Abstract: Methods, computer program products and programmed data processing systems for converting comic art material into a digital slideshow format suitable for viewing on the display of a data processing device or system, particularly mobile devices such as a telephones, media players or personal digital assistant (PDA) having a relatively small display screen. The invention includes: methods for identifying cell boundaries in comic art works; methods for defining motion paths for panning/scrolling a viewing window over comic art material; methods for facilitating the production of such slideshows by means of automated storyboarding of clips and the use of hierarchical timeline displays allowing timeline sections to be collapsed and expanded; the application of visual zoom effects to slideshow content and slideshow products incorporating such zoom effects.

Title: PROCESSING COMIC ART
Processing Comic Art

FIELD OF THE INVENTION

The present invention relates to methods, computer program products and programmed data processing systems for converting comic art material into a digital slideshow format suitable for viewing on the display of a data processing device or system, particularly but not exclusively on a mobile device such as a telephone, media player or personal digital assistant (PDA) having a relatively small display screen. The invention further concerns aspects of the way in which processed material is delivered to a consumer on a visual display, again including relevant methods, computer program products and programmed data processing systems. The invention further relates to improved digital comic art products.

BACKGROUND TO THE INVENTION

Comic art (sometimes also known as "sequential art") is a form of graphical art consisting of sequences of images which are commonly combined with text, often in the form of speech balloons or image captions. Comic art has evolved into a literary medium with many subgenres.

Forms of printed comics include comic strips (most commonly four panels long) in newspapers and magazines, and longer comic stories in comic books, graphic novels and comic albums. While the present invention is applicable to all forms of comic art it is particularly applicable to comic art consisting of multiple pages, typically dozens or even hundreds of pages.
Comic art generally consists of a number of panels or "cells" intended to be read in a particular sequence. A page usually includes a number of cells. Cells may be rectangular and be laid out in a regular grid or may vary in shape and/or size to suit the content. Speech balloons may extend beyond the border of a cell and overlap adjacent cells. Cells may have well defined borders but the sophistication of contemporary comic art means that the boundaries of individual cells may not be well defined, or content may be shared between two or more cells. The intended sequence of the cells may also vary from the standard left to right/top to bottom layout (or right to left/top to bottom in the case, for example, of Japanese manga material).

There is an increasing demand for comic art content to be made available via computing devices, particularly mobile devices such as telephones, media players and PDAs. Such devices generally have relatively small display screens, so that one problem associated with the delivery of comic art to these kinds of devices is adapting the content to the display. A further problem is the efficient conversion of paper-based content to a suitable digital format.

Digitally formatted comic art content may be delivered to a consumer in a variety of ways. In a simple case, a digitised comic document could be provided as, for example, a PDF file, which a consumer can navigate manually using a conventional file viewer. A preferred approach is to provide the comic art content in the form of a "slideshow" or "animation" that displays the cells in a sequence of frames, which may include sophisticated transitions between cells (e.g. cinematic style transitions such as fades and dissolves) and other effects such as audio or tactile (e.g. vibration) effects associated with particular cells. Where an individual cell is too large to be displayed effectively on a given target display device,
the display window may be panned/scrolled over the area of the cell. This may be done automatically using a simple algorithm to ensure that the complete content of each cell is made visible to the consumer. An alternative approach is to digitise the content on a per-page basis, with the locations and sequence of individual cells identified, and to automatically pan/scroll the display window over the entire page so as to display the cells in the correct sequence. A further alternative is for parts of particular cells to be displayed as sub-sequences of frames within the main cell sequence.

The concept of comics being delivered electronically on a single purchase or subscription basis is presently increasing in popularity in the far eastern market and can be expected to spread to other markets.

The established process for converting printed comics into digital format requires scanning, cleaning, cropping and sequencing (animating) the comic pages and cells. Existing software packages specifically for this purpose rarely provide tools for cleaning and slicing up comic cells. Many authors prefer to use conventional graphics packages for at least some comic art processing functions.

A typical comic book of interest consists of around 200 pages with 4 to 10 cells per page. The task of manually slicing each individual cell is quite laborious. Using existing methods, it may take as long as 14 days to convert one comic book.

An existing digital comic conversion/authoring package provides tools for creating motion paths on large cells or whole page bitmaps, defining the manner in which the display window pans/scrolls across the content. This task again requires manual input from authors in setting up sequence
rectangles (or "frames"). Sequence rectangles are overlay boxes placed on top of comic cells and sequentially numbered to create the motion path between cells.

SUMMARY OF THE INVENTION

In its various aspects, the present invention provides a range of tools that can be used individually or in combination in the process of converting published comics to digital format and packaging them as digital slideshows for viewing on mobile devices with effects such as transitions, motion path, and vibration. The various tools seek, for example, to automate the process of slicing existing comic art into individual cells, to provide semi-automatic methods of slicing cells that cannot readily be distinguished on a fully automatic basis, and to improve the efficiency of manual slicing processes when manual intervention becomes necessary.

Since the process of slicing cells from the original material may involve sub-dividing individual cells into two or more parts, or slicing two or more adjacent cells as a single sliced item, an individual item sliced from the original material is referred to herein as a "clip", as distinguished from a single cell of the original material. A "clip" may thus correspond to a single cell of the original material, to a group of two or more adjacent cells or to a part of a cell. That is, a clip is a discrete item "sliced" from an original comic art bitmap for inclusion in a digital slideshow. As used herein, "frame" means an area of the comic art work corresponding to the display window of the target display device that forms part of the slideshow sequence. A single clip may be the source for a single frame or for a sub-sequence of frames (frames may also be derived from complete pages of original art work as described elsewhere). A slideshow may simply comprise a sequence of static frames. Alternatively, a slideshow may
include panning/scrolling the display window over a clip or page of artwork (as mentioned above) and/or visual zoom effects (as described below). In these cases, a "key frame" is a frame corresponding to the display window defining the beginning or end of a pan, scroll or zoom.

A complete digital comic authoring environment incorporating the tools provided by means of the various aspects of the present invention preferably includes the following features and functionality:
- Integrated solution that allows authors to quickly package existing comic content for delivery to mobile handsets or other devices.
- Intelligent or semi-intelligent tools for slicing comic pages into clips
- Visual editor for story animation
- Animation paths and behaviour automation techniques.

The various aspects of the present invention as described below by reference to the exemplary embodiments may be summarised as follows:

1. **Automatic Cell Detector/Slicer**

Tools to automatically detect and extract cells/clips. Slicing comic pages is the single most tedious task that authors currently have to contend with. The present automation technology works for both rectangular and non-rectangular cells. These tools typically perform a completely automatic initial detection of cells without manual intervention. The results of the initial automatic detection may be subject to manual adjustment prior to extraction of the individual clips.

2. **Slice Wand Tool**
Tools to automatically detect the boundary of an individual cell in response to a manual identification of a point within the cell. The slice wand technology may be used independently but also allows cells that are missed by the automated techniques to be quickly identified and added to a cell list/clip library in an authoring workspace.

3. - Free form and Shape (Polygon) Slice Tool

This shape slicing and free form tool makes cropping irregular shapes easier to handle. The current pick of tools offer slice boxes which cut rectangular shapes only. This prior technique is good for cells that are square or rectangular, however cells that are not require extra work in the form of background masking or subtraction.

4. - Assisted Manual Slicing

This tool facilitates manual slicing of cells, either independently or as part of a hierarchy of automated and/or semi-automated and/or manual slicing methods.

5. - Motion Path Analyser

The capability of identifying cells has been extended to calculate points on the image for creating a linear motion path. This feature adds a huge advantage over other comic authoring tools in the form of cut down work flow. Whereas competitor tools require a laborious manual setup for the animation sequence, the present motion path analyser can batch process all pages or clips and create the animation sequence.

6. - Motion Path Arrows Overlay
This tool assists authors by visually presenting the animation sequence by drawing a series of lines from the start point to the end point. These graphic lines update as motion path points are added, updated, or deleted.

7. - Automated Storyboarding

This tool facilitates the process of assembling collections of clips into complete sequences, either in a fully automated manner or to assist manual editing.

8. - Hierarchical Storyboard/Timeline

This aspect facilitates the editing of large, complex sequences of frames by allowing an author to maintain a better overview of the work as a whole and to navigate more efficiently between parts of the work during the editing process.

9. - Zoom Effects

These effects provide additional editing options in the author workspace to enable more dynamic and visually interesting end products.

10. - Authoring Software Packages and Data Processing Systems

Any one or more of the foregoing aspects of the invention may be incorporated or combined in software packages and data processing systems for comic art authoring. While some of the aspects are particularly applicable to comic art content derived from pre-existing paper-
based material, others are clearly applicable to original, digitally generated content.

The invention in its various aspects is defined in the claims appended hereto.

BRIEF DESCRIPTION OF THE DRAWINGS

Embodiments of the invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

Fig. 1 shows an example of a typical window of a computer user interface, showing cells of comic art displayed and bounded by slice boxes detected by an automated algorithm in accordance with one aspect of the invention;

Fig. 2 illustrates the manner in which the automated detection algorithm may highlight regions that may not have been correctly detected;

Fig. 3 illustrates an example of an authoring workspace as displayed in a typical window of a computer user interface, in accordance with further aspects of the invention;

Fig. 4 shows the authoring workspace of Fig. 3, showing how key frame markers may be indicated on a timeline portion of the workspace;

Fig. 5 illustrates diagrammatically the operation of an automatic cell detection algorithm in accordance with one aspect of the invention;

Fig. 6 illustrates the algorithm of Fig. 5 as applied to non-rectangular cells;
Figs. 7-9 are diagrammatic illustrations of the operation of free form and shape (polygon) slice tool in accordance with a further aspect of the invention;

Fig. 10 illustrates an example of a sequence of cells, similar to Fig. 1, in which motion path reference points have been identified in accordance with a further aspect of the invention;

Fig. 11 is similar to Fig. 10, showing a motion path overlay interconnecting the reference points of Fig. 10;

Fig. 12 is a diagrammatic illustration of an assisted manual slicing process in accordance with a further aspect of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

The basic procedure for authoring a comic file using preferred embodiments of the present invention is as follows:

A. Cell Based Comic

1. Human author scans comic pages or loads pre-digitised image from network or local store
2. Author draws slice boxes on top of comic page cells. Slice boxes are drawn via automated/semi automated tools, with manual intervention as required.
3. The author can reposition and resize the slice boxes to exactly fit each comic cell on the bitmap or to otherwise define the boundaries of the required clips.
4. Author can add extra slice boxes if required (manual and semi-automated); e.g. to sub-divide individual cells.
5. Author extracts the individual clips via an automated process.
6. Author adds clips to a time line and adjusts duration (the time for which each clip will be displayed on playback of the animation).
7. Author can add pan and scale animation (motion paths) for each clip using key frames and tweening.
8. Author adds transitions effects between clips by dragging and dropping transition blocks from a library of transitions.
9. Author adds audio and/or tactile (e.g. vibration) effects (if any) to the presentation.
10. Author publishes the comic animation into a binary file.

B. Whole Page Comic

1. Human author scans comic pages or loads pre-digitised image from network or local store.
2. Author adds pages to a time line and adjusts duration.
3. Author adds motion path points to the comic page using cell region detection.
4. Author can add extra key-frames to animation as well as modify the position and scale properties at each key-frame of the animation.
5. Author adds audio and/or tactile (e.g. vibration) effects (if any) to the presentation.
6. Author publishes the comic animation into a binary file.

In the case of both the cell-based or whole page approach, the end product is a "digital slideshow", comprising a sequence of frames derived from the original comic art work.
In accordance with one preferred embodiment, the present invention provides for a three step process for converting printed comics into digital format for delivery to mobile clients. The three steps are:

1. Capture
2. Storyboard
3. Publish

Step 1 - Capture & Slice

This is the first step and allows comic strips to be loaded if they already exist in digital image format. Authors can also acquire images from a scanner using, for example, an integrated TWAIN library. Each comic page bitmap can be enhanced and cleaned up using filters, colour correction, paint and other basic tools.

This step also handles slicing of the comic pages into separate clips using automated and semi automated techniques. Four preferred options for the slicing process according to aspects of the invention are as follows:

1. Automated cell detection/slicing tool, preferably using a two-pass algorithm with edge detection and cell identification in a scan line pattern. The tool defines perimeters for individual cells in a page and draws "rubber bands" around the identified cell boundaries as slice boxes. These slices can be manually controlled for adjusting size and position. The tool is able to identify rectangular and irregular shaped cells and cut them out. (This type of "rubber band" bounding of selected screen areas/objects for subsequent manipulation is well known in a variety of conventional software applications, such as word processors and graphics editors, and will not be described in detail herein).
2. A manual technique in which an author draws a single slice box of specific shape or free form (ff) shape.

3. A semi-automated tool ("slice wand") that draws an approximate rubber band around a cell when an author clicks anywhere inside the cell boundary.

4. Assisted manual slicing that allows manual slicing of multiple cells on the basis of minimal user input.

The cell detection and slicing tools such as the automatic slicing tool, the shape and free form tool, the slice wand tool and assisted manual slicing are used in combination to quickly extract the clips from a comic page.

These tools offer a new and unique work flow to the tedious task of cropping and slicing allowed by existing tools such as conventional crop and mask tools.

The automatic slicing tool is useful for quickly identifying cells that have clearly defined boundaries. In most comic pages where the background colour is not a texture and cells have clear boundaries, a very high success rate can be achieved in correctly identifying cells. As shown in Fig. 1, the tool identifies rectangular (and non-rectangular) shaped cells, indicated by handles 10. As is well known from a variety of conventional software applications, handles on the corners and sides of a bounding box of this general type may be manipulated, e.g. to re-size the bounding box and/or its content.
Preferably, the automatic detection algorithm may highlight (for example, in red) all regions which it has not been able to deduce as shown in Fig. 2, which shows "missed" cells bounded by a bold line 12.

In cases where cells are missed or not identified, due to lack of borders or complex overlapping with neighbouring cells, the remaining slice tools are useful. The slice wand can be used to identify cells without borders as it analyses bitmap data from a seed pixel and fans out in an expanding grid pattern until an edge of a cell or end of a bitmap is found to create an approximated bounding box on the cell region. Using the freeform or shape slicing tool an author can manually draw accurate slice boxes for missed cells. Assisted manual slicing can be used as an alternative or adjunct to automatic cell detection.

The end result of this first step is a library of clips, which (as described further below - see Automated Storyboarding) may be ordered in a sequence determined by the initial slicing algorithm and any subsequent manual intervention. The user may be able to select sequencing parameters to suit the nature of the source material (e.g. a choice between sequencing on the basis of left-to-right/top to bottom or right-to-left/top-to-bottom).

Step 2 - Storyboard

The use of computer generated storyboards and/or timelines is well known in the fields of video editing, multimedia authoring etc. In this context, a "timeline" is a display showing, for example, thumbnail views of the individual elements of a sequence of clips or frames in which the horizontal dimension of each thumbnail is proportional to the length of time that the corresponding clip or frame will be displayed on playback of the
sequence. Clips can typically be dragged and dropped into the timeline and their display durations can be adjusted by clicking and dragging.

The Storyboard or 'Animate' step involves the use of a computer software application that provides the author a workspace to create the comic presentation by using a timeline on to which comic clips are dropped. As shown in Fig. 3, an exemplary embodiment of a workspace displays a "canvas" 14, including a display window 16 (corresponding to the display window of the target display device for the final published work), a library window 18, and a timeline 20. Software applications of this general type, in which media clips are dragged and dropped onto a timeline, transition effects are applied between clips etc. are well known in applications such as video editing and will not be described in detail herein.

Worthy of note in relation to this application is that the bitmap for a clip corresponding to a clip selected on the timeline is displayed on the canvas 14, with the display window 16 superimposed on the bitmap showing the size and aspect ratio of the target display window relative to the bitmap. Tools provided by the software application allow the position and scale of the bitmap to be adjusted relative to the display window in order to define the desired properties of particular frames of the slideshow presentation represented by the timeline.

The clip library window 18 holds all comic clips sliced in step 1.

An author will typically start by picking a clip for starting off the animation and drop it in the timeline. The first element in the timeline by default starts at time marker 00:00. As content is added a preview of it appears in the canvas. An initial clip sequence may be generated automatically...
based on the sequence determined in the first step and described further below (Automated Storyboarding).

The timeline may be organised into tracks (e.g. a content track, clip properties track, effects tracks etc.), as is also known from video editing applications and the like. As described further below, the timeline may provide for a hierarchical structure of slideshow elements allowing sections of the timeline to be collapsed and expanded to facilitate the authoring process. The timeline is linear and animation starts at time marker 00:00 (min:sec) and increments progressively.

Each clip can be assigned an animation from a preset template of animation behaviours such as pan left to right, top to bottom, etc. Animation paths within a clip may be assigned by dragging animation behaviour from a predefined list of animations.

Motion paths in a clip are created by adding key-frames (as illustrated in Fig. 4 which shows key frame markers 22 in timeline 20) in a clip block and adjusting properties of the bitmap object. These properties are typically scale, position, and orientation. Properties of an object are adjusted via a clip properties window.

The final step is Publish where the comic is exported to a binary file format.

Step 3 - Publish

This is the final step and lets the author export the comic in a format suitable for viewing/playback on the target device.
The various aspects of the present invention further include a number of technology tools and techniques for use in comic authoring software.

**Automatic Cell Detector/Slicer**

A first aspect of the invention provides a cell boundary detection tool that performs a feature extraction process to identify the edges (boundaries) of cells in a bitmap, typically using a sequence of image processing algorithms.

In one exemplary embodiment, the algorithm performs a number of processing tasks. It starts, preferably, by normalising the image colours and/or brightness and/or contrast to a reasonable range. It then uses an edge detection algorithm to identify edges in the content of the bitmap, thereby defining a list of edges, followed by an edge testing algorithm that processes the detected edges to locate closed paths that are identified as cell boundaries.

Edge detection algorithms are well known in the field of digital image processing. Edge detection processes generally operate to mark the points in a digital image at which the luminous intensity changes sharply. Most edge detection processes are either search-based or zero-crossing based. Examples of edge detection methods include Sobel, Roberts Cross, Prewitt, Canny, Rothwell and Marr-Hidreth. For the purposes of the present invention, a preferred approach to edge detection is a combination of the Sobel and Prewitt techniques, using a larger convolution matrix (about 4X4 matrix) with rotatable coefficients of the operator, making it sensitive to detection of different edge orientations.
The list of edges is then processed by a subsequent edge testing algorithm to identify those edges that correspond to cell boundaries. The edge testing algorithm is preferably a scan line algorithm. The scan line algorithm starts at the first (start) pixel of an edge and follows through iteratively checking neighbouring pixels to test for closed corners or contiguous paths until the start pixel is reached.

It then moves on iteratively to the next edge in its list performing the same test to identify the cell boundary shape until no more edges are left.

Some comic art is designed to be read from left to right, and others from right to left. The algorithm below assumes it will be from left to right, but can easily be reflected for other directions.

A preferred algorithm for identifying cells is as follows:

1. Normalise Image (e.g. brightness, contrast, colour)
2. Execute (Sobel) edge detection to highlight all outer edges on image
3. Create an Edge List from edges found in step 2
4. Create an empty Cells List
5. Create an empty Cell Points List
6. Start scan line analysis (Read order = LR)
   i) Begin at first (start) on first edge in list. Check neighbouring pixel using, for example, a 3X3, 5X5 or 9X9 mask operator with suitable tolerance levels, e.g. +-3%.
   ii) (R) Test if neighbouring pixel is continuation of boundary.
   iii) (S) Test if neighbouring pixel is corner
   iv) (T) Test if neighbouring pixel is end of bitmap
   v) (U) Test if neighbouring pixel is equal to start pixel
   vi):
If R = true then
   Add (x,y) to Cell Points List
   /* depending on neighbour pixel tested */
   Move to (x+1 , y ) OR
   Move to(x, y+1) OR
   Move to(x+1 , y+1 ) OR
   Move to(x-1 , y+1)
End if

vii):
If S = true then
   Add (x,y) to Cell Points List
   /* depending on neighbour pixel tested */
   Move to (x, y+1) OR
   Move to(x+1 , y+1 ) OR
   Move to(x-1 , y+1)
End if

viii):
If T = true then
   Add (x,y) to Cell Points List
   If right edge
      Move to (x, y+1)
   End if
   If bottom edge
      Move to (x-1 , y)
   End if
If Left Edge
    Move to (x, y-1 )
End if

If Top edge
    Move to (x+1, y)
End if

End if

ix):
    If U = true then
        Add (x,y) to Cell Points List

Create shape from cell points
Calculate region
Add Cell Data to Cells List
Increment Cells List
Lock region

End if

x) Exclude all edges in cell region discovered
xi) Move on to next edge in list until no more edges left

End

The detection algorithm typically takes in parameters such as image source, and reading order (i.e. left to right/right to left). It then identifies all external most edges and indexes them into a list.
Fig. 5 illustrates diagrammatically how the edge testing algorithm operates. It starts by scanning the bitmap data from top to bottom, left to right until it reaches the first edge in the list (Point A). It then follows through testing for conditions R, S, T, and U. Corner points are denoted by Points B, C, and D in Fig. 5. Point S in Fig. 5 represents condition U being met which results in the cell shape being closed.

It is worth noting that the algorithm handles cell boundaries that are interrupted by overflow speech bubbles or text overlays. The algorithm incorporates such interferences by including the boundaries of these overlay objects and adding the points to the list for the current cell (Fig. 5, zoomed illustration). Fig. 6 illustrates the algorithm applied to non-rectangular cells.

**Slice Wand Tool**

In accordance with a further aspect of the invention, this tool requires user input in the form of a mouse click or the like to capture the co-ordinate of a pixel inside a particular cell and then searches outwards from that pixel to find the outer boundary of the cell. The algorithm, in a preferred exemplary embodiment, performs a boundary flood fill computation testing all neighbouring pixels in an expanding grid pattern, from the seed pixel, until an edge is reached or end of bitmap is found. It then marks the edges and defines a shape for the cell region. The slice shape can be regular or irregular with moveable control points and bounding box with handles to proportionately scale the slice shape.

The preferred boundary flood fill algorithm detects boundaries on the basis of colour and/or contrast differentiation by testing pixels for abrupt
changes in value, maintaining a list of pixels that are candidates for being boundary pixels until a closed path is detected.

The content of a cell may include a closed boundary that could be mistaken by the algorithm for the outer boundary of the cell itself. To deal with this, a minimum cell area may be defined so that the algorithm ignores any closed boundary enclosing less than a predetermined minimum cell area. In difficult cases, the automatic algorithm may be over-ridden manually.

An alternative to the boundary flood fill approach is for the user to select a seed pixel on the boundary and to perform a "path fill" algorithm, seeking neighbouring pixels of similar value to the seed pixel until a closed path is defined.

**Free form and Shape (Polygon) Slice Tool**

In accordance with a further aspect of the invention, this tool allows polygon shapes to be drawn by marking points on a bitmap and creating edges between points to achieve a closed shape (as illustrated in Figs. 7, 8 and 9). The shape can be any size and pattern and cuts out an exact shape of the underlying bitmap. The control points can be moved to change the shape of the slice polygon or resized using the bounding box handles.

**Assisted Manual Slicing**

A further aspect of the invention provides a method for manually slicing and sequencing cells with computer assistance to minimise user input.
An exemplary embodiment is illustrated in Fig. 12. Using a mouse or other pointing device, the user marks a series of points (such as pairs \([1,2], [3,4], [5,6]\)) in the borders between adjacent cells to define a minimal set of slice lines that divide all of the cells A-D on one page from one another (a pair of points may be the end points of a line traced by the user).

The order in which the points are marked may determine, at least in part, the sequence in which the individual cells are to be presented. As each area is enclosed by the lines between points, cell data including a cell shape and a cell region is generated and added to the list of cells.

The points marked by the user need not be located at the end points of the corresponding slice, although they should preferably be close to the end points. Each pair of points defines a single straight slice line. The algorithm that interprets the user input will extend the line passing through each pair of points in each direction until it either intersects another slice line defined by another pair of points or reaches the edge of the page (X). The method is analogous to the manner in which a physical page of a comic might be manually cut into individual cells using a pair of scissors, but minimises the user input required to define all of the required cuts. In the case of complex cell boundary shapes, more than two points may be required to define a single continuous slice. User input may distinguish between end points and intermediate points of a particular line. Here, "end points" includes points from which the line will be extended outwardly to an intersectionbitmap edge, rather than, necessarily, the physical end points of the slice line. The intermediate points indicate the shape of the line between the end points. The positions of said points may be adjusted manually to alter the direction and/or length and/or shape of said slice lines deduced by the algorithm.
The algorithm used to interpret the user input and execute the slices can have knowledge of cell sequencing logic etc., allowing it to adopt a heuristic approach when interpreting the user input.

5 Motion Path Analyser

In accordance with a further aspect of the invention, this technique determines motion path reference points on a comic page bitmap needed to create a motion path for displaying a large image bitmap on a limited screen size device, by panning/scrolling a viewing window over the image as previously mentioned. For present purposes, the bitmap may be a complete page of comic art work or a clip comprising individual cell, a group of cells etc.

10 This technique preferably uses a subset of the cell detection algorithm to create bounding boxes (cell boundaries) for each cell in the bitmap. The cell boundaries may alternatively be determined by any of the other techniques described herein. The dimensions of the bounding boxes are then used to calculate motion path reference points in every cell by reference to the screen size and aspect ratio of a target viewing/display window (e.g. the screen of a target device).

20 The motion path analyser technique uses a method for returning the reference points 24 (Fig. 10) in a cell, a group of cells or a whole page by taking in parameters such as reading order, viewing screen aspect ratio, and bitmap image source.

25 In general, the points 24 (Figure 10) represent the centres of key-frames (i.e. the centres of the viewing window position at the beginnings and ends of discrete movements that together constitute a motion path), and may be
the actual centres of individual cells, or centres of attention within individual cells. The motion path will move the viewing window between these points in straight lines or curves.

Depending on the size and aspect ratio of a cell relative to the viewing window, a single reference point may be centred within that cell and, if required, the cell may be scaled to match the viewing window. If the size and/or aspect ratio of the cell is substantially different from that of the viewing window, multiple reference points may be defined within the cell to define a motion path that pans/scrolls the viewing window over the entire area of the cell. In a simple example, reference points may be selected such that the viewing window begins in the top left corner of the cell, pans to the top right corner, scrolls diagonally to the bottom left corner then pans to the bottom right corner. If the vertical or horizontal dimension of the cell is similar to that of the viewing window, the cell may be scaled in one direction so that only a horizontal scan or a vertical scroll is required to cover the cell the area.

Panning/scrolling may progress continuously between reference points or may pause or slow down at the key frames defined by the reference points.

**Motion Path Arrows Overlay**

In accordance with a further aspect of the invention, this technique is advantageous from a User Interface perspective. As shown in Fig. 11, it draws graphical lines using the reference points determined by the Motion Path Analyser or set by the author to show the motion path of the viewing window over the bitmap. This assists the author in visualising and, if necessary, adjusting the positions of key frames relative to the
underlying bitmap and/or adding or deleting key frames so as to alter the motion path. This can be done by adjusting the positions of the reference points and/or adding/deleting reference points in a display such as that shown in Fig. 11. Alternatively, similar changes can be made within the authoring environment illustrated in Fig. 4, by adjusting the position of the bitmap relative to the viewing window for particular key frames and/or by adding or deleting key frames to or from the timeline and setting properties of the image (i.e. position, scale, etc) in the scene at that key-frame.

Motion paths are generally linear (although they may also follow a curve approaching the points 24) and follow a set trail from point to point in the order of the points.

**Automated Storyboarding**

A further aspect of the invention provides a tool for automatically generating at least a "first draft" storyboard/timeline, for example by dragging a folder of individual cells onto the timeline in the authoring workspace. This assumes that the sequencing information is included with the collection of cells (e.g. in their filenames, in their data content, in the order in which they are identified by the user, or in associated indexing data). A first draft sequence may be defined automatically or semi-automatically during the process of automatic/semi-automatic cell detection as described above, by applying any suitable indexing scheme to the cells.

Varying degrees of automation may be applied: e.g. slideshow parameters such as cell/frame display durations, motion paths, transition effects between cells/frames etc. may be determined based an automated
analysis of individual cell content. By extension, this process could automatically generate and publish a binary version of the presentation.

In automatically determining the duration of the motion paths between motion path reference points separated by varying distances, the duration may be calculated to hold the perceived speed of motion constant. The perceived speed of motion may be measured as the speed of pixels moving across the viewing screen, combined with the rate of zooming. For some comic art, this might be a panning rate of one whole screen width or height, or a zoom adjustment of doubling or halving the image size in a predetermined elapsed time interval such as one elapsed second.

Hierarchical Storyboard/Timeline

A further aspect of the invention relates to the presentation of a storyboard or timeline in the author workspace. By way of example, consider a sequence comprising multiple clips, in which some clips are to be displayed as single frames and others are to be displayed as sub-sequences of multiple frames showing different parts of the clip content. A conventional storyboard or timeline would show each individual frame. For complex sequences, consisting of hundreds or thousands of frames, it may be difficult for an author to maintain an overview of the complete work or simply to navigate to a particular frame of interest. In accordance with this aspect of the invention, slideshow elements may have a hierarchical structure so that a storyboard or timeline can be collapsed or expanded to display different levels of detail of the overall work represented by the timeline.

The storyboard may also be designed to include properties and adjustments relating to each frame, such that the user might see and
manipulate the speed, duration, delay, visual effects and other characteristics. Such properties can be considered as a lower level in the hierarchy of the storyboard.

In a simple exemplary case, sub-sequences of frames within an individual clip could be collapsed into a single clip element in a timeline, or expanded to show the individual frames of the sub-sequence. When collapsed, the horizontal dimension of the single element would be proportional to the duration of the complete sub-sequence. When expanded, the timeline would reflect the durations of the individual frames. This approach may be extended through multiple levels, such as "pages", "scenes", "acts", "chapters" etc. User interface controls may be provided to allow individual sections of the complete work to be collapsed/expanded and to allow the complete work to be collapsed/expanded to any chosen level of the hierarchy.

That is, slideshow elements of the timeline may be grouped in sets that can be collapsed and expanded on the timeline display, and such sets may be further subdivided into sub-sets that can be collapsed and expanded. When a set or sub-set is collapsed, a horizontal timeline dimension of the timeline representation of the set or sub-set may be proportional to the duration of the complete set or sub-set and, when expanded, the horizontal timeline dimensions of the elements of the set or sub-set may be proportional to the durations of corresponding individual slideshow elements.

**Zoom Effects**

A further aspect of the invention relates to the application of visual zoom effects to individual cells, clips or frames. For example, a large cell might
initially be scaled to fit the target display window and the display window
may then zoom into a first area before applying a motion path to display
the complete content of the cell. The display could zoom out again at the
end of the cell and/or multiple zooms in and out could be incorporated into
the motion path. The application of zoom effects could also be automated
or semi-automated based on automated analysis of the cell content. For
example, a whole cell might be shown zoomed out first, and then it might
zoom in on the centres of interest identified by the motion path analysis.
This aspect of the invention extends to published digital comic art content
incorporating such zoom effects.

Techniques for applying visual zoom effects to digital images are well
known in themselves and will not be described in detail herein. However,
the application of such effects in the context of animated presentations of
sequential comic art as described herein have not hitherto been known in
the art.

It will be understood that aspects of the present invention may be
embodied in one or more computer program products, which may be
encoded on any suitable data carrier, and in suitably programmed data
processing (computer) systems. Further aspects may be embodied in
published digital comic art products (works).

Published works created using the present invention may be published in
any format suited for display/playback on particular target devices, such
formats including, but not being limited to proprietary formats (e.g. for
viewing/playback by means of a corresponding viewer, such as a Java
applet) and known multimedia file formats such as Shockwave, Flash,
MPEG, WMV, AVI, MOV etc.
The invention in its various aspects is applicable in the industrial production of digital slideshow presentations of comic art.

Improvements and modifications may be incorporated without departing from the scope of the invention.
Claims

1. A computerised method of processing a bitmap representing at least a portion of a comic art work, said comic art work comprising a plurality of cells, the method comprising automatically detecting boundaries of cells within said bitmap whereby said cells may be identified for use in a digital slideshow presentation of said comic art work.

2. The method of claim 1, wherein automatically detecting said cell boundaries includes applying an edge detection algorithm to said bitmap to identify edges in the content of the bitmap.

3. The method of claim 2, wherein said edge detection algorithm comprises a combination Sobel/Prewitt convolution method.

4. The method of claim 2 or claim 3, wherein automatically detecting said cell boundaries includes applying an edge testing algorithm to said edges to identify closed paths corresponding to cell boundaries.

5. The method of claim 4 wherein said edge testing algorithm comprises a scan line algorithm that scans the bitmap data until it reaches an edge identified by the edge detection algorithm.

6. The method of claim 4 or claim 5, wherein said edge testing algorithm begins at a start pixel of a first edge detected by said edge detection algorithm and iteratively checks neighbouring pixels to determine if a neighbouring pixel is a continuation of a boundary, a corner, an end of the bitmap or is equal to the start pixel.
7. The method of claim 6, wherein operation of the edge testing algorithm adds pixels that are a continuation of a boundary, a corner, or an end of the bitmap to a cell points list and, when the start pixel is reached, creates cell data including a cell shape and a cell region from the cell points and adds the cell data to a cells list.

8. The method of claim 7, wherein unprocessed edges included in the cell region are excluded from subsequent edge testing.

9. The method of any preceding claim, further including the step of displaying said bitmap with the detected cell boundaries superimposed thereon, to enable manual alteration of the detected boundaries.

10. The method of any preceding claim, further including the step of storing, as at least a part of a library of clips, individual regions of the bitmap bounded by said boundaries.

11. The method of any preceding claim, further including the step of storing the bitmap together with data defining the cell boundaries.

12. The method of any preceding claim, including an initial step of normalising said bitmap prior to automatically detecting the cell boundaries.

13. The method of claim 12, wherein normalising said bitmap includes normalising at least one of the colour, brightness and contrast of the bitmap.

14. A computerised method of processing a bitmap representing at least a portion of a comic art work, said comic art work comprising a
plurality of cells, in order to identify boundaries of cells within said bitmap, whereby said cells may be identified for use in a digital slideshow presentation of said comic art work, the method comprising manually indicating a first pixel that is either on or inside the boundary of a cell and automatically searching outwards from said first pixel to detect the outer boundary of the cell.

15. The method of claim 14, wherein automatically searching outwards from said first pixel comprises performing a boundary flood fill computation, when the first pixel is inside the boundary, or a path fill computation, when the first pixel is on the boundary.

16. The method of claim 15, wherein performing a boundary flood fill computation comprises testing all neighbouring pixels in an expanding grid pattern until an edge or an end of the bitmap is reached.

17. The method of any one of claims 14 to 16, wherein a closed boundary enclosing less than a predetermined minimum area is ignored.

18. The method of any one of claims 14 to 17, wherein automatic cell boundary detection can be over-ridden manually.

19. The method of any one of claims 14 to 18, further including the step of displaying said bitmap with the detected cell boundaries superimposed thereon, to enable manual alteration of the detected boundaries.

20. The method of any one of claims 14 to 19, further including the step of storing, as at least a part of a library of clips, individual regions of the bitmap bounded by said boundaries.
21. The method of any one of claims 14 to 20, further including the step of storing the bitmap together with data defining the cell boundaries.

22. A computerised method of processing a bitmap representing at least a portion of a comic art work, said comic art work comprising a plurality of cells, in order to identify boundaries of cells within said bitmap, whereby said cells may be identified for use in a digital slideshow presentation of said comic art work, the method comprising manually indicating a plurality of control points on said bitmap and automatically connecting adjacent pairs of said control points to create a closed polygon that identifies a boundary around at least one of said cells.

23. The method of claim 22, further including manually adjusting the positions of said control points to alter the shape of said polygon.

24. The method of claim 22 or 23, further including the step of storing, as at least a part of a library of clips, individual regions of the bitmap bounded by said boundaries.

25. The method of claim 22 or 23, further including the step of storing the bitmap together with data defining said boundaries.

26. A computerised method of processing a bitmap representing at least a portion of a comic art work, said comic art work comprising a plurality of cells, in order to identify boundaries of cells within said bitmap, whereby said cells may be identified for use in a digital slideshow presentation of said comic art work, the method comprising manually marking points in border regions between adjacent cells so as to define a minimal set of slice lines that divide the cells in the bitmap from one another.
27. The method of claim 26, wherein the order in which said points are marked determines, at least in part, a sequential order of said cells.

28. The method of claim 26 or claim 27, further including, when an area of said bitmap is enclosed by a set of said slice lines, generating cell data including a cell shape and a cell region from the cell points and adding the cell data to a cells list.

29. The method of any one of claims 26 to 28, wherein a slice line defined by two or more of said points is extended in opposite directions until it intersects another slice line or reaches the edge of the bitmap.

30. The method of any one of claims 26 to 29, wherein a single slice line is defined by at least two points.

31. The method of any one of claims 26 to 30, wherein manually marked points are identified by user input as end points or intermediate points of a slice line.

32. The method of any one of claims 26 to 31, wherein the positions of said points may be adjusted manually to alter the direction and/or length and/or shape of said slice lines.

33. The method of any one of claims 26 to 32, further including the step of storing, as at least a part of a library of clips, individual regions of the bitmap bounded by said boundaries.

34. The method of any one of claims 26 to 32, further including the step of storing the bitmap together with data defining the cell boundaries.
35. A computerised method of processing bitmaps representing comic art work, said comic art work comprising a plurality of cells, in order to identify boundaries of cells within said bitmap, whereby said cells may be identified for use in a digital slideshow presentation of said comic art work, comprising a method as defined in any one of claims 1 to 13 or 26 to 34 in combination with a method as defined in any one of claims 14 to 25.

36. A computerised method of processing bitmaps representing comic art work, said comic art work comprising a plurality of cells, for use in a digital slideshow presentation of said comic art work, so as to define at least one motion path along which a viewing window is to be panned and/or scrolled across at least a part of said bitmap, comprising automatically determining a plurality of motion path reference points that define at least the beginning and end of at least one motion path, said reference points being determined automatically by reference to the boundaries of at least one cell of said comic art work and to the size and aspect ratio of a target viewing window.

37. The method of claim 36, wherein said reference points are determined for a motion path relating to a single cell.

38. The method of claim 36, wherein said reference points are determined for a motion path relating to a plurality of cells, taking account of an intended reading order of said cells.

39. The method of any one of claims 36 to 38, wherein said reference points correspond to the centres of key-frames.
40. The method of claim 39, wherein said reference points correspond to actual centres of individual cells, and/or centres of attention within individual cells.

41. The method of any one of claims 36 to 40, wherein paths between reference points comprise straight lines and/or curves.

42. The method of any one of claims 36 to 41, further comprising manually adjusting the position of at least one key frame corresponding to at least one of said automatically determined reference points, relative to said bitmap.

43. The method of any one of claims 36 to 42, wherein cell boundaries within said bitmap are determined, for the purpose of determining said reference points, by a method in accordance with any one of claims 1 to 34.

44. A computerised method of processing bitmaps representing comic art work, said comic art work comprising a plurality of cells, for use in a digital slideshow presentation of said comic art work, in defining at least one motion path along which a viewing window is to be panned and/or scrolled across at least a part of a bitmap, comprising determining a plurality of motion reference points that define at least the beginning and end of at least one motion path, and displaying the bitmap with a visual representation of the motion path defined by said reference points superimposed thereon.

45. The method of claim 44, further comprising manually adjusting the position of at least one key frame corresponding to at least one of said automatically determined reference points, and/or adding or deleting key
frames to or from the motion path, by manipulating said visual representation of the motion path in the display of the motion path or by modifying, adding or deleting corresponding key frames and key frame properties in a timeline based authoring environment.

46. The method of claim 44 or claim 45, wherein said reference points are determined by the method of any one of claims 36 to 43.

47. A computerised method of processing digital representations of comic art work, said comic art work comprising a plurality of cells, for producing a digital slideshow presentation of said comic art work, comprising generating a library of clips extracted from said digital representations and representing cells of said art work, said library including information defining a viewing sequence of said cells, and sequencing frames of said slideshow on the basis of said viewing sequence information.

48. The method of claim 47, further including determining at least one slideshow parameter comprising at least one of cell/frame display durations, motion paths and transition effects between cells/frames, on the basis of automated analysis of individual cell content.

49. The method of claim 48, including automatically determining the duration of motion paths between motion path reference points separated by varying distances, said durations being calculated to hold the perceived speed of motion constant.

50. The method of claim 48 or claim 49, further including automatically generating a binary version of a complete slideshow presentation.
51. A computerised method of processing digital representations of comic art work, said comic art work comprising a plurality of cells, for producing a digital slideshow presentation of said comic art work, comprising generating a library of clips extracted from said digital representations and representing cells of said art work, and generating a display representing a sequence of slideshow elements, including selected ones of said clips and/or frames derived from said clips, as a timeline, said slideshow elements having a hierarchical structure whereby sections of the timeline may be collapsed or expanded in the timeline display to show differing levels of detail of the overall timeline.

52. The method of claim 51, wherein slideshow elements of the timeline are grouped in sets that can be collapsed and expanded on the timeline display.

53. The method of claim 52, wherein said sets are subdivided into sub-sets that can be collapsed and expanded on the timeline display.

54. The method of claim 52 or 53, wherein, when a set or sub-set is collapsed, a horizontal timeline dimension of the timeline representation of the set or sub-set is proportional to the duration of the complete set or sub-set and, when expanded, the horizontal timeline dimensions of the elements of the set or sub-set are proportional to the durations of corresponding individual slideshow elements.

55. The method of any one of claims 52 to 54, wherein a set or sub-set of slideshow elements comprises a sub-sequence of frames within an individual clip.
56. A computerised method of processing digital representations of comic art work, said comic art work comprising a plurality of cells, for producing a digital slideshow presentation of said comic art work, comprising generating a library of clips extracted from said digital representations and representing cells of said art work, defining a viewing sequence in which said clips are to be viewed, and including defining visual zoom effects to be applied to at least selected ones of said clips during viewing of said slideshow presentation.

57. The method of claim 56, wherein said visual zoom effects include at least one of scaling a cell to fit a target display window, zooming into a first area of a cell before applying a motion path to display the complete content of the cell, zooming out at the end of a motion path, and multiple zooms in and out incorporated into a motion path.

58. The method of claim 56 or 57, wherein the application of zoom effects is automated or semi-automated based on automated analysis of cell content.

59. The method of claim 58, wherein said zoom effects are applied automatically or semi-automatically based on motion path analysis in accordance with the method of any one of claims 36 to 43.

60. A digital slideshow presentation of comic art work, incorporating visual zoom effects applied to elements of said slideshow presentation in accordance with the method of any one of claims 56 to 59.

61. A computer program product comprising executable program code for use in generating digital comic art slideshows using a method according to any one of claims 1 - 59.
62. A computer program product as claimed in claim 61, comprising a computer usable medium having said executable program code embodied in said medium as computer readable program code means.

63. A data processing system programmed for use in generating digital comic art slideshows using a method according to any one of claims 1-61.
Left to Right
Top to Bottom

BITMAP IMAGE EDGE

MASK

CORNER TEST

Fig. 5
Fig. 6

Fig. 7

\(\square = \text{Handler}\)