – indicating the level of light emitted from the bacterial population. Once again, note that for each instance or column entry within the seventh dimension (the third row or **row 3** of the 2D face of the 3D representative matrix), the **PHOTOLUMINESCENCE** information unique to the seventh dimension could be combined with the associated wavelength information from the fifth dimension as well as the associated environment information from the sixth dimension. Thus, column entries within the third row on any surface on which instances are disposed could display the 3D video in **RED**, **GREEN** or **BLUE** light (depending on the column number – for example **RED** for column 1, **GREEN** for column 2 and **BLUE** for column 3) filtered to display only those bacterial populations that **BOTH exhibit luminescence AND remain alive** in the presence of a **HOSTILE** or **FAVORABLE** environment depending on the specific value of the **ENVIRONMENT** parameter specified for the affected instance.

From the foregoing, it should be very clear to one of ordinary skill in the art that this MEDICAL / SCIENTIFIC application demonstrates that an arbitrary dimensional user interface created in accordance with the principles of the present invention indeed permits any arbitrary dimensional data to be faithfully and directly represented by the instances of the user interface. As explained earlier, the cumulative effects of successive dimensions could be reflected faithfully and directly in any given instance. This is very clear from the faithful and direct representation of the cumulative effects of the fifth dimension (**WAVELENGTH**), the sixth dimension (**ENVIRONMENT**) and the seventh dimension (**PHOTOLUMINESCENCE**) on all affected column entries or user interface instances as shown here.

The day of the week (labeled **DAY** for the purpose of this illustration) on which data was captured constitutes the subject of the eighth dimension (the **third dimension** of the **3D representative matrix**) and instances associated with this parameter could be disposed on a separate surface for a specific value of the parameter.

In this MEDICAL / SCIENTIFIC application, data was available for **Monday**, **Wednesday**, **Friday** and **Sunday**.

Note that although data was not available for **Tuesday**, **Thursday** and **Friday**, the principles of the present invention permit the synthesis of data or instances for each of these days for which data was not available. Furthermore, data could be synthesized for sub-day points, that is, for any hour or in fact for any time (even down to the millisecond or down to any arbitrary desired time precision, limited only by the precision of the hardware and/or software systems on which the embodiment of the present invention is implemented) whatsoever between the first day – **Monday** – and the last day – **Sunday** – for which data was available.

Consequently, the present invention permits seamless navigation of the arbitrary dimensional data (8D data in this specific application) even for points or days for which data was not available or for which data was not collected or captured. The synthesis or creation of instances for Tuesday on the basis of the principles of the present invention will be illustrated shortly.

For this application, the 2D matrix row values and their associated parameters and actual data dimensions are as follows:

i = 1 (WAVELENGTH {RED, GREEN, BLUE}) [FIFTH DIMENSION]

i = 2 (ENVIRONMENT {HOSTILE, FAVORABLE}) [SIXTH DIMENSION]

i = 3 (PHOTOLUMINESCENCE {INTENSITY}) [SEVENTH DIMENSION]

The following table shows the associated column entries or instances for **Monday**.

$\{UI_{11}\}_{Monday}$	$\{UI_{12}\}_{Monday}$	$\{UI_{13}\}_{Monday}$
$\{UI_{21}\}_{Monday}$	$\{UI_{22}\}_{Monday}$	$\{UI_{23}\}_{Monday}$
$\{UI_{31}\}_{Monday}$	$\{UI_{32}\}_{Monday}$	$\{UI_{33}\}_{Monday}$

The following table lists the associated column entries or instances for **Wednesday**.

$\{UI_{11}\}_{Wednesday}$	$\{UI_{12}\}_{Wednesday}$	$\{UI_{13}\}_{Wednesday}$
$\{UI_{21}\}_{Wednesday}$	$\{UI_{22}\}_{Wednesday}$	$\{UI_{23}\}_{Wednesday}$
$\{UI_{31}\}_{Wednesday}$	$\{UI_{32}\}_{Wednesday}$	$\{UI_{33}\}_{Wednesday}$

The following table depicts the associated column entries or instances for **Friday**.

$\{UI_{11}\}_{Friday}$	$\{UI_{12}\}_{Friday}$	$\{UI_{13}\}_{Friday}$
$\{UI_{21}\}_{Friday}$	$\{UI_{22}\}_{Friday}$	$\{UI_{23}\}_{Friday}$
$\{UI_{31}\}_{Friday}$	$\{UI_{32}\}_{Friday}$	$\{UI_{33}\}_{Friday}$

The following table illustrates the associated column entries or instances for **Sunday**.

$\{UI_{11}\}_{Sunday}$	$\{UI_{12}\}_{Sunday}$	$\{UI_{13}\}_{Sunday}$
$\{UI_{21}\}_{Sunday}$	$\{UI_{22}\}_{Sunday}$	$\{UI_{23}\}_{Sunday}$
$\{UI_{31}\}_{Sunday}$	$\{UI_{32}\}_{Sunday}$	$\{UI_{33}\}_{Sunday}$

INTERPRETATION OF INSTANCE CONTENTS

Each instance could contain user interface elements such as buttons, sliders, text, and so on for data navigation (including 3D video navigation in this example) and for page flipping to enable display of data for other days (and in fact for any arbitrarily selected day, time or point between the first and the last days for which data was collected) apart from the specific day of the week which the instance represents. This feature could be made common to all instances. In particular, user interface elements enabling arbitrary selection of data for any desired day of the week for display could be provided in each instance.

The following table provides illustrative descriptions of the contents of the column entries or instances for **Monday**. Data for other days for which data was available or for which data was collected or captured such as Wednesday, Friday and Sunday could be interpreted in a similar fashion.

Displays 4D image data or 3D video of original bacterial colony rendered in **RED** light. $\left\{UI_{11}\right\}_{Monday}$	Displays 4D image data or 3D video of original bacterial colony rendered in **GREEN** light. $\left\{UI_{12}\right\}_{Monday}$	Displays 4D image data or 3D video of original bacterial colony rendered in **BLUE** light. $\left\{UI_{13}\right\}_{Monday}$
Displays filtered 4D image data or 3D video of	Displays filtered 4D image data or 3D video of	Displays filtered 4D image data or 3D video of

bacterial colony showing only those bacteria remaining alive in hostile or favorable environment in **RED** light. Contains user interface elements for selection of desired environment variable, namely, **HOSTILE or FAVORABLE.** Displays representation of percentage of population of original colony remaining	bacterial colony showing only those bacteria remaining alive in hostile or favorable environment in **GREEN** light. Contains user interface elements for selection of desired environment variable, namely, **HOSTILE or FAVORABLE.** Displays representation of percentage of population of original colony remaining	bacterial colony showing only those bacteria remaining alive in hostile or favorable environment in **BLUE** light. Contains user interface elements for selection of desired environment variable, namely, **HOSTILE or FAVORABLE.** Displays representation of percentage of population of original colony remaining

alive after exposure to hostile or favorable environment. $\{UI21\}_{Monday}$	alive after exposure to hostile or favorable environment. $\{UI22\}_{Monday}$	alive after exposure to hostile or favorable environment. $\{UI23\}_{Monday}$
Displays filtered 4D image data or 3D video of bacterial colony showing only those luminescent bacteria remaining alive in hostile or favorable environment in **RED** light. $\{UI31\}_{Monday}$ Displays representation of intensity for **LUMINESCENCE.**	Displays filtered 4D image data or 3D video of bacterial colony showing only those luminescent bacteria remaining alive in hostile or favorable environment in **GREEN** light. $\{UI32\}_{Monday}$ Displays representation of intensity for **LUMINESCENCE.**	Displays filtered 4D image data or 3D video of bacterial colony showing only those luminescent bacteria remaining alive in hostile or favorable environment in **BLUE** light. $\{UI33\}_{Monday}$ Displays representation of intensity for **LUMINESCENCE.**

SYNTHESIZING OR CREATING NEW INSTANCES FOR TUESDAY

As noted previously, the principles of the present invention permit the creation or synthesis of new instances for days (in the case of this specific MEDICAL / SCIENTIFIC application) or more generally for points or times for which data was not originally available.

Using tri-linear interpolation, any desired instance for any desired day or hour or minute or second or any arbitrarily selected time between the first day – Monday – and the last day – Sunday – for which instance data was available could be synthesized or created.

To demonstrate the synthesis or creation of a new instance for Tuesday for which no data was available or for which no data was collected or captured, consider the middle column entry or instance for the representative matrix for Tuesday, namely, $\{UI_{22}\}_{Tuesday}$.

As is clear from the forgoing, no data was available for this instance labeled $\{UI_{22}\}_{Tuesday}$. To synthesize or create this new instance, note that this instance is located at the center of the 3 x 3 representative matrix of instances for **Tuesday**. Consequently, this instance could be treated as located at x, y, z coordinates (0.5, 0.5, 0.5) by considering the origin of the coordinate system comprising the eight neighboring instances, namely, $\{UI_{31}\}_{Monday}$, $\{UI_{33}\}_{Monday}$, $\{UI_{11}\}_{Monday}$, $\{UI_{13}\}_{Monday}$, $\{UI_{31}\}_{Wednesday}$, $\{UI_{33}\}_{Wednesday}$, $\{UI_{11}\}_{Wednesday}$, $\{UI_{13}\}_{Wednesday}$ to be located at the instance labeled $\{UI_{31}\}_{Monday}$ and assuming a normalized 3D coordinate system with unit length (1.0) as the distance between the two neighboring 3 x 3 representative matrices for Monday and Wednesday.

For simplicity, set $\{UI_{22}\}_{Tuesday}$ equal to T(0.5, 0.5, 0.5) and interpret the resulting value of $\{UI_{22}\}_{Tuesday}$ as a pixel-wise operator that could be used to compute, synthesize or create any desired pixel value for the affected instance on **Tuesday**.

Thus in the following equation, replace x with 0.5, y with 0.5, z with 0.5 and set the V000 to V111 values to their corresponding column entries or instances within the neighboring 3 x 3 representative matrices for **Monday** and **Wednesday**.

T(x, y, z) = (V000*(1-x) + V100*x) * (1-y) + ((V010*(1-x)+V110*x)*y)*(1-z)+V001*(1-x)+V101*x)*(1-y)+((V011*(1-x)+V111*x) *y)*z

Where:

V00 = $\{UI_{31}\}_{Monday}$

V100= $\{UI_{33}\}_{Monday}$

V010= $\{UI_{11}\}_{Monday}$

V110= $\{UI_{13}\}_{Monday}$

V001= $\{UI_{31}\}_{Wednesday}$

V101= $\{UI_{33}\}_{Wednesday}$

V011= $\{UI_{11}\}_{Wednesday}$

V111= $\{UI_{13}\}_{Wednesday}$

Now for any given pixel with coordinates x, y, z (on the image coordinate system and not on the imaginary conceptual 3D matrix surface used to compute the value of T(x, y, z)) and specific time (represented by the frame number for that pixel within the 3D video) within the new instance for **Tuesday**, the correct value can be computed using the value obtained for T(0.5, 0.5, 0.5) utilizing the corresponding pixel values for the corresponding neighboring instances from **Monday** and **Wednesday** as illustrated.

By selecting appropriate values for x, y, z in the equation for the evaluation of T(x, y, z) and adopting substitutions equivalent to those illustrated in the forgoing for **Tuesday**, any desired new instance could be synthesized, predicted, computed or created using existing neighboring instances.

As already noted, one of ordinary skill in the art would appreciate that this is not the only way to synthesize new user interface instances but that any suitable means could be employed including, but not limited to, means that do not require the utilization of existing instances. Other interpolation schemes could also be applied.

Very Large Data Management

One common way to represent data associated with user interface instances is to interpret the data as an image comprising groups of pixels or groups of fractions of pixels for sub-pixel precision. Such pixels could be interpreted as representing suitably formatted colors such as the typical red-blue-green-alpha (RGBA) color quad as well as associated 2D or 3D coordinates. Other interpretations suited to different applications are possible.

It should be apparent to one of ordinary skill in the art that the data associated with the arbitrary dimensional instances permitted by the principles of the present invention could become very large and difficult to manage in resource-limited environments such as the present Internet.

Efficient data management techniques pertaining to such applications including the use of predictive loading of relevant data and possible subsequent presentation on a display window or computer monitor or any other suitable device or system based on a dynamic prediction of the user's point of view within the data stream could be applied to enable practical implementation and acceptable performance of applications of the present invention for very large data sets and related applications on off-the-shelf personal computer systems.

When data associated with any selected instance is characterized in the form of pixel values of images representing the data, dynamic view prediction of the user's field of view and associated intelligent data management techniques could be used to facilitate efficient data navigation. Based on the observation that interactive rendering of the image data set involves the display of a relatively small (compared to the size of the underlying image data itself) view window, the preferred embodiment of the present invention teaches the use of a robust two-tier or bi-level representation of the image. The first level contains a virtual view of an entire image frame as a single continuous set of pixels. FIG. 6 illustrates the first level for a two-dimensional image frame of width P_W and height P_H pixels. Coordinate axes are labeled X-axis and Y-axis in FIG. 6. The region of interest or view window is indicated as V in FIG. 6. Since a single image frame can be very large, it is generally impractical to attempt to load the entire image frame (typically corresponding to tens of gigabytes or even terabytes or more of physical memory for certain applications) into memory at once. Consequently, the second level comprises a segmentation or partitioning of each image frame into distinct image blocks of a size and color depth that facilitates straightforward manipulation on an average personal computer. This partitioning scheme is shown in FIG. 7, where the image of FIG. 6 has been segmented into NxM distinct image blocks labeled B_{11}, $B_{12},\ldots,B_{NM}$. The size of each partition can be chosen such that the view window, V, straddles just a couple of image blocks. In this case only those image blocks in the second layer that are covered or straddled by the view window need be loaded into memory for the manipulation or rendering of the view, leading to a significantly reduced memory footprint.

The use of a two-tier image representation scheme permits alternate views of the image data that make further manipulation easier. For example, the simplicity of the first level permits the applications of a multi-resolution pyramid representation of the image data, such as that described by Peter J. Burt et al. in “The Laplacian Pyramid as a Compact Image Code”, IEEE Transactions on Communications, 1983, pp. 532-540, for efficient compression, storage and transmission and optionally for adaptive rendering that maintains a constant frame rate. A thumbnail of the entire image could also be generated at the first level. Such a thumbnail could be used to display a lower resolution version of the view window while waiting for image data to be retrieved and/or decompressed. Furthermore, the dynamic view prediction and on-demand loading algorithms described hereinafter are readily applicable to the second tier's image block representation.

Following is an outline of the process of visualizing the data sets according to the preferred embodiment. First, a view window V is specified as illustrated in FIG. 6. The view window represents the segment of the current image frame that is indicated by the view parameters. In a given implementation, three view parameters such as the pan angle, the tilt angle or azimuth and the zoom or scale factor could be used to control the view. Other relevant view parameters or factors could be considered as appropriate for any given application. User input could be received via the keyboard and/or mouse clicks within the view window. Suitable gesture recognition interfaces or touch-based interfaces or brain-computer interfaces or any other suitable interface could be used to receive input, provide feedback or generally enable user interaction. Alternatively, a head-mounted display and orientation sensing mechanism could also be used. Views could be generated based on view window size and received input. In order to facilitate interactive rendering without the latencies and other limitations associated with the prior art, the rate of change of each of the view parameters with respect to time could be computed dynamically. The computed rate of change could then be used to predict the value of the parameter at any desired time in the past or future.

The following equation illustrates the use of the dynamic view prediction algorithm for a specific view parameter P - which could be the pan, tilt, zoom level, or any other suitable parameter.

$$p = p_0 + KaT$$

In the foregoing equation, P is the predicted value of the parameter at time T, p_0 is the current value of the parameter, a is the dynamically computed rate of change of the parameter with respect to time and K is a scale factor, usually 1. The values of the parameters predicted by the foregoing equation could be used to determine which specific image blocks need to be loaded into memory at any given time.

A computer software implementation using a background thread dedicated to loading those image blocks that are covered by the current view as well as any additional image blocks that might be needed for rendering the view in the future or past, that is, a number of future or past time steps, could be used. The exact value or duration (or optimum value for applications in which variations in the time step are acceptable) of the time step depends on the requirements of a given application. While typical applications on a desktop computer used alone or in association with the current Internet might offer acceptable performance with time steps measured in milliseconds and generally under one second or some other similar figure, other more demanding applications may require much shorter time steps. Much longer time steps in the order of seconds, minutes, hours or longer could be acceptable in less demanding or other unique applications. Since the number of image blocks per frame is usually small, it is practical to preload, pre-fetch or pre-synthesize as appropriate, image blocks that would be required for rendering one or more time steps ahead or behind the current presentation time or the current time step – permitting smooth rendering at real-time rates. Hence the prediction of the subset of data that would correspond to the selected region of interest one or more time steps separate from the current presentation time or the current time step as explained in the foregoing could be utilized to permit smooth and efficient navigation of very large data sets even on resource-limited systems. The image data could be distributed from a server over the Internet or other network or accessed from local storage on a host computer. Any other alternative source and method of distribution could be used where appropriate.

It should be obvious to one of ordinary skill in the art that although data represented as images has been used to illustrate the concepts for very large data navigation according to the principles of the present invention, the invention is not limited to image data but could be applied directly to any large data. For simplicity and convenience, the data could be converted or transformed into image format where appropriate and then converted or transformed back into the original format as demanded by specific applications of the present invention. Raw data in any other format apart from the image format could also be processed directly as illustrated for images or in any covered equivalent or alternative manner to facilitate smooth and efficient navigation of the data even in resource-limited environments.

Studies with image visualization systems have consistently shown that the use of a damping or inertial function to facilitate gradual changes in view parameters leads to the perception by users of a vastly smoother, more natural and more intuitive viewing process. This observation can be exploited by the dynamic view prediction algorithm of the present invention to provide smooth, interactive distribution and visualization of very large image data sets even over relatively slow network connections such as the current Internet and other bandwidth-limited scenarios without the latencies and other deficiencies associated with the prior art. The gradually changing view parameters would then permit many more future or past time steps to be computed, preloaded, pre-fetched and/or pre-synthesized as appropriate to a greater degree of accuracy.

It should be noted that while the foregoing preferred embodiments for the management of very large data sets from systems based on the principles of the present could be appropriate in situations where memory, computing and associated resources are limited with respect to the amounts of data processing required for effective utilization of the system, much simpler and more straightforward data management techniques could be employed in situations with fewer data points from the instances without deviating from the present invention.

The foregoing preferred embodiments do not exhaust the scope of the present invention. In particular, one of ordinary skill in the art would readily appreciate that any graphical user interface including, but not limited to, contemporary user interfaces such as those available in widely used operating systems such as Microsoft Windows, Apple OS (iPhone OS, iPad OS, and so on), Google Android OS and so on, could readily be converted into a much more versatile, useful, responsive and efficient arbitrary dimensional graphical user interface by applying the principles of the present invention.

Any graphical user interface could be converted into an arbitrary dimensional ($\alpha - \dim ensional$) user interface according to the principles of the present invention by disposing one or more instances of the user interface converted into arbitrary dimensional instances on the basis of the principles of the present invention as disclosed in the foregoing specification and suggested by equivalents and alternatives thereto on one or more arbitrarily-sized and arbitrarily-shaped surfaces. The foregoing embodiments illustrate how this conversion could be carried out.

It should be understood that numerous alternative embodiments and equivalents of the invention described herein may be employed in practicing the invention and that all such alternative embodiments and equivalents fall within the scope of the present invention.

Claims

What is claimed is:

1. A method for creating a graphical user interface comprising the computer-implemented steps of:

dividing a background image comprising arbitrarily-sized and arbitrarily-shaped arbitrary dimensional elements into one or more arbitrarily-sized and arbitrarily-shaped arbitrary dimensional partitions, each partition pertaining to one of the elements;

converting each of the partitions into an individual graphical user interface window comprising user interface elements,

wherein each partition is individually associated with a set of rules that defines attributes and behaviors of the partition, and the rules are based on the nature of the background image underlying the partition,

wherein each partition may be altered by a user to assume any arbitrary size and any arbitrary shape without distorting the underlying background image, and displaying representations of the partitions and permitting interaction with the partitions.

2. The method of Claim 1 wherein said partitions are identified manually.

3. The method of Claim 1 wherein said partitions are identified automatically.

4. The method of Claim 1 wherein said partitions are identified semi-automatically.

5. The method of Claim 1 wherein said partitions are identified on the basis of a user or designer profile.

6. The method of Claim 5 wherein said user or designer profile is synthesized manually.

7. The method of Claim 5 wherein said user or designer profile is synthesized automatically.

8. The method of Claim 5 wherein said user or designer profile is synthesized semi-automatically.

9. The method of Claim 5 wherein said user or designer profile is managed using a universal file format.

10. The method of Claim 1 wherein said partitions are conceptual.

11. The method of Claim 1 wherein said partitions are literal.

12. The method of Claim 1 wherein at least one partition contains at least one user interface element.

13. The method of Claim 1 wherein no partition contains any user interface element.

14. The method of Claim 1 wherein said user interface is at most two-dimensional.

15. The method of Claim 1 wherein said user interface is at most three-dimensional.

16. The method of Claim 1 wherein said user interface is at most four-dimensional.

17. The method of Claim 1 wherein said user interface is at most five-dimensional.

18. The method of Claim 1 wherein said user interface is at most six-dimensional.

19. The method of Claim 1 wherein said user interface is at least seven-dimensional.

20. The method of Claim 1 wherein at least one new instance of said user interface is synthesized.

21. The method of Claim 20 wherein said synthesis is carried out using at least one existing instance of said user interface.

22. The method of Claim 1 wherein one or more design characteristics of one or more partitions are guided by one or more intended presentation characteristics of the affected partition.

23. The method of Claim 22 wherein said partitions are identified manually.

24. The method of Claim 22 wherein said partitions are identified automatically.

25. The method of Claim 22 wherein said partitions are identified semi-automatically.

26. The method of Claim 22 wherein said partitions are identified on the basis of a user or designer profile.

27. The method of Claim 26 wherein said user or designer profile is synthesized manually.

28. The method of Claim 26 wherein said user or designer profile is synthesized automatically.

29. The method of Claim 26 wherein said user or designer profile is synthesized semi-automatically.

30. The method of Claim 26 wherein said user or designer profile is managed using a universal file format.

31. The method of Claim 22 wherein said partitions are conceptual.

32. The method of Claim 22 wherein said partitions are literal.

33. The method of Claim 22 wherein at least one partition contains at least one user interface element.

34. The method of Claim 22 wherein no partition contains any user interface element.

35. The method of Claim 22 wherein said user interface is at most two-dimensional.

36. The method of Claim 22 wherein said user interface is at most three-dimensional.

37. The method of Claim 22 wherein said user interface is at most four-dimensional.

38. The method of Claim 22 wherein said user interface is at most five-dimensional.

39. The method of Claim 22 wherein said user interface is at most six-dimensional.

40. The method of Claim 22 wherein said user interface is at least seven-dimensional.

41. The method of Claim 22 wherein at least one new instance of said user interface is synthesized.

42. The method of Claim 41 wherein said synthesis is carried out using at least one existing instance of said user interface.

43. The method of Claim 22 wherein said user interface is managed using a universal file format.

44. The method of Claim 1 wherein navigation of data associated with at least one instance of said user interface comprises steps of:

defining a region of interest within the data;

presenting a subset of the data corresponding to said region of interest;

predicting the subset of said data that would correspond to said region of interest one or more time steps separate from the current presentation time;

utilizing said predicted subset in said navigation of said data.

45. The method of Claim 44 wherein said one or more time steps separate from the current presentation time are behind the current presentation time.

46. The method of Claim 44 wherein said one or more time steps separate from the current presentation time are ahead of the current presentation time.

47. The method of Claim 22 wherein navigation of data associated with at least one instance of said user interface comprises steps of:

defining a region of interest within the data;

presenting a subset of the data corresponding to said region of interest;

predicting the subset of said data that would correspond to said region of interest one or more time steps separate from the current presentation time;

utilizing said predicted subset in said navigation of said data.

48. The method of Claim 47 wherein said one or more time steps separate from the current presentation time are behind the current presentation time.

49. The method of Claim 47 wherein said one or more time steps separate from the current presentation time are ahead of the current presentation time.

50. A method for creating an arbitrary dimensional graphical user interface comprising steps of:

creating one or more arbitrary dimensional instances of said user interface;

disposing said one or more arbitrary dimensional instances of said user interface on one or more arbitrarily-sized and arbitrarily-shaped surfaces;

permitting interaction with said one or more arbitrary dimensional instances of said user interface.

51. The method of Claim 50 wherein at least one new instance of said user interface is synthesized.

52. The method of Claim 50 wherein said synthesis is carried out using at least one existing instance of said user interface.

53. The method of Claim 50 wherein said user interface is managed using a universal file format.

54. The method of Claim 50 wherein navigation of data associated with at least one instance of said user interface comprises steps of:

defining a region of interest within the data;

presenting a subset of the data corresponding to said region of interest;

predicting the subset of said data that would correspond to said region of interest one or more time steps separate from the current presentation time;

utilizing said predicted subset in said navigation of said data.

55. The method of Claim 54 wherein said one or more time steps separate from the current presentation time are behind the current presentation time.

56. The method of Claim 54 wherein said one or more time steps separate from the current presentation time are ahead of the current presentation time.

57. The method of Claim 50 wherein one or more design characteristics of one or more elements within said one or more arbitrary dimensional instances of said user interface are guided by one or more intended presentation characteristics of the affected element.

58. The method of Claim 57 wherein at least one new instance of said user interface is synthesized.

59. The method of Claim 58 wherein said synthesis is carried out using at least one existing instance of said user interface.

60. The method of Claim 57 wherein said user interface is managed using a universal file format.

61. The method of Claim 57 wherein navigation of data associated with at least one instance of said user interface comprises steps of:

defining a region of interest within the data;

presenting a subset of the data corresponding to said region of interest;

predicting the subset of said data that would correspond to said region of interest one or more time steps separate from the current presentation time;

utilizing said predicted subset in said navigation of said data.

62. The method of Claim 61 wherein said one or more time steps separate from the current presentation time are behind the current presentation time.

63. The method of Claim 61 wherein said one or more time steps separate from the current presentation time are ahead of the current presentation time.

FIG. 1

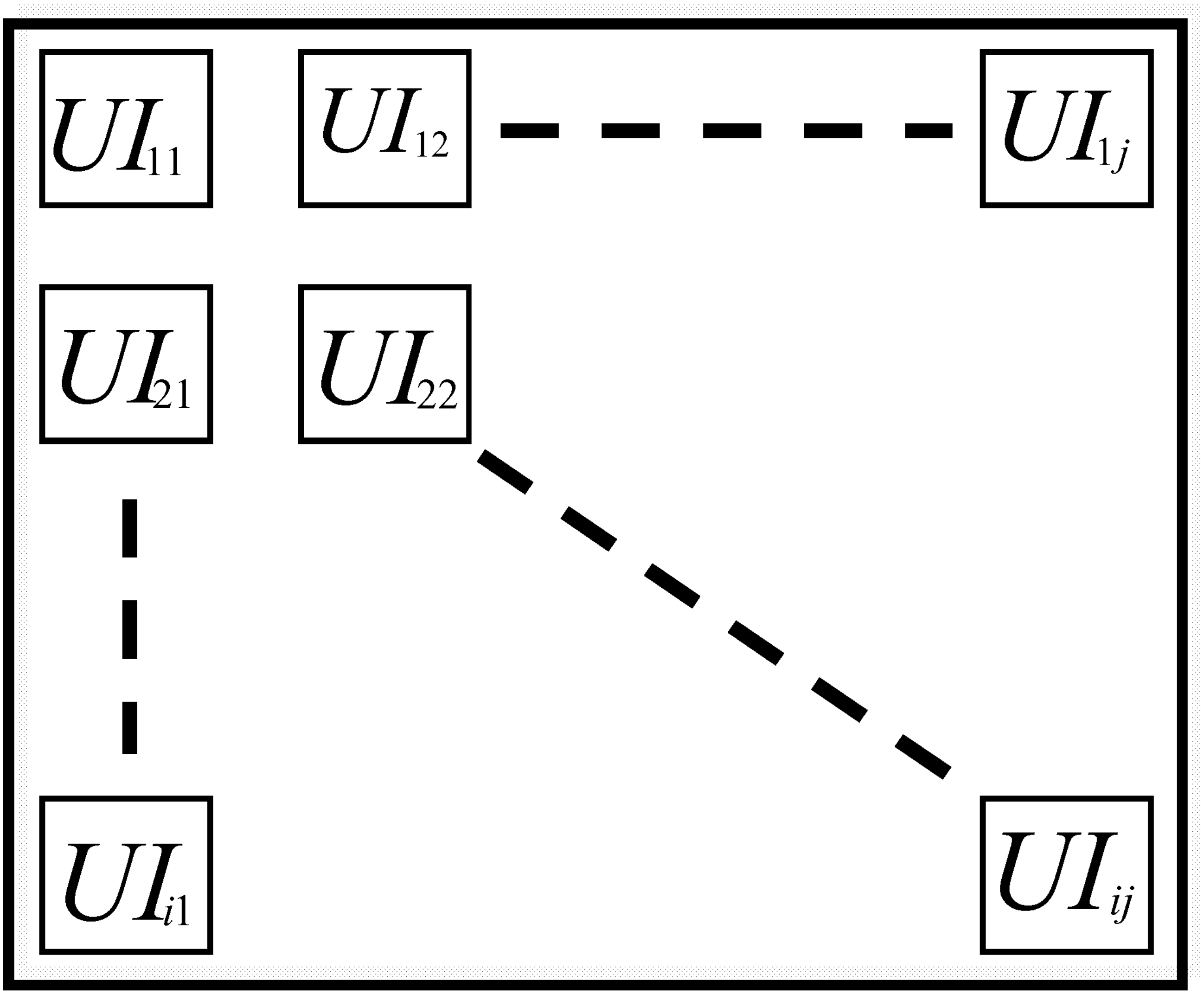


FIG. 2

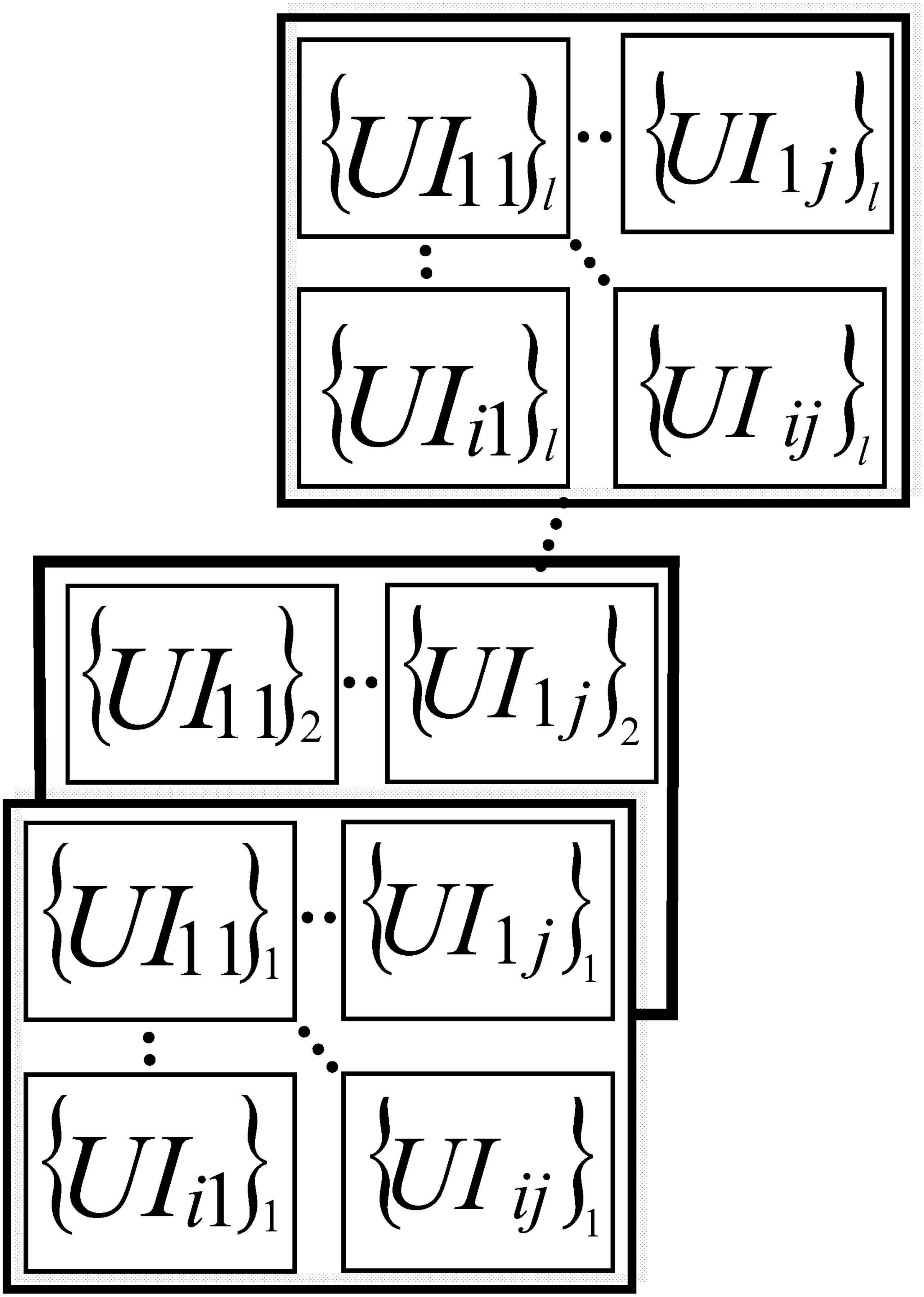

$\{UI_{11}\}_l$
$\{UI_{1j}\}_l$
$\{UI_{i1}\}_l$
$\{UI_{ij}\}_l$
$\{UI_{11}\}_2$
$\{UI_{1j}\}_2$
$\{UI_{11}\}_1$
$\{UI_{1j}\}_1$
$\{UI_{i1}\}_1$
$\{UI_{ij}\}_1$

FIG. 3

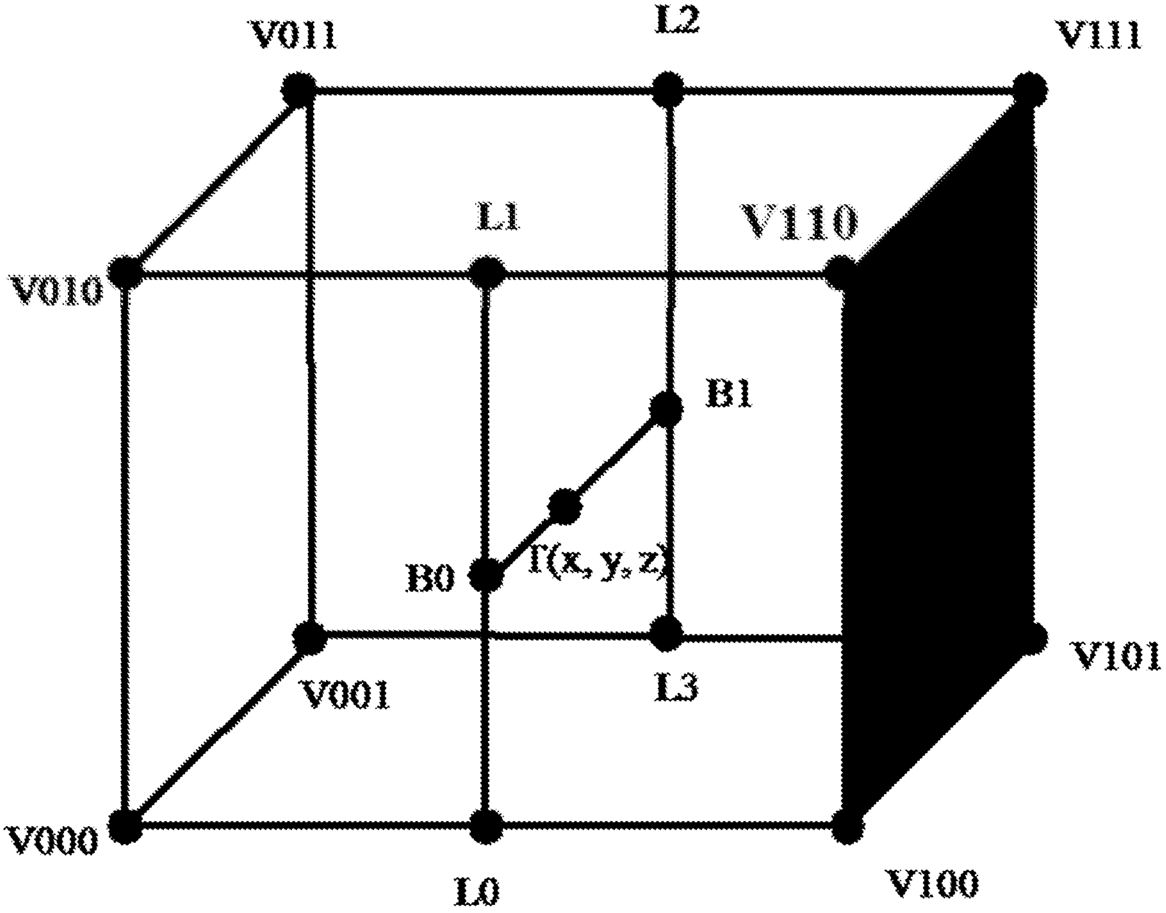


FIG. 4

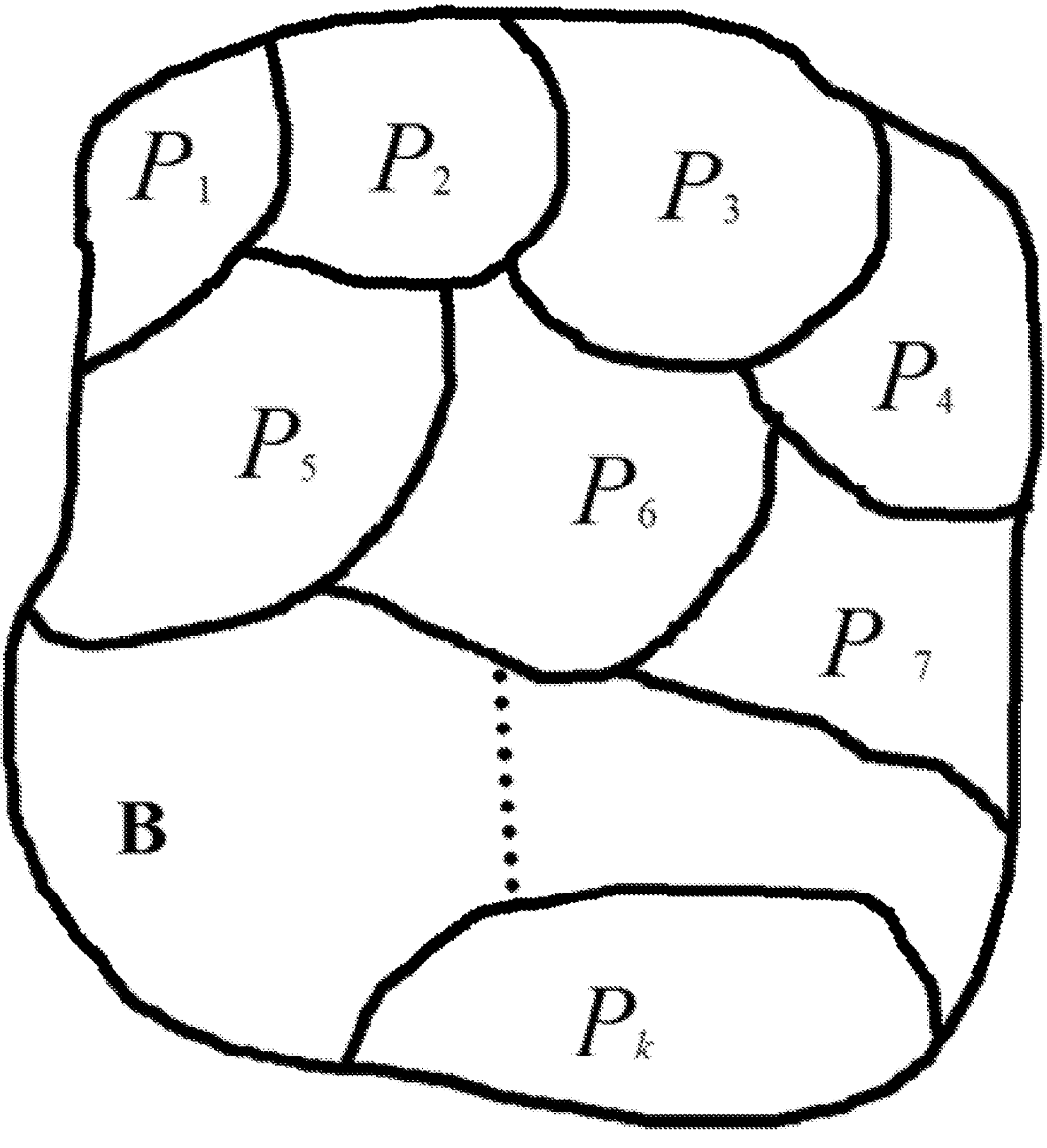


FIG. 5

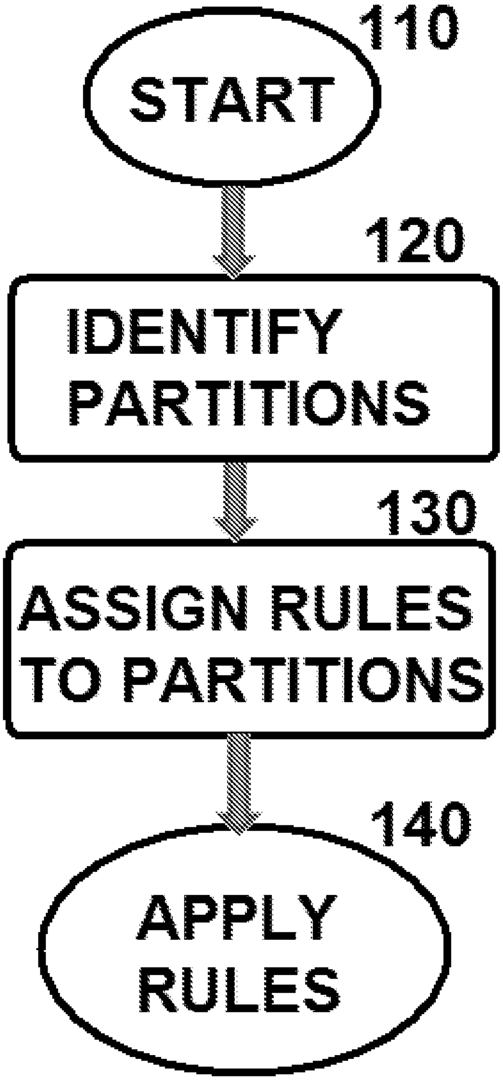

110
START
120
IDENTIFY PARTITIONS
130
ASSIGN RULES TO PARTITIONS
140
APPLY RULES

FIG. 6

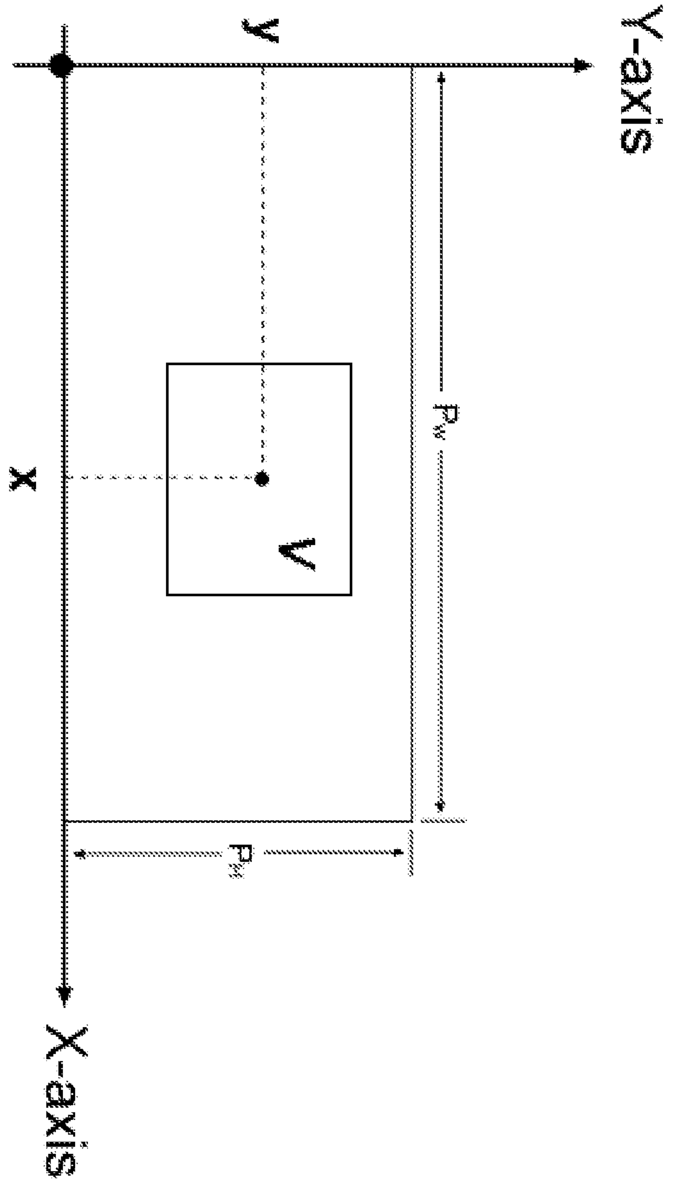
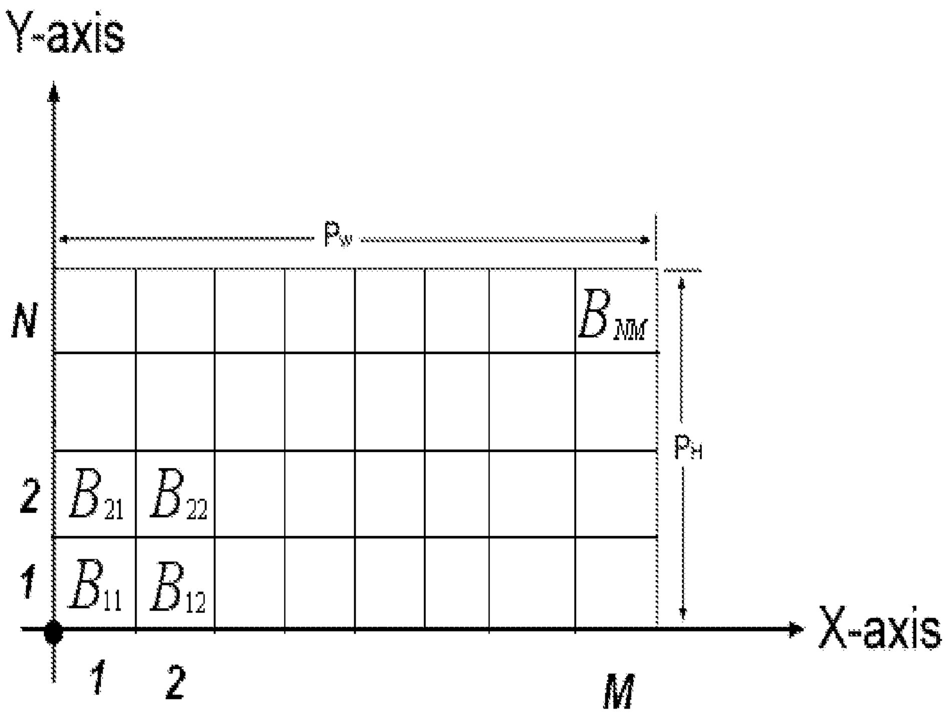


FIG. 7



INTERNATIONAL SEARCH REPORT

International application No
PCT/US2012/066159

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F9/44
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F G06T

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal, WPI Data, COMPENDEX, INSPEC, PAJ

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2012/042268 A1 (EKPAR FRANK EDUGHOM [JP]) 16 February 2012 (2012-02-16)	1-19, 22-40, 43,50, 53,57,60
Y	figures 1,2 paragraph [0013] - paragraph [0027]	20,21, 41,42, 44-49, 51,52, 54-56, 58,59, 61-63

-/--

☒ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search 8 April 2013	Date of mailing of the international search report 15/04/2013
Name and mailing address of the ISA/ European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Authorized officer Del Chiaro, Silvia

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2010/054578 A1 (EKPAR FRANK EDUGHOM [JP]) 4 March 2010 (2010-03-04) figures 1,2 paragraph [0013] - paragraph [0024] -----	20,21, 41,42, 44-49, 51,52, 54-56, 58,59, 61-63
X	US 2005/225572 A1 (EKPAR FRANK E [JP]) 13 October 2005 (2005-10-13) claim 1 paragraph [0006] - paragraph [0009]; figure 1 -----	1,50

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/US2012/066159

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2012042268 A1	16-02-2012	NONE	
US 2010054578 A1	04-03-2010	NONE	
US 2005225572 A1	13-10-2005	NONE	