THE JOURNAL OF VISUALIZATION AND COMPUTER ANIMATION, VOL. **8**: 165–185 (1997)

Automatic Cel Painting in Computer-assisted Cartoon Production using Similarity Recognition

CHUEH-WEI CHANG AND SUH-YIN LEE
Institute of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C. (email: cwchang@info1.csie.nctu.edu.tw | sylee@csie.nctu.edu.tw)

SUMMARY

The task of painting each cel of an animated sequence has always been a tedious work using traditional methods. In order to simplify and speed up the process, the development of an automatic cel painting system is desired. This paper presents a method for assisting the production of cartoon animation using computers in order to reduce the production expense and to improve the overall quality. It signals a great improvement in the working environment where 2-D animation is produced. The basic idea of our automatic cel painting mechanism is to extract both statistical and topological features of regions from input animation drawings, and to search for similar corresponding regions in the previous drawing. If the similarity measures are within the accepted thresholds, the corresponding colour will be inherited. A new integrated labelling algorithm constructing statistical region and topological graph features is proposed. Furthermore, a similarity measure of regions is defined and an algorithm of partial matching of attribute graphs is presented. The experimental result shows the feasibility and practicability of the proposed approach. © **1997 by John Wiley & Sons, Ltd.**

KEY WORDS: cartoon animation; similarity measure; recursive labelling; graph matching

1. INTRODUCTION

Animation is one of the commonly used media which can communicate information via images.[1] It refers to the process of dynamically generating a series of frames with a set of objects, and each frame is an alteration of the previous frame. Traditional cartoon animation is a labour-intensive job. In spite of the fact that artists are needed to create high quality animation, many of the steps that traditional animation involve are repetitive and require few special skills.

Computer animation is often carried out using a similar principle to traditional animation. When computer graphics technology matures for picture generation and processing, computer animation will become one of the natural applications.[2–4] Therefore, the idea of a computer-assisted cartoon animation system is becoming an important research issue and has been studied by many researchers.[5,6] The applications of cartoon animation have expanded from such areas as entertainment, advertisement,

CCC 1049–8907/97/030165–21

Received August 1995
Revised June 1996

presentation and art to the multimedia fields. On the theatrical film level, high-resolution, 2-D computer-animated special effects are no longer unusual.[7,8] Computer-assisted cartoon animation is becoming more popular for two reasons.[9] One is the progress of computer technology which makes it possible to obtain computer generated real-time animation. The other is that low-cost graphical environments have become available and many of them are affordable by production houses. Today's computer animation systems mimic not only the animator's tools, but those of the director, layout artist, inker, painter, background artist, and others in a traditional animation studio.[10]

In contrast to 3-D or 2-D animation with models,[8] several experiments have been done in image composition by combining levels of artwork into a final composite frame;[11,12] in the resolution-independent rational zoom of bitmaps using B-spline interpolation;[13] and in bitmap transformations to achieve smoothness in the animation and prevent object distortion.[14] Although modern digitizing tablets are even coated to make the friction of the stylus simulate the drag of a pencil on paper, there are still many mechanical limitations on the accuracy with which one can draw using currently available input devices. Therefore, free-hand drawing is still the favourite method used in the computer animation production processes.

In order to speed up the production processes, the computer can play an important role in relieving the tedium and expense of hand colouring. But cartoon animation has many complicated deformation characteristics. Achieving automatic cel painting using motion track detection between drawings is very difficult. Only in a few instances, with simple animation sequences, can motion paths be tracked easily using template matching.

Traditionally, animation drawings contain components such as regions, arc segments and points. To make the matching processes more efficient and easy, these components should be transformed into image representations other than a pixel image. Basically, there are two major approaches for image representation, namely the parametric approach and the syntactic approach. In the parametric approach, objects are represented by features, such as colour and size, which can be easily calculated from images. On the other hand, in the syntactic representation, image features are represented as sets of symbolic entities, or in a notation of image primitives.

A more powerful approach for image representation has been emanating from the attribute graph approach by combining the parametric and syntactic approaches. The image features are represented by attribute entities, where the attributes represent some semantic parameters of the spatial features. Moreover, the semantic information of the relationships among the image features is represented by the attribute graph associated with the relations between their corresponding entities. Owing to the powerful capability exhibited by graphs as tools for representation and analysis, attribute graphs are adopted in our research. According to these representations, similarity measurement is provided to find the corresponding region between related drawings. Similarity recognition is considered to be a task of finding the best fit region in these drawings using attribute graph representations. The problem of calculating graph-to-graph distance has to be solved. It is necessary to define a similarity measure between two regions of drawings.

In this paper, we propose a mechanism for automatic cel painting in a computer-assisted cartoon production system. In Section 2, the basic steps of computer-assisted production and the automatic painting procedure are presented. Section 3 describes

several drawing preprocessings which must be done before the extraction of the drawing features. In Section 4, the method of obtaining statistical features of regions and the extraction of topological information, called the integrated labelling algorithm, is described in detail. In Section 5, we define the similarity measurements for region features and graph features. Furthermore, a graph mapping algorithm is also stated. In Section 6, an illustrating example and the experimental results are discussed. In Section 7, we give the conclusions.

2. COMPUTER-ASSISTED PRODUCTION

The production of cartoon animation is a complicated task.[15–17] The sequence of steps required to create a typical animated film is long and highly complicated, and each studio operates slightly differently. There are numerous steps in the traditional animation process that can be assisted or enhanced by computer graphics.[18] The problem in designing a large-scale cartoon animation system is not only that of solving individual algorithmic problems, but rather selecting some combination of techniques that will provide a wide range of capabilities at a reasonable cost.

A large animation studio may have in different parts of its production pipeline tens of thousands of drawings at one time. A brief description of computer-assisted techniques for the cartoon animation process, as shown in Figure 1, is as follows:

1. *Scanning*. Optical scanning is the best way for input of background and free-hand drawing images into a computer as a two dimensional array of pixels (raster image).
2. *Cel painting (colouring)*. The computer replaces cel colouring as painting each enclosed area in each drawing by using an algorithm called an area flooder (or filler).
3. *Image composition*. Artwork arrives at the final production step in digital form. The digital representations of each layer are transformed and merged together,[11] generating final frames that consist of raster images. These images can then be recorded, either on film or videotape, for editing in the traditional manner.

Since animation is the expression of movement, we can obtain a time series of drawings. These drawings are ordinarily separated on a scene-by-scene basis. There have been two working models for automatic cel painting. One is the batch model.

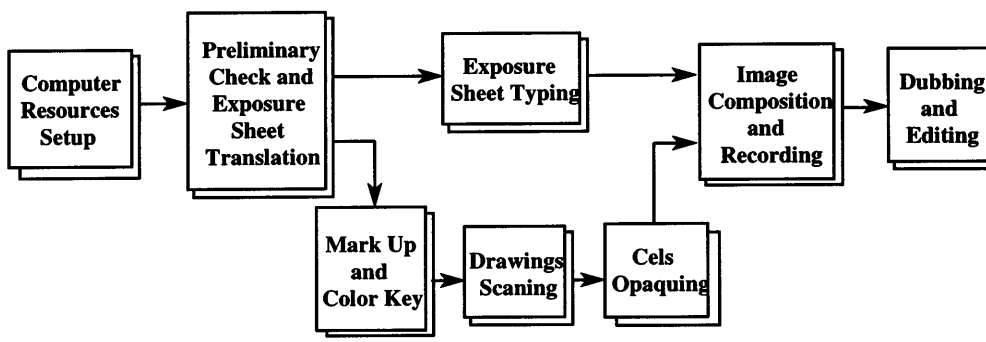

Figure 1. Computer-assisted production flow

This model requires the user to paint several key frames manually first, and the computer calculates the remaining in-between drawings using an automatic interpolation method. The other is the iterative model which uses a one-by-one forward prediction approach from the first drawing to the last one. We use the latter model in our system.

The basic idea of our iterative automatic cel painting mechanism is to extract region features from the input animation drawing, and to search for similar corresponding regions in the previous reference drawing, as shown in Figure 2. If all the similarity measures are within the accepted thresholds, then the corresponding colour will be inherited. The iterative handling processes for automatic cel painting is summarized as: (1) opaque the first drawing manually; (2) obtain the next drawing and record all of the region feature information related to the drawings; (3) perform image preprocessing for the new drawing to obtain a better drawing feature quality; (4) establish region features and graph structures using an integrated labelling algorithm; (5) make the matching and similarity measurement to find the colour of the target region from the previous corresponding region, and fill colours into this target region; (6) repeat steps (2)–(5), until all drawings in the current scene have been painted.

Madeira *et al.*[15] propose a similar computer-assisted painting system, but, using different region matching and colour assignment mechanisms. Their approach provides a good data structure for image compression and the generation of in-betweening. But, this approach also causes a complicated string comparison and region assignment procedure.

3. PREPROCESSING OF DRAWING IMAGES

Animation drawings contain components such as regions, arc segments and points. Several processes must be done before making the image transformations, feature extraction and similarity measurements.[15] We use the definition of *4-connectivity*[19] where two pixels are assumed to be connected if they are connected via a chain of horizontally or vertically adjacent pixels. That is, a pixel P at co-ordinates (x, y) has four horizontal and vertical neighbours whose co-ordinates are given by $(x+1, y)$, $(x-1, y)$, $(x, y+1)$, $(x, y-1)$, called pixel *Right*, *Left*, *Down*, and *Up*, respectively.

A *region* in a raster scan bitmap image is a closed area with different colours at the boundary. Every pixel in the region is 4-connected. In order to obtain region features of each drawing, it is necessary to extract regions from an original drawing first of all. Generally, regions in cartoon animation drawings are drawn black with

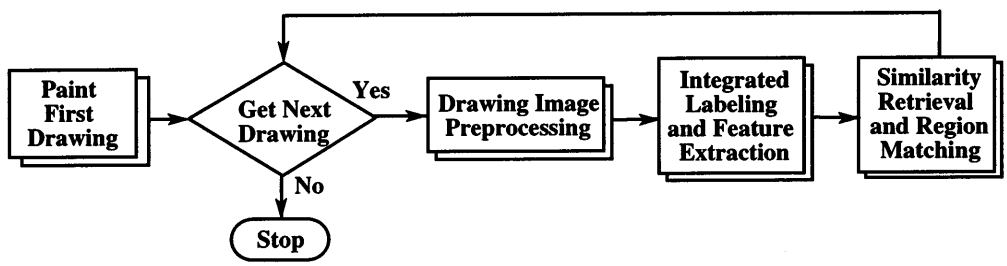

Figure 2. The iterative automatic cels painting model

pencil, and the background of the drawings is usually white. In each drawing, there is a sketch of the characters. The gray-level *histogram* has a characteristic form, such as having two steep peaks, because of the difference of brightness between characters and background. That is, we have already got very strong edge information in each drawing. Despite that, drawing preprocessing is essential because the line quality of an input drawing is often degraded by sketching or scanning. Therefore, in order to obtain more precise information about a drawing, some image enhancement is needed. These processes can be done automatically in our system.

The first step of drawing region recognition is to analyse the shape of the input drawing, so we have to preset the drawing to a binary image for further processing. In bilevel *thresholding*,[19] the brightness value of each pixel is compared with a threshold value, and the pixel is assigned to one of two categories depending on whether the threshold is exceeded or not. A threshold can be chosen from the values between the two peaks in the histogram and the region can be defined as the set of pixels whose value is below the threshold. The objective is to locate the bottom of the valley so that the object can be segmented from the background. After this process, the border shape of each region in the drawings will be clear and sound.

If the original drawings are too noisy, the black regions may contain scattered white points, and the white regions scattered black points. This condition is known as salt-and-pepper *noise*.[19] Noise in the drawings will influence the correctness of feature extraction. Therefore, noise removal is needed for improving the quality of the input image. Two kinds of noise must be removed before labelling. They are small regions and small dots. The way we check for isolated noise points is to run a mask over the image. The size of mask depends on the size of noise the user wishes to delete from the drawing. For small regions, we refill them with black edge colour to reduce the region number in the matching process for this drawing.

A *labelling*[19] process decomposes the drawing into a collection of regions. Each region is given as a closed connected area that belongs to the same colour. We assign each region a unique identifier, so these regions can be recognized in the following similarity measurement processes. That is, each connected region component receives a unique identity code and every image pixel is labelled with the identity code of the region component to which it belongs. A labelling processing can be implemented by an area fill algorithm with some assisted region information. An area fill algorithm can identify connected components in the drawing that are closed areas consisting of topologically connected pixels. A new integrated labelling algorithm that constructs statistical region and topological graph features at the same time is described in the next section.

4. FEATURE REPRESENTATION AND EXTRACTION

After all of the preprocessing work has been completed, the system has to extract the features of the drawing. In cartoon animation, the region features can be classified into two categories: (1) shape of regions, and attributes affecting their appearance. (2) spatial layout and connectivity of components of each region. According to this classification, two types of feature representations are used in our drawing analysis. They are (1) *statistical region feature vector* representation for primitive image features of each region in a drawing, and (2) *topological attribute graph* representation

for the spatial relationship between regions. We now show how these features are obtained using a new integrated drawings labelling algorithm.

4.1. Statistical region features

In each region, several useful statistical image features need to be extracted and can be used as a basis for similarity measurement between regions. These statistical features are defined as follows.

1. *Area*: the number of pixels in a specific region. This can be done by counting the pixels during labelling process.
2. *Bounding box*: the smallest rectangle that can enclose a specific region. This can be done by recording the minimum and maximum X, Y axis locations in the labelling process.
3. *Aspect ratio*: the ratio of the length of a bounding box and its width.
4. *Density*: the ratio of the area of a specific region over the area of its bounding box.

Furthermore, the synchronized movements of a group of objects which are related to each other must be considered. We treat this information as *motion direction*. Once the object has been located in the next drawing, its direction of motion is used to constrain the search in the subsequent drawings. This reduces the possibility of a mismatch, while also speeding up the matching process. Every time a match is found, the direction of motion is updated from the current drawing. The region statistical feature vector can be represented by the RegionFeature class structure described as follows:

```
Class RegionFeature {
    int         RegionID;
    int         RegionColor;
    Point       FloodSeed;
    Point       CenterPoint;
    Vector      MotionDirection;
    int         Area;
    Rectangle   BoundingBox;
    float       Elongateness;
    float       Density;        }
```

The RegionID is a unique region label number. This number will be flooded into the whole area of current region and can be used as a classification among regions. The FloodSeed is a seed position used by the area fill algorithm to flood the region. The RegionColor is the final colour in this region.

4.2. Topological attribute graph structure

The attribute graph representation with attributes attached to vertices and edges offers powerful capabilities that are very useful for image analysis, such as the proper handling of actual dimension and hierarchy of the images, and the topological invariance. To achieve similarity measurement for logically structured drawings, we introduce a planar

topological graph representation of these drawings. A *topological attribute graph* is a graph that describes the planar spatial relationship of a drawing and is represented by the form

$$D_g = (V, E), \tag{1}$$

where $V = \{v_1, \ldots v_n\}$ is a finite set of n region vertices with statistical region attributes (RegionFeature class) and $E = \{e_1, \ldots, e_r\}$ is a set of r edges with adjacency relationship attributes, that is, relationships connected by border arc segments. Figure 3 shows the topological relationship among regions in a drawing with vertices $V = \{A, B, C, D, E\}$ and edges $E = \{(A,B), (A,C), (B,C), (B,D), (B,E), (C,D), (C,E), (D,E)\}$.

4.3. Integrated labelling algorithm

As the result of region feature extraction, the feature vectors and topological graphs of regions in each drawing will be created by using the RegionFeature class structure and the attribute graph. We attempt to accomplish these tasks in a labelling process.

Riekert[20] proposes a transformation that can determine certain topological features, such as neighbourhood and connectivity, for each drawing. This transformation is performed in three steps : (1) connected component labelling, (2) vectorizing the components, and (3) establishing topological relationships. Riekert†s algorithm combines all three steps in a single pass. The input of this algorithm is an image with colour map representing a classification of the plane. The output of the algorithm is a topological network of regions in a graph representation.

Riekert's algorithm does not meet our requirements completely because our input drawings have bilevel border arc segments with varying line width. Also, we need more information, such as area, bounding box and seed of flood, extracted during this labelling process. So, we propose a new integrated labelling algorithm using the scan line conversion method to generate statistical and topological information at the same time, and then an attribute graph matching algorithm can be processed directly after this labelling task has been done.

Our *integrated labelling algorithm*, as shown in Algorithm 1, is done in a line-by-line, left-to-right, top-to-bottom basis. For each scan line in this bilevel (black–white) raster bitmap drawing, the algorithm checks whether each pixel is a clean (white) pixel, border (black) pixel or a labelled pixel (with label number). A clean pixel means a new

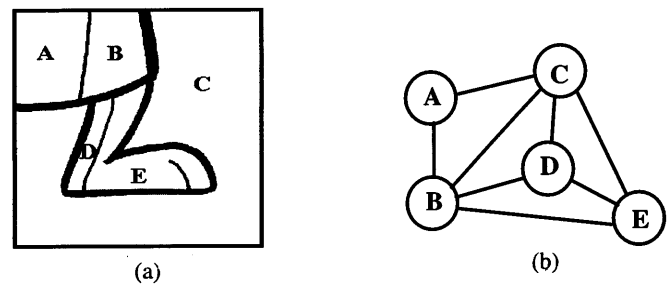

Figure 3. Topological relationship representation; (a) original regions with labels; (b) corresponding attribute graph

Algorithm 1. Integrated Labeling
Input. A bilevel (black-white) drawing image.
Output. A region information table with both statistical region features and a topological graph structure.
Method.

```
Initialize RegionID;
For each scan line x in this drawing
    For each y pixel in a scan line
        If pixel value p(x, y) is clean (nonlabeled pixel) then
            Call subroutine AddNewRegion(RegionID) to allocate a new region feature structure in
                region information table;
            Increase RegionID by one;
            LineWidth = 0;
            Add this seed into its region feature structure FloodSeed = (x, y);
            Call subroutine AreaFill(x, y, NewLabelColor, right, clean, LineWidth) to fill this region
                using area fill algorithm with new label color number;
            Calculate remaining statistical region features and add into its region feature structure;
        End-of-If
    End-of-For
End-of-For
```

End-of-Algorithm Integrated Labeling.

Algorithm 1. Integrated labeling

region is appearing. This new region and its border will be flooded with a unique label colour. When the scanning reaches the bottom-right corner of the image, there will be no more clean pixels (no new region).

To keep track of the statistical and topological information encountered during a line-scanning cycle, a dynamic *region information table* must be maintained by the algorithm. This region information table, as shown in Table I, contains intra-region feature data and inter-region adjacent data. Each entry stores feature information, such as region ID, area, aspect ratio and original flood seed position, which is the same as the RegionFeature class structure stated previously. Inter-region adjacent data records the adjacent relationship between regions, including the degree (branching number) of the vertex and the *adjacent region set*.

During every scan cycle in the labelling process, the entities in the table are examined and updated if necessary. Whenever a new region is found, it is added to the region

Table I. A region information table example

Region ID	Region colour	Degree	Adjacent region set	(Other statistical features)
A	Black	2	B, C	...
B	Green	4	A, C, D, E	...
C	Red	4	A, B, D, E	...
D	Brown	3	B, C, E	...
E	Black	3	B, C, D	...

information table by the *AddNewRegion* routine. Several implementation details should be noticed : (1) the pixel of top-left corner in a drawing image should not be a border colour. (2) a border pixel should not be accumulated into a region area.

A new recursive area fill (flood) algorithm *AreaFill* to label a closed region as described in Algorithm 2 has been designed. This flood algorithm will substitute connected pixel values starting from a seed position for a new label colour number, if the flooded pixel is a clean pixel or is a right or down connected border. Also, this algorithm will check the left and up connected pixels for a new adjacent colour number. When an adjacent region is encountered, the region ID of each adjacent region is added into the region set of the other by the *AddNewAdjacent* routine. This flood process will stop if its target pixel value is a labelled colour.

5. SIMILARITY MEASUREMENT

The *similarity* between two image regions, a *target region* and a *reference region*, is defined as the maximum number of similar features that are common between them, or is defined as the minimum changes that need to be performed on one object in order to produce the other region.

Algorithm 2. 4-Connectivity Recursive Area Fill

Input. An image with closed regions, seed position *(x, y)*, label color, source direction, source type, and border line width.

Output. A specified region with new label color.

Method.

```
If pixel value p(x, y) is clean and source type is clean, then
    Replace pixel with new label color;
    Increase RegionArea by one;
    Record x, y min. and max. value for BoundingBox;
    Call AreaFill(x-1, y, LabelColor, left, clean, LineWidth);
    Call AreaFill(x+1, y, LabelColor, right, clean, LineWidth);
    Call AreaFill(x, y+1, LabelColor, down, clean, LineWidth);
    Call AreaFill(x, y-1, LabelColor, up, clean, LineWidth);
Else
    If p(x,y) is border and LineWidth < WIDTH then
        Replace pixel with new label color;
        LineWidth = LineWidth + 1;
        Call Area-Fill(x+1, y, LabelColor, right, border, LineWidth);
        Call Area-Fill(x, y+1, LabelColor, down, border, LineWidth);
    Else
        If p(x, y) has a color and not equal to LabelColor then
            Call subroutine AddNewAdjacent(x, y) to record (x, y) label into adjacent region
                set;
        End-of-If
    End-of-If
End-of-If
End-of-Algorithm Recursive Area Fill.
```

Algorithm 2. 4-Connectivity recursive area fill

Previous proposed methods of similarity measurement of regions are classified into three categories.[21–24] (1) For spatial relationship matching, each region is represented by a 2-D string. The problem of region matching then becomes a problem of 2-D subsequence matching. (2) For statistical region feature matching, each region is represented by a feature vector. The distance of each feature vector between the current target and previous reference drawings is calculated and the smallest is selected. (3) For graph feature matching, the structure of each drawing is described by topological graph notation. Measurement is performed by structural matching between the current target and the previous reference graph.

5.1. Progressive region matching procedure

The key idea of our progressive region matching procedure is that almost all of the changes which occur in animation can be tracked over a time interval by region features. Using statistical and topological features, we perform our automatic region matching procedure in three levels progressively. If the target region cannot find a best match from the reference regions, then the matching process will go deeper to another level. That is, the matching procedure will stop if a best matched reference region is found in some level.

Level 1. Region comparison

For each target region, find a most similar (best matched) reference region from the previous reference drawing using statistical features.

To avoid exhaustive search of each region in the previous reference drawing, several constraints must be satisfied before going to the next checking procedure. Subscripts c and p stand for the current and previous drawings, respectively. H stands for the threshold value of the specific feature.

(a) the spatial position of the reference region in the previous drawing must be in an effective range related to target region in the current drawing:

$$|CenterPoint_c + MotionDirection_p - CenterPoint_p| < H_{Range} \tag{2}$$

(b) the reference region has a similar shape and does not make a big deformation compared with the target region:

$$|Area_c - Area_p| < H_{Area} \tag{3}$$

$$|Elongateness_c - Elongateness_p| < H_{Aspect} \tag{4}$$

$$|Density_c - Density_p| < H_{Density} \tag{5}$$

Level 2. Graph match

For each target region with unknown colour, find a best matched reference region using topological relationship features.

This takes into account both the distribution of the colours and the adjacency of these colours. The recognition problem for a topological feature is transformed into an attribute graph matching problem with several considerations:

(a) The graph matching process must precede the comparison between graphs and provide a planar graph mapping.
(b) The graph matching process must consider the order of adjacency in the neighbourhood and due to the subdivision and deformation characteristics of animation, it must be rotation-free.

Level 3. Alternation

For each remaining nonmatched target region, find a background colour as an alternative solution. If no colour is picked up in the above matching procedure, the original colour in the same place on the previous reference drawing will be selected.

In our system, the similarity measurement of both statistical and topological features between drawings is based on the minimum distance (cost function) criterion. The closer a reference candidate is to a target region, the more similar the reference candidate to the target region is. We do not attempt to solve this problem using a fully general graph matching method because it is very time consuming to compare region feature one-by-one.

5.2. Statistical feature similarity measurement and weight assignment

In order to determine the similarity of statistical region features between two corresponding regions, a target region r_i^c of the current drawing D_r^c and a reference region r_j^p of the previous reference drawing D_r^p, we need to define the similarity measurement for two regions first.

For each (i, j) pair of target region i, $r_i^c \in D_r^c$, and reference region j, $r_j^p \in D_r^p$, the *distance* (*cost function*) of the kth statistical region feature is calculated as

$$\delta(i, j, k) = |r_{i^c,k} - r_{j^p,k}| \tag{6}$$

where $r_{i^c,k}$ and $r_{j^p,k}$ are the kth feature values of regions r_i^c and r_j^p, respectively. By giving m statistical features, an (i, j) pair of target region i and reference region j is called a *qualified candidate region* pair, if for every statistical feature k,

$$\delta(i, j, k) < H_k, \; 1 \le k \le m, \tag{7}$$

is satisfied, where H_k is a given threshold for the kth feature.

Once all of the differences of statistical features are computed, the overall deviation of the qualified candidate region pair can be computed. For each qualified candidate region pair (i, j), the *overall deviation* is calculated as

$$\Delta(i, j) = \sum_k W_k \delta(i, j, k) \tag{8}$$

where W_k is the weighting factor of the kth feature. The value $\Delta(i, j)$ is then used to determine the most similar region. Each evaluated value varies according to similarity. The better the similarity, the smaller the value.

The most similar region pair (i, j) is the region pair with the smallest overall deviation

value among qualified candidate region pairs. Therefore, for each target region *i*, the *cost of the most similar region j* is defined as

$$S_i = \min_{j, r_j^P \in D_r^P} \Delta(\mathrm{i}, \mathrm{j}) \tag{9}$$

The reference region *j* with the smallest overall deviation value is the most similar region of *i*.

After this process, the colour of the region in the current target drawing will be inherited from the most similar corresponding region in the previous reference drawing, if all of the measured statistical region features are within the accepted thresholds. Sometimes, it will happen that two different target regions select the same reference region as the most similar one. If the conflict ranking condition occurs, a re-ranking process will handle it.

5.3. Topological feature similarity measurement

After the first level of region comparison, there still possibly remain several unmatched regions in which their colours have not yet been decided. The process might not work properly if the related feature differences between target and reference regions did not qualify. We must go deep into the second (graph match) level to decide the colour of a target region. In the following of the paper, an attribute graph is abbreviated as a graph.

In graph representation, a *target vertex* represents a target region with unknown colour in the current drawing D_g^c, and a *reference vertex* is one of the reference regions in the previous drawing D_g^P. A *target graph* of target vertices is a one-level subgraph of a current drawing graph. A *reference graph* of reference vertices is also a one-level subgraph of a previous graph. A one-level graph consists of four components: a root vertex for either a target or a reference vertex, the edges emanating from the root vertex, the vertices on which these edges terminate, and the edges between terminal vertices. Both the target and the reference graph can be constructed from the adjacent region set in region information tables and only those vertices of adjacent closed regions need to be produced in these graphs.

To decide the colour of a target region, we first consider the adjacency relationship and colour distribution of different regions. With the help of the correspondence of topological relationship and surrounding neighbour colour, we can get more evidence in support of the colour of those unknown target regions. Consideration of the adjacent colour relationship rejects a reference vertex if its colour adjacency relationship is inconsistent with the colour adjacency topology of the target vertex. If almost all of the adjacent colours appear in the adjacent region set of a reference vertex, we may conclude that this reference vertex has the colour of the target vertex.

Next, for the similarity measurement of a topological graph, it is considered to be a task of finding the best matching graph between the graph D_g^P of the previous drawing and the target graph, a subgraph of D_g^c, in the current drawing. We can transform this colour comparison problem into a non-exact graph matching problem[25] where corresponding vertices may have different colours. The simplest way for graph matching is via a brute force (round-robin permutation) method. However, since the running cost for this method would be proportional to the number of regions, it is inappropriate for a large scale graph implementation. Eshera and Fu[22] propose an approach for calculating a

global distance between attributed relational graphs. The distance between two graphs is defined as the minimal total cost of the sequence of transformations that must be performed on one of the two graphs in order to produce the other graph. Therefore, an essential part of the problem involves optimization over all valid sequences of transformations to find the one with minimum total cost.

Fu's approach is still very inefficient for our system. Using the same principle, a new graph-matching algorithm is proposed using the statistical and topological information of the surrounding region. Our graph matching approach, as described in Algorithm 3, consists of three parts:

1. *Candidate finding*: find out a qualified candidate vertex from the previous graph.
2. *Mapping generation*: specify the target and the qualified candidate vertex, generate all the drawing graph mappings from respective adjacent region set.
3. *Cost calculation*: for each mapping, calculate the accumulation cost and find out the qualified candidate vertex of the best match.

Let $V^c = \{v_1{}^c, \ldots v_n{}^c\}$ and $V^p = \{v_1{}^p, \ldots, v_m{}^p\}$ be the sets of vertices of graphs $D_g{}^c$ and $D_g{}^p$, respectively, where $n \geqq 1$ and $m \geqq 1$. A target vertex $v_i{}^c$ is in V^c *with adjacent region set* $A_i{}^c = \{v_1{}^c, \ldots, v_{n'}{}^c\}$, a reference vertex $v_j{}^p$ is in V^p *with adjacent region set* $A_j{}^p = \{v_1{}^p, \ldots, v_{m'}{}^p\}$. A pair (i, j) of target vertex $v_i{}^c$ and reference vertex $v_j{}^p$ is called a *qualified candidate vertex pair*, if they satisfy the following criteria which filter out most of the cases with significant appearance differences. The reference vertex $v_j{}^p$ is called a *qualified candidate vertex*.

1. As described in *Level 1*(a) of the region matching procedure, the spatial position of reference vertex $v_j{}^p$ must be within an effective range of the target vertex i.
2. The ratio of degree between reference vertex $v_j{}^p$ and target vertex $v_i{}^c$ must be in the range of two thresholds,

$$H_{DegreeL} \leq \frac{\#(A_j{}^p)}{\#(A_t{}^c)} \leq H_{DegreeH} \tag{10}$$

   where $\#(A_i{}^c)$ and $\#(A_j{}^p)$ are the respective cardinalities of adjacent region sets of the target vertex $v_i{}^c$ and the reference vertex $v_j{}^p$.
3. The ratio of the number of the same colour sets between the reference vertex and the target vertex must be above a threshold.

$$\frac{\#(C_j{}^p \cap C_i{}^c)}{\#(C_i{}^c)} \geq H_{Same} \tag{11}$$

   where $\#(C_i{}^c)$ and $\#(C_j{}^p)$ are the numbers of colours in adjacent region sets of the target vertex $v_i{}^c$ and the reference vertex $v_j{}^p$, respectively.

Then, for a qualified candidate vertex pair (i, j), to generate a *drawing graph mapping* is to find a mapping $T_{i,j}{}^k$: $[n'] \rightarrow [m']$ between vertices of the target graph $V_i{}^c$ and the reference graph $V_j{}^p$, such that for each $v_{i'}{}^c$ in $A_i{}^c$, there exists one and only one $v_{j'}{}^p$ in $A_j{}^p$, and $T_{i,j}{}^k(v_{j'}{}^p$, where $\#(A_i{}^c)$ is n' and $\#(A_j{}^p)$ is m'. More than one $v_{i'}{}^c$ can map to the same $v_{j'}{}^p$. Furthermore, there may exist k such mapping(s), $k \geqq 1$.

The one-level target graph of a target region with unknown colour can be constructed by obtaining the adjacent region ID and colour from the adjacent region set in the region information table to decide the vertices and the edges. Then an attribute graph matching between a target graph and a reference graph will be performed.

Algorithm 3. Attribute Graph Matching

Input. A full previous graph and a target vertex v_i^C with attribute graph.

Output. A candidate vertex with best matched fitness or a null match.

Method.

BestFitness=null;

For each reference vertex v_j^P in V^P

If v_j^P satisfies the following conditions:

(1) $| CenterPoint_i + MotionDirection_j - CenterPoint_j | < H_{Range}$ and

(2) $H_{DegreeL} \leqq \#(A_j^P) / \#(A_i^C) \leqq H_{DegreeU}$ and

(3) $\#(C_i^C \cap C_j^P) / \#(C_i^C) \leqq H_{Same}$

Then

For each $v_{i'}^C$ in A_i^C and $v_{i'}^C$ has color t

Add vertex $v_{i'}^C$ into color set $B_{i',t}$;

For each $v_{j'}^P$ in A_j^P

If $v_{j'}^P$ has the same color t as $v_{i'}^C$ Then

Add vertex $v_{j'}^P$ into color set $B_{j',t}$;

End-of-For

End-of-For

For each existing color t in color list B

If $\#(B_{i',t}) = 1$ and $\#(B_{j',t}) = 1$ Then

$v_{i'}^C$ in $B_{i',t}$ and $v_{j'}^P$ in $B_{j',t}$ are the vertices of mapping Type 1;

If $\#(B_{i',t}) > 1$ or $\#(B_{j',t}) > 1$ Then

those $v_{i'}^C$ in $B_{i',t}$ and $v_{j'}^P$ in $B_{j',t}$ are the vertices of mapping Type 2 in group t;

End-of-For

Generating all of the mappings according to Type 1 unique same mapping color pairs and Type 2 same color groups by permutation;

For each generated mapping

Calaulate the evaluation value;

If $FitValue(v_j^P) > MaxFit$ Then

$BestFitness = v_j^P$;

$MaxFit = FitValue(v_j^P)$;

End-of-If

End-of-For

End-of-If

End-of-For

If *BestFitness* = *Null* Then

Return (*Null Match*);

Else

Return (*BestFitness*);

End-of-If

End-of-Algorithm Attribute Graph Matching.

Algorithm 3. Attribute graph matching

It is necessary to define a graph similarity measurement that evaluates the similarity between two graphs. From now on, we need to consider the mapping and the cost evaluation criteria between adjacent regions in the similarity measurement procedure. For each qualified candidate vertex pair (i, j) of mapping, if the kth mapping $T_{i,j}^{k}: [n'] \rightarrow [m']$ from v_i^c to v_j^p is specified, the cost evaluation criteria are defined as follows:

1. *Same surrounding colour*. This information provides strong support for the decision of similarity in corresponding regions (i, j) of different drawings such that the more the common colours, the more similar the two corresponding regions. The cost of the same surrounding colour is defined as the number of the vertex pairs of the same colours with respect to the graph mapping $T_{i,j}^{k}$:

$$CostSameColor(i, j, k) = \#\{(v_{i'}{}^c, v_{j'}{}^p) \mid Color(v_{i'}{}^c) = Color(v_j{}^p), (v_{i'}{}^c, v_{j'}{}^p) \in T_{i,j}^{k}\}, \tag{12}$$

   where $Color(v_{i'}{}^c$ and $Color(v_{j'}{}^p)$ are the infilled colours of the $v_{i'}{}^c$ and $v_{j'}{}^p$, respectively.
2. *Same surrounding edge with same colour pair*. It also provides strong evidence for the decision of similarity in corresponding regions (i, j). In this criterion, two corresponding edges are called the same colour edge if these two edges have the same colour pairs at both ends. The cost of the same surrounding edge is defined as the number of the vertex pairs, for which their corresponding edges are the same colour edge, with respect to the graph mapping $T_{i,j}^{k}$,

$$\begin{aligned} CostColorPair(i, j, k) = \#\{(v_{i'}{}^c, v_{j'}{}^p) \mid EdgeOf(v_{i'}{}^c, v_{i''}^c) \\ = EdgeOf(v_{j'}{}^p, v_j^p), (v_{i'}{}^c, v_{j'}{}^p) \text{ and } (v_i^c, v_j^p) \in T_{i,j}^{k}\} \end{aligned} \tag{13}$$

   where $EdgeOf(v_{i'}{}^c, v_i^c)$ and $EdgeOf(v_{j'}{}^p, v_j^p)$ are the ordered colour pairs of the infilled colours in vertices $v_{i'}{}^c$, v_i^c and $v_{j'}{}^p$, v_j^p, if there exists an edge from $v_{i'}{}^c$ to v_i^c, and from $v_{j'}{}^p$ to v_j^p, respectively.
3. *Extra neighbour colour*. It provides the information about similar colours in nearby regions. The cost of the extra neighbour colour is defined as the degree of the colour difference between candidate reference vertex and target vertex,

$$CostExtraColor(i, j) = \#(C_j{}^p - C_i{}^c) \tag{14}$$

   For each pair (i, j), no matter what kind of mapping is generated, the cost of the extra neighbour colour is the same.
4. *Degree difference*. It provides the information about region splitting. The cost of the vertex degree difference is defined as the difference of the vertex branches between two vertices,

$$CostDegree(i, j) = \#(A_j{}^p) - \#(A_i{}^c) \tag{15}$$

The similarity of the kth mapping between a given target graph and a candidate graph measured as the accumulation of the evaluated values of corresponding vertices and edges, also called *fitness*, is defined as follows:

$$\begin{aligned} FitValue(i, j, k) = W_{cs} \cdot CostSameColor(i, j, k) \\ + W_{cc} \cdot CostSameEdge(i, j, k) + W_{ce} \cdot CostExtraColor(i, j) \end{aligned} \tag{16}$$

$$+ W_{cd} \cdot CostDegree(i, j)$$

where W_{cs}, W_{cc}, W_{ce} and W_{cd} are the weighting factors of the costs of same surrounding colour, same surrounding edge, extra neighbour colour and degree difference, respectively.

According to the cost evaluation criteria, we set up heuristics to speed up the mapping generating process. That is, the mapping process categorizes colours in adjacent regions of target and candidate vertices into three *colour mapping types*: (1) Type 1—*unique colour mapping pairs*. This type has only the same colour pairs of one-to-one mapping from target graph to reference graph. Each pair has a unique colour in their adjacent regions. (2) Type 2—*same colour groups*. This type collects all of the vertices both in target and candidate vertices with the same colours into groups. Each group has its own colour. (3) Type 3—*miscellaneous unmatched colours*. This type is the remaining colours in both adjacent regions of target and candidate vertices.

Assume that a target vertex i has $\#(A_i{}^c) = n'$ adjacent regions and candidate vertex j has $\#(A_j{}^p) = m'$ adjacent regions, $n' \leqq m'$. The numbers of adjacent regions in mapping types 1, 2, 3 of vertex i and vertex j are n'_1, n'_2, n'_3 and m'_1, m'_2, m'_3, respectively, $n' = n'_1 + n'_2 + n'_3$ and $m' = m'_1 + m'_2 + m'_3$, also $n'_1 = m'_1$.

For each Type 1 vertex in $A_i{}^c$, there exists one and only one mapping between target and candidate vertex, that is, a one-to-one and onto mapping $T_{i,j}^{k,1}$: $[n'_1] \rightarrow [m'_1]$. Each Type 1 vertex can be placed in a one-to-one and onto correspondence between target and reference vertices. There may exist more than one group of Type 2 vertices. For the rth group of Type 2 vertices in $A_i{}^c$, $r \geqq 1$, if $n_{2,r}{}' > m_{2,r}{}'$ then we choose $m_{2r}{}'$ vertices from $n_{2,r}{}'$, and generate a $T_{i,j}{}^{k,2r}$: $[n^{2,r}{}'] \rightarrow [m^{2,r}{}']$ permutation mapping. The number of ways of choosing p out of q different objects disregarding order, $p > q$, is given by $p(p - 1) \cdots (p - q + 1)$. We can permute each of these groups in the same way for the remaining $n'_2 - m'_2$ vertices. For Type 3 vertices, we choose $n'_2 - m'_2 + n'_3$ vertices from m'_3, and generate a $T_{i,j}{}^{k,3}$: $[(n_2{}' - m_2{}' + n_3{}')] \rightarrow [m_3{}']$ mapping. Because the Type 3 vertices will not affect the accumulation of fitness value, we can discard the permutation of them.

More than one mapping will be generated if there is no one-to-one mapping with same vertex degree and same adjacent colour. For example, as in Figure 4, the vertex X is the target vertex in target graph with $n' = 3$ and vertex B is the candidate vertex with $m' = 5$. Then, $n_1{}' = m_1{}' = 1$ with colour Yellow; $n_{2,1}{}' = 1$ and $m_{2,1}{}' = 2$ with colour Black, $n_3{}' = 1$ with colour Green, and $m_3{}' = 2$ with colours Blue and Gray. There are

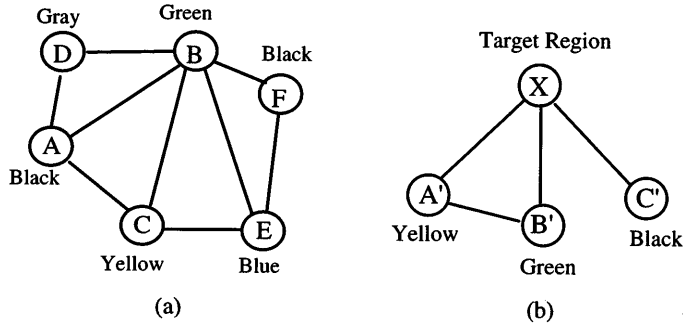

Figure 4. An example of graph matching: (a) graph of the previous drawing; (b) target graph with target vertex X

six mapping patterns as : (1) {(A', C), (B', F), (C', A)}, (2) {(A', C), (B', D), (C', A)}, (3) {(A', C), (B', E), (C', A)}, (4) {(A', C), (B', A), (C', F)}, (5) {(A', C), (B', D), (C', F)}, (6) {(A', C), (B', E), (C', F)}. We only consider two of them: {(A', C), (*), (C', A)} *and* {(A', C), (*), (C', F)}. The (*) means that it is a *don't-care permutation.*

After these mappings are found, we need to compare them with each other by calculating the similarity measure of related graph attributes. Because there may exist many mappings, the cost of the most similar vertex should be the one with the largest fitness value. Therefore, for each target vertex v_i^c, the cost of the most similar reference vertex v_j^p can be defined as

$$M_i^c = \max_{\forall j, \forall k, v_j^p \in D_g^p} FitValue(i, j, k) \tag{17}$$

As a result of the graph match level, the colour of the region in the current target drawing will be inherited from the most similar (maximum fitness) corresponding region in the previous reference drawing.

We use Figure 4 as our example again. The corresponding cost evaluation result is listed in Table II. The evaluated weighting factors of fitness in equation (16) are as follows: W_{cs} is 10·0 for each vertex with the same colour; W_{cc} is 5 for each vertex with one edge of the same colour; W_{ce} has a penalty of −3·5 for each vertex with an extra colour; W_{cd} has a penalty of −2·5 for each vertex with one branch difference. The result of evaluation is the summation of all matching factors. In this case, since vertex E has the largest fitness value of 35·0, so the colour of vertex E (blue) will be selected as the colour of the target region according to the target-reference mapping pairs (X, E), (A', C), (B', B), (C', F)}.

6. EXPERIMENTAL RESULTS

A prototype of an automatic cels painting system, as shown in Plate 1, using our proposed drawing region recognition method, was implemented on a workstation under a Unix system based on the C++ language and X11/Motif window manager. This prototype system includes two major canvases for reference and current drawings, and four minor canvases for following drawings. Several image processing functions can be selected manually to modify the image details of drawings and to change the region colour which inherits the wrong colour. Our prototype system also provides relevant information about show and character model databases, weight and threshold settings,

Table II. Results of cost evaluation

Vertex label	Different mapping	Vertex degree	Candidate vertex	Same colour	Same edge	Extra colour	Branch difference	Best matching total value
A	1	3	Y	2	1	1	0	21·5
B	2 (best)	5	Y	3	0	1	2	21·5
C	1	3	Y	2	0	1	0	16·5
D	—	2	N	2	0	1	1	0
E	1	3	Y	3	1	0	0	35·0
F	—	2	N	1	0	1	1	0

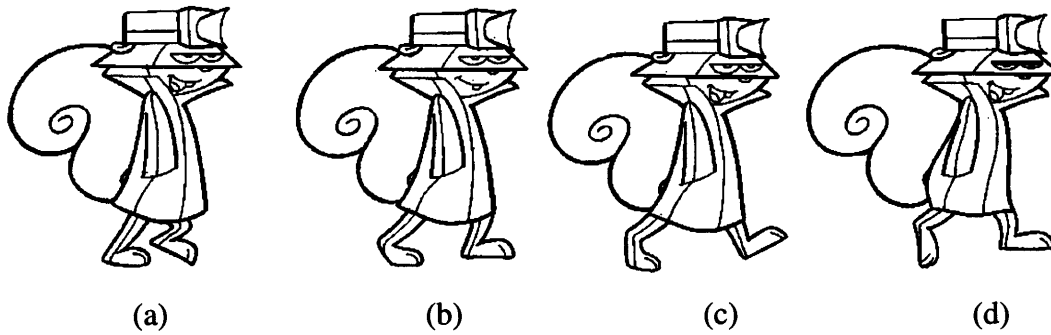

Figure 5. A sequence of animation drawings

statistical and topological features. The original drawing images have 512×486 pixels and contain 72 regions on average in the case of Plate 1. The speed of similarity measurement process takes approximately 4·8 seconds per drawing. At present, from 65 to 95 per cent of the regions can inherit the correct colour from the corresponding drawing depending on the appropriate tuning of threshold values and weight parameters. The infilled colours shown here just illustrate the correspondences between two matched regions, and are not actual colours appearing in the character colour model. An example and several special problems will be discussed in the following subsections.

6.1. Examples

To test the iterative model of our automatic cel painting procedure, we use the sequence of animation drawings in Figure 5 as our experimental samples. After the integrated labelling process, we can obtain a region information table. This table contains the statistical region and the topological graph features. Then we can finish the first region comparison level in our automatic painting process. The thresholds and weights of region features used in this system are listed in Table III.

6.2. Future work

Several cases give rise to complex results that will reduce the precision of similarity measurement and need to be solved in the future. These include: (1) A big action will

Table III. The thresholds and weights of all the features used in this system

Changing ratio criterion	Area	Motion offset (pixel)	Aspect ratio	Density	Same colour	Degree rate low	Degree rate high
Threshold	1·5	40	0·8	0·5	0·6	1	1·8
Weighting factor	2	0·5	1·5	1	—	—	—

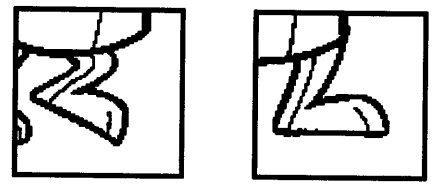

Figure 6. Big action

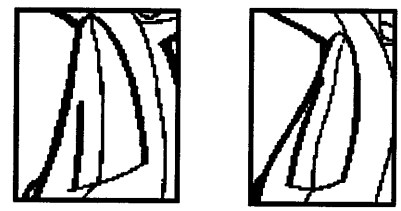

Figure 7. Split region

cause a significant change in the shape of some regions, as shown in Figure 6. It will be quite difficult to find corresponding regions using only statistical features. (2) Some drawings may have mismatching regions due to object occlusion. This will split regions into more than one subregion, or merge several regions into one big area, as shown in Figure 7. (3) Sometimes, an error propagation situation will occur. That is, if some regions are filled with a wrong colour, then this wrong colour will be propagated to the next drawing and to the next. (4) Weights and thresholds in statistical feature similarity measure must be tuned carefully according to different types of cartoon styles. (5) It will be time consuming if the number of regions is too large. The matching process will take a lot of computation time to find a best fit region. Sometimes, the time a matching process takes will be more than a manual process does. (6) Cases such as a mouth opening or an actor appearing on stage, are very complicate to solve, as shown in Figure 8. (7) Interpolation is not linear. The distance between two corresponding regions is not always regularly divided. (8) Actions will be separated into different layers if necessary, as shown in Figure 9. These cases are worth further investigation and will be our future work.

7. CONCLUSIONS

Animation is the basic technology for several applications, such as scientific visualization, visual simulation, virtual reality and multimedia. Furthermore, a number of researchers use video for recording data during experiments and then use animation for further data analysis. Technology for motion analysis and video understanding is still in the develop-

Figure 8. No correspondence

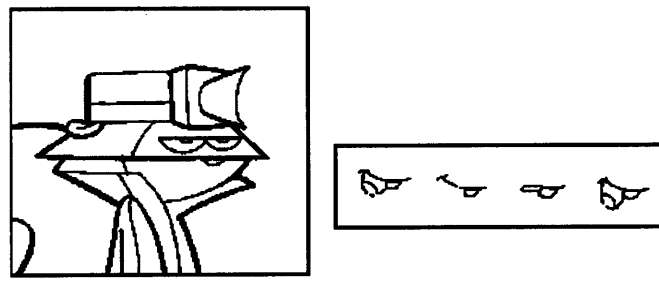

Figure 9. Layers separated

ment stage. The model-based recognition that requires reasoning computation for video understanding is far beyond the capabilities of current artificial intelligence and pattern recognition techniques.

In this paper, an automatic cel painting system is proposed. In the feature extraction step, an integrated labelling method is employed. A new approach of region similarity measurement using feature vector and partial graph matching is presented. Matching is performed according to the combination of the statistical region features and topological graph features. A prototype system has been constructed and the experimental results show the feasibility and practicability of the proposed approach. This automatic cel painting mechanism can offer animation producers a cost-effective alternative to traditional methods. It signals a great improvement in the working environment where 2-D animation is produced. Our system includes many facilities for spatial analysis and motion analysis. It also outlines the various video object processing aspects of animation characteristics necessary for providing content-based video access. This approach can be adapted to other problems in data analysis.

There are many important future works which need to be done for computer-assisted cartoon animation, such as automatic in-betweening, resolution-independent output, colour calibration, large cartoon image database, user interface, exposure sheet editor, antialiasing, rejection-threshold learning procedures, motion tracking and deformation detection.

ACKNOWLEDGEMENTS

We would like to thank anonymous referees for their helpful comments and suggestions. We also thank Wang Film Production Co. for their provision of cartoon drawings.

REFERENCES

1. J. L. Encarnacao *et al*., 'Advanced research and development topics in animation and scientific visualization', in R. A. Earnshaw and D. Watson (eds), *Animation and Scientific Visualization*, Academic Press, 1993, pp. 37–73.
2. F. Leah and M. lee, *Insider Autodesk Animator: The Complete Guide to Animation on a PC*, New Riders Publishing, 1990.
3. N. Magnenat Thalmann and D. Thalmann (eds), *Computer Animation '91*, Springer-Verlag, 1991.
4. B. Tony and R. Cheryl, *Using MacroMind Director*, Carmel QUE, 1990.
5. J. Lasseter, 'Principles of traditional animation applied to 3D computer animation', *Computer Graphics*, **21**, (4), 35–44 (1987).
6. N. Magnenat Thalmann and D. Thalmann, 'An indexed bibliography on computer animation', *IEEE Computer Graphics and Applications*, **5**, (7), 76–86 (1985).
7. G. MacNicol, '2D animation: alive and well', *Computer Graphics World*, March (1990), pp. 41–50.

8. M. Morrison, *Becoming a Computer Animator*, SAMS Publishing, 1994.
9. P. Sorensen, 'User-friendly animation', *Computer Graphics World*, March 1990, pp. 129–133.
10. C. X. Durand, 'The TOON project: requirements for a computerized 2D animation system', *Computer & Graphics*, **15**, (2), 285–293 (1991).
11. J. F. Blinn, 'Compositing, part II: practice', *IEEE Computer Graphics and Applications*, **14**, (6), 78–82 (1994).
12. B. A. Wallace, 'Merging and transformation of raster images for cartoon animation', *Computer Graphics*, **15**, (3), 253–262 (1981).
13. C. X. Durand and Daniel Faguy, 'Rational zoom of bit maps using a B-spline interpolation in computerized 2D animation', *Computer Graphics Forum*, (9), 27–37 (1990).
14. C. X. Durand,'Bit map transformations in computerized 2D animation', *Computer & Graphics*, **13**, (4), 433–440 (1989).
15. J. S. Madeira, A. Stork and M. H. Groβ, 'An approach to computer-supported cartooning', *The Visual Computer*, (12), 1–17 (1996).
16. N. Magnenat Thalmann and D. Thalmann, *Computer Animation: Theory and Practice*, 2nd edn, Springer-Verlag, 1990.
17. J. W. Patterson and P. J. Willis, 'Computer assisted animation: 2D or not 2D?', *The Computer Journal*, **37**, (10), 829–839 (1994).
18. M. Levoy, 'The design and implementation of a large-scale computer-assisted cartoon animation system', *Technical Report*, Wang's Film Production, 1984.
19. R. C. Gonzalez and R. E. Wood, *Digital Image Processing*, Addison-Wesley, 1992.
20. W. Riekert, 'Extracting area objects from raster image data', *IEEE Computer Graphics and Applications*, **13**, (3), 68–73 (1993).
21. J. R. Bach, S. Paul and R. Jain, 'A visual information management system for the interactive measurement of faces', *IEEE Trans. on Knowledge and Data Engineering*, **5**, (4), 619–628 (1993).
22. M. A. Eshera and K. S. Fu, 'A graphic distance measure for image analysis', *IEEE Trans. System, Man, and Cybernetics*, **14**, (3), 398–408 (1994).
23. S. Y. Lee, M. K. Shan and W. P. Yang, 'Similarity measurement of iconic image database', *Pattern Recognition*, **22**, (6), 675–682 (1989).
24. K. Wakimoto, M. Shima, S. Tanaka and A. Maeda, 'Content-based measurement applied to drawing image databases', *Storage and Measurement for Image and Video Database, SPIE* **1908**, 74–84 (1993).
25. S. K. Chang, *Principles of Pictorial Information Systems Design*, Prentice-Hall Inc., 1989.