Computer-assisted auto coloring for 2D animation characters

Qiu, Jie

2008

Qiu, J. (2008). Computer-assisted auto coloring for 2D animation characters. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/41312>

<https://doi.org/10.32657/10356/41312>

Computer-Assisted Auto Coloring for 2D Animation Characters

Qiu Jie

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfilment of the requirement for the degree of
Doctor of Philosophy

2008

Acknowledgments

I would like to thank Dr. Seah Hock Soon, my supervisor, for his time, effort, and constant support during this research. His invaluable suggestions, honest criticism, kind encouragement were a constant source of inspiration throughout the research. His personal enthusiasm and serious attitude to research work benefit me well. I am also thankful to Dr. Tian Feng, who gave me a lot of precious advice and help in my research. I would like to give my special thanks to Mr. Chen Quan, who often discussed with me about the project and problems met during the research. My heartfelt thank goes to Mr. Lu Ji, Mr. Lu Yixiang and Mr. Liu Qi, who gave me a lot of invaluable suggestions in programming skills. Deep thanks should also be given to Miss Zhang Haiwen and Mr. Lu Peng, who taught me a lot of knowledge about animation. I also feel thankful to my friends in the Interaction and Entertainment Research Centre (IERC) and Centre for Advanced Media Technology (CAMTech): Dr. Qian Kemao, Dr. Wu Zhongke, Dr. Zheng Jianmin, Dr. He Yin, Dr. Henry Johan, Miss Li Li, Mr. Lai Kok Sen, Mr. Zhang Jianfeng, Mr. Xie Xuexiang, Mr. Wang Yimin, Miss Xiao Xian, Miss Wang Jing, Mr. Chen Chen and Mr. Wei Lei for their warm assistance and encouragement.

I should appreciate the financial support for this research provided by Nanyang Technological University, and the space and equipment provided by IERC and CAMTech. Special thanks should also go to the staff at IERC and CAMTech, Assoc. Prof. William Russell Pensyl, Mr. Yee Lee Poh, Mr. Chua Ngee Tat, Mr. Alan Chng Hee Siah and Mr. Lee Shang Ping. They bring me many convenient accesses to the utilities in the lab. Finally, I would like to take this opportunity to express my gratitude to my family members. I sincerely thank my parents for their constant support, encouragement and blessings.

Contents

Acknowledgments	i
Abstract	v
List of Figures	vii
List of Tables	xi
1 Introduction	1
1.1 Motivation	1
1.2 Objective	2
1.3 Overview of our approach	4
1.4 Report organization	6
2 Survey	8
2.1 Animation production process	8
2.2 Existing 2D animation systems	13
2.3 Image matching techniques for animation frames	15
2.4 Auto coloring algorithms	17
3 Feature-based Region Matching	21
3.1 Region Segmentation	21
3.1.1 Region	22
3.1.2 Preprocessing	22
3.1.3 Region segmentation and labelling	25
3.2 Feature-based Region matching	30
3.2.1 Features of Region	30
3.2.2 FDV and TDV	39

3.2.3	Region matching	52
3.3	Results and Analysis	53
3.4	Summary	62
4	Topology Enhanced Auto Coloring	65
4.1	Stable topology	67
4.2	Hierarchical region matching	71
4.2.1	Hierarchization	71
4.2.2	Hierarchical region matching	79
4.2.3	Applicability analysis	80
4.3	Auto coloring process	82
4.3.1	Matching master frames	83
4.3.2	Coloring the first frame of a sequence	84
4.3.3	Coloring a sequence of frames	86
4.3.4	Postprocessing	90
4.4	Results and analysis	91
4.4.1	Test 1	91
4.4.2	Test 2	92
4.4.3	Test 3	97
4.4.4	Test 4	97
4.4.5	Tests 5 & 6	100
4.4.6	Test 7	104
4.5	Comparison with other algorithms	105
4.6	Summary	109
5	Auto Coloring with Character Registration	110
5.1	Character registration	111
5.1.1	Skeleton extraction	111
5.1.2	Skeleton registration	114
5.1.3	Skeleton matching	117
5.1.4	Component segmentation	122
5.2	Auto coloring process	124

5.2.1	Occlusion detection	124
5.2.2	Auto coloring	126
5.3	Results and Analysis	128
5.4	Limitation	133
5.5	Summary	135
6	Conclusion and Future work	137
6.1	Conclusion	137
6.2	Future work	140
6.2.1	Region feature expansion	140
6.2.2	Accuracy quantization	140
6.2.3	Automatic weight adjustment	142
6.2.4	Character relocation	143
6.2.5	Further investigation on 2D animation	143
	References	148
	Publication	157

Abstract

Inking/coloring of the individual animated characters for each frame is one of the most time-consuming and labor-intensive procedures in cel animation production. Many systems and algorithms have been proposed to assist the coloring of cel animation production. However, these systems or algorithms are either not efficient enough, in terms of the degree of dependence on user intervention, or yield to significant limitation. *Computer-Assisted Auto Coloring (CAAC)* for cel animation remains a tough issue in research.

In this thesis, we investigate a wide range of effective information in animation frames and accordingly propose novel auto coloring techniques by extracting the information and employ it to match the regions composing the character in each frame. Each region in an uncolored target frame is matched with a set of candidate regions in one or more colored reference frames, and then inherit the color of its best-matched one.

By investigating region features such as curve length, character points, area etc., we propose a novel feature-based region matching algorithm. The dissimilarities between region features are inventively measured by *Feature Dissimilarity Values (FDVs)* and a *Total Dissimilarity Value (TDV)* is calculated based on these FDVs, making the comparison of region dissimilarity easy and straightforward. Based on the region dissimilarity, we obtain each region's best-matched regions among the frames, and establish the correspondences among them accordingly. High matching accuracy is achieved with the integration of various effective region features. The technique is automatic and provides a robust solution for matching regions in 2D animation characters.

Based on the proposed feature-based region matching algorithm, we propose an auto coloring approach which covers the entire painting process in traditional 2D animation production. The relationship among different character parts in 3D space is made use of by extracting the stable topology from 2D frames, without constructing a 3D model.

This method requires no change in animators' traditional drawing approach, and avoids the inaccuracy due to insufficient details from a rough 3D model, or the high cost of constructing a precise and detailed 3D model. To increase the accuracy of region matching, a *Hierarchical Region Matching (HRM)* approach, expanding the hierarchy for 2D animation characters, is proposed by us. In addition, we propose the method of coloring animation frames based on a set of master frames. The character in the first frame of a sequence is thus able to be colored automatically, based on the color information of characters in master frames. Integrating the above achievements, we propose our auto coloring approach, which can color characters in animation frames with high accuracy for most cases. A comparison with other existing auto coloring algorithms is also made to highlight the advantage of our approach over them.

Finally, we propose a solution for auto coloring a sequence with occlusion in some frames, as the first attempt to handle special cases. Based on our novel character registration method, we segment the character into components, and detect and locate occlusion in certain components. We then apply different matching methods for components with and without occlusion respectively. Inaccurate matching due to the occlusion is thus partially tackled, making the auto coloring approach more robust. The approach requires minor user-intervention, and knowledge-based coloring rules may further be introduced based on the character registration. In summary, it provides a basic foundation for further investigation on handling special cases.

List of Figures

1.1	Master frames	1
1.2	Region color (courtesy of Anime International Co., Inc.)	3
1.3	Term illustration	4
2.1	Traditional 2D animation production process	9
2.2	Model sheet (courtesy of Walt Disney Animation Canada Inc.)	10
2.3	Cleanup over a light box	12
3.1	Region	22
3.2	Noise-reduction	23
3.3	Line-thinning and Gap-closing	24
3.4	Image expansion	25
3.5	Masks for region identification	26
3.6	Region segmentation and labelling: Method 1	26
3.7	Scan line seed fill algorithm	28
3.8	Region segmentation and labelling: Method 2	29
3.9	Image for segmentation speed testing	29
3.10	Antenna and badge	31
3.11	Instability of junction points	32
3.12	The contour of a dolphin	35
3.13	The curvature graph	35
3.14	Character points	35
3.15	Character points detected by k -cosine measure	37
3.16	Contour change (courtesy of Anime International Co., Inc.)	37
3.17	Representation of relations	38

3.18	Small change in area of the arm	40
3.19	Combination of relations with neighboring regions and character points .	41
3.20	Contour detection	42
3.21	Inner region detection	44
3.22	Image cleaning	45
3.23	Character points detection	46
3.24	Optimization for character points detection	47
3.25	Isolated regions	53
3.26	Shape-independent weights versus shape-dependent weight	54
3.27	Variation of λ_a	55
3.28	Variation of λ_{cl}	56
3.29	Test 1: angel (resolution: 208×224 pixels)	57
3.30	Test 1: Region segmentation	58
3.31	Test 1: Character points	59
3.32	Test 2: stomach (resolution: 351×401 pixels)	60
3.33	Test 2: Region segmentation	61
3.34	Test 3: pig (resolution: 517×584 pixels)	62
3.35	Test 4: rabbit (resolution: 316×486 pixels)	63
4.1	Layer separation (courtesy of Anime International Co., Inc.)	66
4.2	Using 3D character to assist 2D animation production (courtesy of Ma- hendra Sidharta Suryadi)	68
4.3	Stable topology	70
4.4	Traditional 2D animation hierarchy	71
4.5	Expanded hierarchy of region level	72
4.6	A character construction process	73
4.7	Character approximation	73
4.8	Convex hull	74
4.9	Hierarchization result	76
4.10	Grouping method 1: expanded hierarchy	77
4.11	Diversity $D_{CH}(i, i')$	77

4.12	Hierarchization result	78
4.13	Grouping method 2: expanded hierarchy	79
4.14	Segmented components for master frames	81
4.15	Segmented components for a sequence of frames	82
4.16	Stable relation judgment	84
4.17	Extracted stable topology	85
4.18	Inherited stable topology	86
4.19	Recomposition rule	87
4.20	Topology recomposition	88
4.21	Region matching process	89
4.22	Topology verification	90
4.23	Postprocessing	90
4.24	Test 1: Input images (resolution: 778×700 pixels)	92
4.25	Test 1: Results	93
4.26	Test 2: Master frames (resolution: 550×1100 pixels)	94
4.27	Test 2: Results (resolution: 998×892 and 988×1240 pixels)	95
4.28	Test 3 (resolution: 870×825 pixels)	96
4.29	Test 4 (resolution: 1168×826 pixels)	98
4.30	Analysis for Test 2	99
4.31	Coloring results comparison	100
4.32	Test 5 (resolution: 506×620 pixels)	101
4.33	Test 6 (resolution: 504×652 pixels)	102
4.34	Test 7 (resolution: 1500×914 pixels) (courtesy of Anime International Co., Inc.)	103
4.35	Error analysis for Test 7	104
4.36	Expression maps (courtesy of Anime International Co., Inc.)	106
4.37	Limitation of the statistical feature	107
4.38	Coloring results using our algorithm	107
4.39	Coloring results using the algorithm proposed by HS Seah and F Tian . .	108
5.1	Occlusion	110

5.2	Skeleton estimation ($\rho = 0.8, \kappa = 0.5$)	115
5.3	Extracted branches	116
5.4	Extracted topology graph	116
5.5	General topology model	117
5.6	Skeleton registration	117
5.7	Wrong matching	118
5.8	Geometric matching	119
5.9	Registration based on geometric matching	120
5.10	Subgraph and sub-skeleton	121
5.11	Topological re-adjustment	122
5.12	Registration based on topological re-adjustment	122
5.13	Component segmentation	123
5.14	Segmentation results	124
5.15	Change in regions	126
5.16	Auto coloring process	127
5.17	Test 1 (resolution: 768×576 pixels)	129
5.18	Error analysis for Test 1	130
5.19	Test 2 (resolution: 484×1084 pixels)	131
5.20	Test 2: Component segmentation for the left leg	132
5.21	Test 3 (resolution: 570×410 pixels)	133
5.22	Error analysis for Test 3	133
5.23	Occlusion within a component	135
5.24	Information loss	136
6.1	Test 1 in Chapter 3	141
6.2	Test 1 in Chapter 5	141
6.3	Conventional relocation method in inbetween drawing	144
6.4	Another type of model sheet (courtesy of Warner Bros.)	146
6.5	Different ways of coloring	146
6.6	Shot types	147

List of Tables

3.1	Coding rules	50
3.2	Conversion costs	51
3.3	Test 1: Region quantization	58
3.4	Test 1: Matching region 3 in Frame 2	59
3.5	Test 1: Matching region 4 in Frame 1	59
3.6	Test 2: Matching region 10 in Frame 3	61

Chapter 1

Introduction

1.1 Motivation

Figure 1.1: Master frames

Traditionally, 2D animation (or “cel animation”) production is a labor-intensive artisan process of building up animated sequences all by hand. The process consists of the following major tasks: scripting, story boarding, character model sheets design, exposure sheets, drawing, inking/coloring and compositing [61]. After a story is split into scenes, characters are drawn in various poses as *master frames* on model sheets, as illustrated in Figure 1.1. The master frames are drawn in different view angles, which usually contains the front, side and back views of a character, defining the main characteristics of individual characters. Exposure sheets are then produced to give animators detailed instruction for drawing, compositing and shooting of each frame. Next, the main animators draw

CHAPTER 1. INTRODUCTION

some extreme frames showing the major features of each character's action. Assistant animators then draw key frames to break down the action between extreme frames, whereas junior animators produce inbetweens to yield a smooth animation. The drawing of the background is carried out at the same time. All these frames are consequently transferred onto transparent sheets of celluloid (cels) by xerography, and frames of different layers are painted and combined into composited frames according to the exposure sheets. Eventually, an animation film is produced by shooting the composited frames at a certain rate, usually 24 frames per second. Among the entire production process, most work and time is spent on two tedious tasks: drawing and inking/coloring of the individual animated characters for each frame, which take up approximately 60% of total labor required in traditional 2D animation [32]. Introducing computer assistance in traditional cartoon production, the workload of the animators can be reduced, production cost lowered and time saved. This is the motivation of our research.

1.2 Objective

The objective of *Computer-Assisted Cel Animation (CACAni)* is to design and apply computer graphics and imaging techniques to develop a generic 2D animation system which can be applied to the entertainment industry and multimedia production. The aim of this project is to develop cutting-edge technique that aims at optimizing the image-related working process by introducing computer support in the painting stage of traditional cartoon production. In this project, we are researching on a computer method to mimic the way human applies colors to the character in each frame. Auto coloring approaches are to be proposed to color the character in each frame correctly.

In the coloring stage of traditional 2D animation production, master frames are sent to animators for them to color the characters as they appear in any postures in a sequence of frames. A character is composed of a set of closed regions where the color is filled. As

Figure 1.2: Region color (courtesy of Anime International Co., Inc.)

illustrated in Figure 1.2, each region is filled in a flat color. As one frame is composed of many layers, artistic effects like shading can be created in a separate layer. Accordingly, each region of the character is first painted with a flat color which will subsequently be composited with its shadow layer. Animators first establish correspondences among regions in master frames and uncolored line drawings. Subsequently they color regions separately according to the correspondences.

In traditional cel animation, the various postures of a character lead to various shapes of regions which represent the same part of the character in different frames. Hence it is difficult to quickly identify correct correspondences among regions. Even for two successive frames, the change between the characters in them may be very big due to fast motion. A reliable algorithm should be able to deal with the difficulty mentioned above.

1.3 Overview of our approach

Before introducing our approach, some terms are defined here for the thesis, as illustrated in Figure 1.3. When matching a region from one frame with those in the other one or several frames, the region to be matched is defined as a *Target Region (TR)*, and those regions which are matched with TR are defined as TR's *Candidate Regions (CRs)*. The frame which contains TR is defined as a *Target Frame (TF)*, and the frame which contains TR's CRs is a *Reference Frame (RF)*. If TR i is matched to a CR i' , the region i' is treated as the region i 's *best-Matched Region (MR)*, and the correspondence between them is recorded as $i \rightarrow i'$. Two regions i and j are defined as *related regions* if one is neighboring to or inside the other. And the relation between them is represented as $\langle i, j \rangle$.

Figure 1.3: Term illustration

A feature-based region matching algorithm is proposed based on the dissimilarity between regions concerning their features. After the line drawings are scanned into the computer, several preprocessing steps are applied: noise-reduction, line-thinning and gap-closing. After preprocessing, the character in each frame is segmented into a set of regions. Subsequently, color, area, curve length, character points and relations with

neighboring regions are selected as the region features that are to be matched. Feature Dissimilarity Values (FDVs) of one feature or combined features are computed to represent dissimilarities of designated features between regions. Based on these FDVs, Total Dissimilarity Values (TDVs) which represent dissimilarities of all features between regions are acquired using the method of least-squares. Each region i is matched to the region i' with the minimum value of $TDV(i, i')$.

Based on the proposed feature-based region matching algorithm, an auto coloring approach is proposed to cover the entire painting process of traditional 2D animation production. Stable topology is investigated first, which makes use of 3D information to assist 2D region matching. A *Hierarchical Region Matching (HRM)* approach is then proposed with its applicability analyzed, expanding the region level of traditional 2D animation into several component levels by a convex-hull-based hierarchization method. Subsequently the auto coloring is introduced as a three-stage process. At the first stage, regions in master frames are matched to establish the correspondences among them. According to the established correspondences, a stable topology is extracted. At the second stage, regions in the first frame of a sequence are matched with those in all master frames, and the colors are inherited from their best-matched ones. A new subset of stable topology is hence extracted from the target frame, with reference to the stable topology of master frames and the established correspondences. At the third stage, regions in each frame are matched with those in the prior frame by applying the proposed HRM approach. The stable topology is first recomposed for each component level. Subsequently, components and regions are matched from the first to the last component level. Color information and a new subset of stable topology is then inherited at each frame. The stable topology is updated and propagated in this way, assisting the matching at each frame.

In some cases, occlusion exists in certain frames of a sequence. The occlusion can greatly influence the region features and hierarchization results, thus resulting in wrong

CHAPTER 1. INTRODUCTION

region matching. To detect and handle the occlusion, an approach is proposed with the assistance of character registration. First some general topology models are manually defined. Then the skeleton of each character is obtained based on the evolution of the active contour, and the topology graph is also extracted. With the skeletons, a frame which has the complete branches as the general topology model is selected as the first reference frame and colored. Each branch of the skeleton and topology graph in this frame is manually registered with the general topology model. Subsequently, skeleton in the selected target frame is matched with the reference frame based on the geometric and topological information. The branches of the skeleton and topology graph in the target frame are registered accordingly. Referring to the registered skeleton, components are segmented at the first component level, each corresponded to a branch in the general topology model. Occlusion is thus detected and located in certain components based on the variation of region areas. Subsequently, two kinds of matching methods are advanced for the components with and without occlusion respectively. Inaccurate matching due to the occlusion is hence tackled, making the auto coloring approach more robust.

1.4 Report organization

Following this chapter, Chapter 2 reviews traditional 2D animation production process, existing 2D animation systems, some image matching techniques for animation frames and several existing auto coloring algorithms.

In Chapter 3, our feature-based region matching algorithm is proposed and discussed in detail. The stable topology and a HRM approach are proposed in Chapter 4, and the auto coloring process is introduced in detail. Chapter 5 describes our latest research on character registration. Applying the approach, occlusion can be detected and handled with minor user-intervention. Inaccurate matching due to occlusion is hence tackled, and

CHAPTER 1. INTRODUCTION

knowledge-based coloring rule can be further introduced to optimize the coloring process in the future.

Conclusion is drawn with some future work discussed in the last chapter.

Chapter 2

Survey

2.1 Animation production process

To develop a 2D animation system that can save time and labor, it is important to understand what are the traditional and current animation production processes and which parts of the work can be assisted by computers. These are presented in this chapter.

The main stages of the animation production process are illustrated in Figure 2.1 and delineated as follows [89, 73]. For higher production efficiency, there is often overlap and even parallelism in practice [73].

Story development In this first stage, the story of the animation is developed and the subsequent stages serve to visualize this story to the viewer in the form of a 2D animation.

Sound track Traditionally, sound track is produced after the story is developed by recording dialogues and music in the prototype form in order to drive the animation timing later. Alternatively, sound track can also be added in the final editing stage.

Model sheet Based on the story and the chosen animation style, model sheets are prepared to guide animators to draw and color characters consistently. Master frames

Figure 2.1: Traditional 2D animation production process

as shown in Figure 1.1 are a part of model sheets. Additionally, the faces of characters are also provided with various expressions or view angles, and the color to be used and shadows to be applied may also be shown in model sheets. Figure 2.2 is a typical model sheet from Walt Disney Animation Canada Inc.

CHAPTER 2. SURVEY

Figure 2.2: Model sheet (courtesy of Walt Disney Animation Canada Inc.)

Storyboard After all the characters are designed and illustrated on model sheets, the story is then transcribed into a storyboard, which is basically made up of a stack of papers with sketchy drawings of the envisaged movie shown on them. Shooting scripts, timing charts and dialogues are also shown in the storyboard.

Leica test Leica test is used as the first attempt to put together a film of the required length. It is filmed from stills corresponding to the storyboard with a soundtrack added to test the timing of the sound, especially for dialogues. The aim of this test is to give animators an impression of the movie so that they can spot the weaknesses and rectify them by redesign.

Scene staging Scene staging is the mapping out of each scene, in particular showing the relationship between the characters and the background elements within the same scene. These relations are clear presentation of ideas which include avoiding anything which might distract the viewer's attention, placing characters, viewpoints, expressions etc., even the timing of a scene, so that the viewer's attention is engaged most directly. Questions on how to break down the scene into layers may also be involved in this stage.

Exposure sheet The information from the storyboard, scene staging and soundtrack is transcribed onto an exposure sheet. An exposure sheet is basically a table which has a row for each frame, giving animators detailed instructions for drawing, compositing and shooting of each frame.

Drawing As mentioned in Section 1.1, extreme frames, key frames and inbetweens are drawn in succession. The process of inbetweening takes a lot of time and labor in the whole animation production. It is also the focal point of most computer-assisted character animation systems [23].

Line test Line test is produced to reveal and correct problems in line drawings. The line drawings, without background, are shot and played back to verify that the movement is correct and that characters are interacting accurately.

Cleanup The rough key drawings obtained after the prior stages are necessary to be cleaned up by tracing. The cleanup animators superimpose a piece of blank paper on top of each rough drawing on the *light box* and then draw clean strokes on the upper blank paper with reference to the underlaying key drawing. A light box, also called the *trace box*, as shown in Figure 2.3, holds a fluorescent light fixture that shines up through a rotatable disk of frosted glass or milky-white plastic. This allows animators

CHAPTER 2. SURVEY

to see through the sheets and easily trace images on a sheet of paper below the one he is working on. The peg bar on the disk helps to register the sheets with corresponding holes.

Figure 2.3: Cleanup over a light box

Inking/Coloring The conventional way in this stage is transferring the clean line drawings onto sheets of celluloid (cels) by xerography. Usually several layers of cels which contain animated elements and the background are produced separately. Then inking/coloring is performed on the cels, each with the right degree of opacity. With the computer assistance, the task of inking/coloring is normally completed using some animation systems or image processing software like Adobe Photoshop after all layers are scanned into the computer. Because the coloring process is very tedious and time-consuming, it is an emphasis of current research of CACAni, consequently, the topic of our research.

Rostrum camera To get the final movie, all layers of each frame are now composited together on the rostrum camera based on the exposure sheets, illuminated and shot at certain rate (for example, 24 frames per second), possibly incorporating photographic special effects as well.

Editing To transform the shot film or the movie file into a final product, postprocessing and editing is applied where the film is assembled, sorted and spliced to polish the final production.

2.2 Existing 2D animation systems

3D animation seems to have drawn more attention than 2D animation nowadays. A lot of 3D animation systems are available such as 3DS Max and Maya. But there are also some computer-assisted systems for professional or amateur 2D animation production. Here we review the coloring modules of some existing 2D animation systems. Other systems like Plastic Animation Paper [2] and CelAction2D [6] are not reviewed in this thesis as they have no coloring module.

The existing 2D animation systems can be classified into three categories according to their coloring capabilities:

Manual coloring The most common coloring function provided by the 2D animation systems is the manual coloring. To color a region, the user must select a color from a palette and point to a pixel in the region. Consequently all pixels within the region can be painted in the same color by some flooding algorithms, like the area flooding algorithm [56] used in TicTacToon [34].

Although animators need not paint all frames using brushes and pigments entirely by hand, painting in this manner is still tedious because the system requires the clicking of the mouse to paint and using the keyboard to change the drawings frame by frame.

Besides TicTacToon by Toon Boom Technologies Inc., Inkwell [58], Adobe Flash [8] and Mirage Studio [4] by Bauhaus Software, Inc. also belong to this category, though they support painting with textures or gradients.

Batch coloring Besides the common manual coloring function, a lot of systems facilitate the painting process with the batch coloring function. Users can define the regions to paint for a sequence of frames by clicking or dragging the cursor, and the painting module flips through all drawings and colors all regions the cursor touched in the same color.

This kind of batch coloring is adopted by US Animation [12], Toon Boom Studio [10] by Toon Boom Technologies Inc., RETAS! Pro [9] by CELSYS, Inc., Toonz [11] by Digital Video S.p.A. (DV), Animation Stand [3] by Linker System and FlipBook Painter [7] by DigiCel Inc. In FlipBook Painter, the batch coloring can be further controlled by the frame numbers. Users can input a number N using a keyboard to limit the batch coloring within the range between the current and following N frames.

In a color animation system based on the multiplane technique [55] developed by Cornell University, the “seeds” (the pixels where the flooding start) to drive the area-filler are interpolated along with the profile line, which is a bit different from the complete user definition in the above systems. However, the seeds in the beginning and ending frames are still needed to be defined manually.

The batch coloring saves a lot of time as some regions are not needed to be painted frame by frame. But for some regions which have big translation or variation, or overlap with other regions in some frames, users may encounter difficulty to determine a suitable dragging path or proper seed positions.

Limited auto coloring Animo [5] produced by Cambridge Animation Systems has an “Autofill” function which can fill similar regions over a range of frames after the first frame is colored. It works best if the number of regions remains the same and the distance between the corresponding regions from frame to frame is not “huge”. The main problem of Autofill is that it can only handle small changes between regions and

is influenced greatly by the numbers of regions in different frames. If the frame to be colored has more regions (because of occlusion or other reasons) than the first colored frame, some regions will be left uncolored. So it is viewed as a kind of limited auto coloring.

As reviewed above, no or only limited auto coloring function is provided in existing 2D animation systems. Hence, coloring remains tedious. Investigation should be further done to find a suitable auto coloring algorithm, which is able to save more time and labor in the traditional 2D animation production.

2.3 Image matching techniques for animation frames

In 2D animation, characters are not simply the projection of 3D character models. Animators, for aesthetic reasons, deliberately break the rules of geometry and physics as they apply to real-world objects to exaggerate reality [73]. Consequently, matching of animation frames are more difficult than camera-acquired images because the geometric and photometric information in animation frames do not have the same integrity as in the live images. The existing techniques for matching animation frames can be roughly classified into two groups: vector-based and raster-based.

Vector-based techniques first vectorize the strokes in a cartoon drawing with various curve fitting approaches. A collection of vertices (sample points) and vectors (short line segments) or curves obtained by vectorization forms a high-level description of the drawing, and is used by various kinds of matching processes that follows. These techniques usually need users to indicate the control points, thus divide contours of regions into strokes. Kort [50] proposes a matching algorithm which analyzes and classifies each drawing into vectorized strokes, chains of strokes and relations that hold among them. Rules are defined to decide which parts of different drawings may be matched. A cost function based approach determines the correct matching of strokes. Though Kort's

CHAPTER 2. SURVEY

method doesn't require users to indicate control points, users still need to interactively choose the best grouping result for stroke chains. A high matching accuracy is not guaranteed with high image complexity, which requires indispensable manual correction.

For animation frames with complicated strokes, it is quite hard to obtain an optimal vectorization result. This is because the stroke junctions usually break a stroke drawn by the artist into several strokes in the vectorized result. Even if an optimal vectorization and accurate stroke matching is available, another matching process is still needed to establish the correspondences among regions from those among strokes, as a region is usually closed by one or more stroke(s) or stroke segment(s). Since a high accuracy in stroke matching is not guaranteed, the error in stroke matching may lead to wrong region matching. Due to the difficulty, inevitable user intervention and uncertain accuracy in the vectorization and stroke matching processes, vector-based techniques are not adopted in our auto coloring research.

Raster-based techniques do not require drawings to be vectorized. Instead, they usually need various image features to be extracted from the drawings and then establish correspondence by matching the features. The correspondence can be between points, strokes, and also regions.

One category of raster-based matching techniques is the continuous approach based on the optical flow field [46, 66, 92]. Many image matching algorithms have been implemented in the field of computer vision to solve the correspondence problem in stereovision and motion analysis. As images in those studies are usually acquired by cameras, geometric and photometric information could be explored and applied for matching. However, for hand-drawn cartoon line drawings, no photometric information is available and consistent geometric information cannot be assured as well [94].

The other category of raster-based matching techniques is the feature-based matching. This technique treats the image as samples of the scene taken at discrete time, and selects

discrete features as tokens that are to be matched, allowing either small motion or large motion. If various independent features are used for matching, the approach is not restricted to line drawings, but can be applied to match virtually any kind of images. Xie [94] develops a technique that tries to establish the correspondence of feature points by comparing their characteristics such as curvature and distance between two neighboring feature points. Though it is used for generating inbetweens, the selection and matching of feature points offers valuable ideas that might be incorporated into our research, which will be covered in Section 3.2.1.

After studying the above two groups of image matching techniques, we choose feature-based matching to match regions at a high level, and the chosen features will be discussed in Section 3.2.1. In the next section, we review three auto coloring algorithms, which are feature-based matching techniques.

2.4 Auto coloring algorithms

Three auto coloring algorithms, which can color successive frames according to the color information of their previous frames, are discussed here.

JS Madeira et al. [61] propose an auto coloring algorithm based on shape dissimilarity of regions. Sobel operators are applied to estimate the gradient directions of each pixel at the region's contour, subsequently a measure of the curvature variation at each contour pixel is obtained by computing the difference between two succeeding gradient direction values. Based on the measurement of the curvature variation, each contour description is then coded into a string by quantization. Then, the shape dissimilarity of two regions are measured by comparing their respective shape coding strings. Some heuristics are used to avoid unnecessary comparisons. Hence, the least conversion cost of two strings is computed. This represents dissimilarity of their corresponding regions using the string comparison algorithm mentioned in [59]. Finally, region assignment is

CHAPTER 2. SURVEY

made to construct correspondences between each region in the reference frame and its best-matched region in the frame to be painted. This is done by selecting the region with the least-valued conversion cost and coloring regions in the same color of corresponding regions in the reference image. Some ideas such as coding contour description into strings and comparing strings to evaluate the dissimilarity between two regions are adapted into our approach as presented in Section 3.2.2.3.

CW Chang and SY Lee [24] present their automatic cel painting method in computer-assisted cartoon production using similarity recognition. Statistical and topological features of regions are extracted using an integrated drawings labelling algorithm and represented by statistical region feature vector and topological attribute graph respectively. Statistical region features include area, bounding box, aspect ratio and density in this approach. After all attributes are obtained, the matching procedure is performed progressively in three levels: Region comparison, Graph matching and Alternation. To find a best matched reference region, the matching process goes from the top to the bottom level and terminates when the region is obtained. Initially, the first level of region comparison is carried out. Qualified candidate region pairs are determined first with the condition that the distance for every statistical feature is within the accepted threshold. Consequently, the overall deviation is calculated based on all of the differences of statistical features, and the region pair with the smallest overall deviation value among qualified candidate region pairs is regarded as the most similar one. Hence the color of the most similar corresponding region in the previous reference drawing is inherited if all of the measured statistical region features are within the accepted thresholds. To match remaining unmatched regions at the second level, a graph-matching algorithm is proposed using the statistical and topological information of the surrounding region, which consists of three parts: Candidate finding, Mapping generation and Cost calculation. Eventually, a background color is used as an alternative solution for the rest of unmatched target regions at the third level.

HS Seah and F Tian [80] advance another algorithm based on displacement vector of pixels. This is based on a Modified Feature-Based Matching Algorithm (MFBA) which improves on the original Feature-Based Matching Algorithm (FBA) proposed by Weng et al. [93]. First, features such as intensities, edgeness, cornerness (positive and negative) are extracted from the reference frame and the frame to be colored, then the average displacement vector and hence the orientation and displacement residuals are computed and applied to the reference frame. Thus, each pixel in the reference frame will have a corresponding displacement pixel. A threshold is set to divide all regions into big regions and small ones. The best-matched region of a big region is the region into which the majority of corresponding displacement pixels in the big region fall. The total number of pixels of a small region is small and is therefore not as reliable as that of a big one. The result of applying the same method of matching big regions is not satisfactory and neighboring relationship is thus added to aid the region matching. Finally, correspondences among regions in the reference frame and the frame to be colored are established. Colors are inherited according to the correspondences. The aid of neighboring relationship is very useful and will be applied in our research in a different way. Details will be covered in Section 3.2.1.2 and 3.2.2.3.

All the above algorithms reduce more time and labor than those existing 2D animation systems reviewed in Section 2.2, because they are not simply batch process of color flooding at the same position. Compared with Animo, these algorithms are more intelligent in finding region correspondences. Nevertheless, the common constraint of these three algorithms is that the changes between two frames must be small. Big changes in regions will challenge the algorithms and result in wrong region matching, which will be analyzed in Section 4.5 in detail. In addition, the first frame of each animation scene still has to be manually colored by animators as the last frame of the prior scene and the first frame of the current scene are totally different, thus no suitable colored reference frame

CHAPTER 2. SURVEY

can be found. It is also time-consuming to color these first frames because an animation may have many scenes. Our approaches, which will be introduced in Chapter 4, aim to solve these problems.

Chapter 3

Feature-based Region Matching

As reviewed in Chapter 2, most auto coloring algorithms treat the character as a set of regions and try to establish the correspondences among them. According to the established correspondences, each region's color is inherited from the previous frame. Hence, a robust region matching algorithm should be advanced to establish region correspondences for a pair of frames automatically.

In this chapter, a novel feature-based region matching algorithm is proposed. First the region segmentation process is introduced to segment a scanned line drawing into a set of regions. Subsequently, region features are investigated and our feature-based region matching algorithm is proposed and discussed in detail, from the computation of each *Feature Dissimilarity Value (FDV)* and the *Total Dissimilarity Value (TDV)* to the region matching process. Examples are then designed to validate the robustness of our algorithm. A summary is given at the end of this chapter.

3.1 Region Segmentation

Segmenting an image is usually the first step in accomplishing any form of image recognition or description [18]. In our research, the scanned image is segmented into a set of regions.

CHAPTER 3. FEATURE-BASED REGION MATCHING

3.1.1 Region

Region is usually defined as an area closed by its outer border. Some regions are closed by both outer and inner borders. In 2D animation frames, the character is treated as a set of areas closed by one or more strokes. In our research, region is defined as an area closed by the borders, and the pixels of the borders. As illustrated in Figure 3.1, the region is defined as the enclosed region area (pixels in blue), the outer (pixels in red) and inner (pixels in green) border pixels.

Figure 3.1: Region

3.1.2 Preprocessing

After a hand-drawing is scanned into the computer, some preprocessing is required to remove noise, obtain the stroke skeletons, and close gaps. It is important to avoid introducing error to the subsequent region matching process.

3.1.2.1 Noise-reduction

The scanned grayscale image is first converted to a binary image by thresholding for the thinning process, and each stroke in the image is composed of black pixels (pixels with value of 0 after thresholding). During the scanning process, some white pixels (pixels with value of 255 after thresholding) surrounded by 4 black pixels may be introduced.

Figure 3.2: Noise-reduction

Since a one-pixel-wide line composed of these white pixels is not intentionally drawn by animators, as shown in Figure 3.2, these white pixels are considered as noise. To prevent them from being wrongly segmented into small regions, they are detected and set to be black pixels.

The noise mentioned above will affect the result of region segmentation and is thus removed in the preprocessing stage. Besides the mentioned noise, other noises like some isolated points, stroke segments (open, not closed), blocks (dirty smudges in the drawings) may also be introduced during the scanning or drawing process. However, these noises will not alter the result of region segmentation. So they will be eliminated after the region segmentation process, which will be covered in the image cleaning process of Section 3.2.2.2.

3.1.2.2 Line-thinning

Thinning is an image processing operation in which binary valued image regions are reduced to lines that approximate the center skeletons of the regions [43]. It is a commonly used operation in the preprocessing stage to facilitate higher-level analysis, particularly, extracting features such as curve length and relations with neighboring regions for our

CHAPTER 3. FEATURE-BASED REGION MATCHING

case. Thinning can be achieved by various methods. In our research, *Smallest Segment Assimilating Nucleus (SUSAN)* thinning rules [82] are adopted to extract the one-pixel-wide skeleton of each stroke.

3.1.2.3 Gap-closing

Small gaps of broken lines may be introduced either from the scanning process or from original line drawings due to the nature of the artist's drawing style. These gaps can result in wrong region segmentation, hence leading to incorrect region matching in the later stage. To prevent it, Seah's gap-closing algorithm [79] is applied to bridge broken lines to obtain correct regions.

Figure 3.3: Line-thinning and Gap-closing

Figure 3.3 illustrates an example of a binary image with all gaps closed and all one-pixel-wide stroke skeletons obtained.

3.1.3 Region segmentation and labelling

To segment an image I into regions after preprocessing, two algorithms can be applied. One is the region identification algorithm proposed in [62], and the other, proposed by us, is based on the scan line seed fill algorithm [78].

To prevent the background being segmented into different regions when strokes touch the image boundary, an image with resolution of $M \times N$ pixels is first expanded by 1 pixel at each of the four directions, resulting in $(M + 2) \times (N + 2)$ pixels, as shown in Figure 3.4. After region segmentation and labelling, the resolution of the image is restored.

(a) Original image

(b) Expanded image

Figure 3.4: Image expansion

Method 1 The region identification algorithm proposed in [62] labels the regions in two steps. In the first step, the entire image I is searched row by row, and each non-zero pixel $I(i, j)$ is assigned a non-zero value v , which is chosen according to the labels of the pixel's neighbors defined in Figure 3.5 (a) and (b). v is set to be a non-zero label of $I(i, j)$'s neighbors. If no such label is available, v is assigned a new label. If the labels of

CHAPTER 3. FEATURE-BASED REGION MATCHING

any of the neighbors differ (label collision), the label pair is recorded as being equivalent in an equivalence table. Figure 3.6 (a) shows the labelling result in 4-connectivity after the first step. Pixels with detected label collisions are indicated in green. In this example, v is chosen to be the lowest value from all non-zero labels. In the second step, the whole image is scanned again to resolve the label collisions as shown in Figure 3.5 (c). Pixels are re-labelled according to the obtained equivalence table in the first step, and the result is illustrated in Figure 3.6 (b).

Figure 3.5: Masks for region identification

Figure 3.6: Region segmentation and labelling: Method 1

Method 2 Scan line seed fill algorithm as introduced in [78] can fill pixels in a region from a seed pixel inside the region. The algorithm pushes and pops seed pixel *seed* from a stack S , and the whole region is filled when the stack is emptied.

The first *seed* is pushed into S .

A seed pixel *seed* is popped from S , and the span (a group of contiguous pixels on a single scan line) containing *seed* is filled to the right and left of *seed* until a boundary is found, and two values X_{left} and X_{right} are updated as the extreme right and left pixels in the span.

In the range of $X_{left} \leq x \leq X_{right}$, the scan lines immediately above and below current one are examined. The right pixel of each span which does not contain either boundary or previously filled pixels is marked as a *seed* and pushed into S .

Figure 3.7 illustrates the algorithm for filling a region clearly. All seeds in the stack are indicated in blue, and the top one is indicated in green.

The scan line seed fill algorithm is used to segment each region from an image I by setting the top-left pixel of each region as the seed pixel:

Initialize the label n as 0.

Scan the image I line by line, black pixels are labelled as 0, and if a non-black pixel p which has not been labelled is met, pause the scanning. Increase the value of n by one, and pixels in the whole region containing p are labelled as n based on the scan line seed fill algorithm.

Continue scanning until the bottom-right pixel of I is visited.

Thus, the image I will be segmented completely into a finite set of regions as

$$I = \{R(i), i = 1, 2, \dots, N\} \quad (\text{Eq. 3.1})$$

where N is the total number of regions and i is the label of a region, as shown in Figure 3.8.

CHAPTER 3. FEATURE-BASED REGION MATCHING

Figure 3.7: Scan line seed fill algorithm

Figure 3.8: Region segmentation and labelling: Method 2

Figure 3.9: Image for segmentation speed testing

The two methods are both obtained by scanning the image from the top to bottom, and the label for each region is determined by the top-left pixel within it. So the labelling result is scanning order dependent and not tolerant to the image rotation. However, the ordering and values of these labels will not affect the later region matching process. The pixels in each region are just labelled to distinguish from others.

An image with a resolution of 778×700 pixels as illustrated in Figure 3.9 is used to test the segmentation speed of the two methods. Method 1 takes 10.7 seconds to complete the region segmentation, while method 2 only needs 0.9 second, i.e, 10 times faster. So method 2 is adopted in our research.

3.2 Feature-based Region matching

One of the most challenging steps in motion and structure analysis is to establish image correspondences [93]. Similarly, region matching between two frames is an essential task of auto coloring. With the segmented regions, a feature-based region matching algorithm is proposed in this section.

3.2.1 Features of Region

Several important features of a region are studied for the region matching algorithm to be introduced later. The intention is to make use of as many as possible valuable features for region matching. The best-matched region cannot be determined by comparing one feature only unless they are only simple regions.

3.2.1.1 Curve length and character points

An animator may color a region representing something he has no concept before, for example, a monster's strange antenna or a robot's strange badge, as shown in Figure 3.10. To find the correct correspondences, one important feature is the contour of a region.

In most frames of an animation, though the contour of a region changes, it is still recognizable because there are some characteristic shapes that do not change much to certain range. After a starting point is selected at the contour, the contour can be viewed as a curve and its length can be computed. It is believed that in most cases the curve length only varies in a limited range and remains invariant to the rotation of characters.

Figure 3.10: Antenna and badge

Considering an open eye and an almost-closed eye for instance, the change of their curve lengths is relatively small.

Besides curve length, some points can be selected as feature points for later matching. Xie [94] uses junction points along the envelope of the character as feature points. The similarity between junction points is measured by their curvature and distance. Nevertheless, large deformation or motion of regions, inner strokes and animators' drawing styles may influence regions representing the same part of the character in different frames. This leads to large dissimilarity between the numbers of junction points at the contours of two regions, the curvature and the distance between two corresponding points. Figure 3.11 illustrates the instability of junction points. The number and the position of junction points for regions representing the same part of the angel differ greatly in Frames 1 and 2. So Xie [94]'s feature points is not suitable to be applied for region matching.

Character points are the sharpest peaks of convex points and the sharpest pits of concave points. They are sometimes defined as "corner". Compared with junction points, character points are more stable in animation frames, as they don't change with the variation of neighboring regions or some inner strokes. In other words, the character points are tolerant to the region translation if they are not occluded. To extract character points for our region matching algorithm, a survey of corner detection is conducted to

CHAPTER 3. FEATURE-BASED REGION MATCHING

Figure 3.11: Instability of junction points

provide valuable ideas for our character points detection algorithm.

Corners have long been important features for computer vision research because they are invariant to image translation, rotation, and change of size. They have been used extensively for image matching, pattern recognition, and data compression [67]. There is no uniform definition of “corner”. It is defined to be the point having high curvature on the boundary of a planar object [77], or the point where two edge boundaries meet,

or where the direction of the edge boundary is changing rapidly [67].

A lot of corner detection algorithms have been proposed. They can be divided into two categories: methods which operate directly on the gray scale image, and those which rely on prior segmentation of the image and subsequent analysis of region boundaries.

The advantage of methods in the former category is that their performance is not dependent on the result of preceding image segmentation. However the computational expense is more significant. Harris-Plessey corner detector [44] and the SUSAN method [83] are two typical methods of this category. In [76], corner masks are used to compute the “corneriness” of each pixel, while a different method is used in [93] to compute “corneriness” out of “edgeness”. Other algorithms for corner detection belonging to this category are proposed as those based on measuring the variance of the directions of the gradient of intensity [85] or based on the wavelet transform [53]. Though these methods can be applied to detect corners in line drawings (binary images), they give little additional information on the angle, orientation of corner other than the location. In addition, most of them are global detectors and cannot determine the regions that certain corners belong to, adding difficulty to the matching of these corners. Given these disadvantages along with the high computational expense, these methods are not adopted in our research.

Most methods of the latter category are associated with computation of curvature. The curvature here can be the surface curvature of a gray scale image [91], or boundary curvature of a binary image. As mentioned in the beginning of this section, a 2D curve can be obtained after a starting point is selected at the contour, so the main focus of our character points detection is related to boundary curvature, not surface curvature.

If a curve is continuous up to the second order derivative, it can be represented by $\{f(t), g(t) | 0 \leq t \leq T\}$, where f, g are C^2 class functions on $[0, T]$, and $(f(0), g(0))$ is the starting point. The curvature K of each point can be represented as an inverse of the

CHAPTER 3. FEATURE-BASED REGION MATCHING

radius of the largest circle that is tangent to the curve at the point, so character points can be defined as the points which have much higher ($K > 0$) or much lower ($K < 0$) curvatures than other points. Curvature $\rho(t)$ at $(f(t), g(t))$ can be computed using the following formula [51]:

$$\rho\{t\} = \frac{f''(t) \cdot g'(t) - g''(t) \cdot f'(t)}{(f'(t)^2 + g'(t)^2)^{\frac{3}{2}}} \quad (\text{Eq. 3.2})$$

Figure 3.12 shows the contour of a dolphin. Figure 3.13 and Figure 3.14 are the curvature graphs of the contour. Accordingly, character points shown in Figure 3.14 are D, G, H, I, K, L, N.

As for regions in real animation, their contours can be of any shapes, the assumption of continuous curve cannot be satisfied in most cases. To preserve the continuity, curve fitting can be applied to approximate the contour. Cubic B-spline is used to approximate handwritten characters in [63], which can also be applied to approximate contours of regions. Though the approximated curve is very similar to the original contour, a lot of shape information may still be lost, especially at the points of inflexions. K-spline is proposed by DHU Kochanek in [49] as an alternative method for using cubic interpolating splines in a key frame animation system. It has three control parameters which allow animators to change the tension, continuity, and bias of the splines. Though K-Spline seems suitable for curve fitting, it is computationally expensive to get curvatures at each pixel. Moreover curvatures at key points that divide the whole contour into many cubic splines cannot be obtained correctly. The above shows that curve fitting is not a suitable method because of low accuracy or high computational cost. As curvatures cannot be computed directly from an approximated continuous curve, it has to be estimated using other methods.

A number of approaches have been reported to estimate curvature other than using curve-fitting. Examples are Differential of slopes (DOS) method [54, 70, 69], Constrained Regularization (CR) method [84] and those based on Differential Geometry [31]. There

Figure 3.12: The contour of a dolphin

Figure 3.13: The curvature graph

Figure 3.14: Character points

CHAPTER 3. FEATURE-BASED REGION MATCHING

are also other methods for corner detection such as Cellular Vectorization Method (CVM) [52] and methods based on corner models [72, 33], Curvature Scale Space (CSS) [64, 65] and the maximum a *posteriori* (MAP) & Bayes theory [96, 97]. In addition, F Arrebola et al. have proposed several corner detection algorithms by local histograms of contour chain code [14], by circular histograms of contour chain code [15], by adaptively estimated curvature function [16] and by contour local vectors [17].

Among these techniques for corner detection, we chose the k -cosine measure proposed by CH Teh and RT Chin [88] because it has low computational cost and directly operates on the contour pixels, which is suitable for our discrete pixel-based contour representation.

Defining the k -vectors at a selected point p_i as

$$\vec{a}_{ik} = (x_i - x_{i-k}, y_i - y_{i-k}) \quad (\text{Eq. 3.3})$$

$$\vec{b}_{ik} = (x_i - x_{i+k}, y_i - y_{i+k}) \quad (\text{Eq. 3.4})$$

Then the k -cosine at p_i is

$$\rho_{ik} = \frac{\vec{a}_{ik} \cdot \vec{b}_{ik}}{|\vec{a}_{ik}| |\vec{b}_{ik}|} \quad (\text{Eq. 3.5})$$

Thus,

$$\theta = \cos^{-1}(\rho_{ik}), \quad (\text{Eq. 3.6})$$

where $\vec{a}_{ik}$ and $\vec{b}_{ik}$ are respectively the vectors from selected point p_i to its left k th point and to its right k th point, and θ is the angle between $\vec{a}_{ik}$ and $\vec{b}_{ik}$. Character points are then defined as points with much smaller θ than other points. A threshold can be set to obtain these points and their values of θ .

Figure 3.15 illustrates the character points detected applying our modified k -cosine measure as introduced later in Section 3.2.2.3. The points with $\theta < 90^\circ$ are detected based on each region's border and indicated in red.

Figure 3.15: Character points detected by k -cosine measure

(a) Frame 1

(b) Frame 2

Figure 3.16: Contour change (courtesy of Anime International Co., Inc.)

3.2.1.2 Relations with neighboring regions

In some cases, the contour changes greatly. For example, when the hair is swaying with the wind, the contour of the region representing the hair changes in both features of curve length and character points greatly, as highlighted in red circles in Figure 3.16. In such a situation, it is hard to match the region with only curve length and character points. However, the relation between the hair and the face is preserved. The relations of the region to neighboring ones are thus useful for region matching, and this feature is

CHAPTER 3. FEATURE-BASED REGION MATCHING

tolerant to translation, zooming, and even rotation of regions or characters. To compare relations of two regions, it is important to find a proper representation of this feature.

Figure 3.17: Representation of relations

Relations with neighboring regions can be represented by two methods as discussed in [62]. One is the region adjacency graph, as shown in Figure 3.17 (a), where nodes corresponding to regions and neighboring regions are connected by an arc. The region adjacency graph stores information about the neighbors of all regions in the image explicitly and is easy to access. One attractive feature of this method is that if a region encloses other regions, the part of the graph corresponding to the areas inside can be separated by a cut in the graph [62]. Different arcs or powers can be used to distinguish “inside” and “beside”. The other method is the relational table, where individual regions are associated with their names and other features, as shown in Figure 3.17 (b). This method is appropriate for higher levels of image understanding and efficient for access [62].

But these two methods of representation only record the topology information of regions or regions’ own features. It is enough for matching small regions in Seah’s method [80] because the changes between two successive frames are small so that the position of each region varies slightly and thus keeps relations with neighboring regions stable.

But in our research, as the changes between master frames and from a master frame to a target frame may be very large, the relations may not be always stable. To estimate the dissimilarity of two regions using their relations with neighboring regions, the relative positions of regions along their contours are also important. Therefore, to achieve a more accurate matching, a region's neighboring regions (colored or uncolored) are recorded successively by searching along its contour, clockwise or counter-clockwise. More details will be covered in Section 3.2.2.3.

3.2.1.3 Area and color

The area of a region, that is the total number of pixels in the region, usually gives animators the first concept of the region, a big one or a small one. As mentioned in Section 2.4 when reviewing HS Seah and F Tian's auto coloring algorithm, big regions and small ones are matched using different methods. Though it is not necessary to do so in our algorithm, the area is worthy of studying to choose region features. In traditional 2D animation, the area of a region varies in different frames, but for a big region, the variation of the area is relatively small compared with its total area. Compared with the change of shapes, the area is more stable. As shown in Figure 3.18, compared with the arm's change of shape, the change of the arm's area is relatively very small ($< 7\%$).

Besides area, the color information of neighboring regions is a determinate element in comparing features of relations with neighboring regions when these regions have already been matched. In HS Seah and F Tian's algorithm, color information of neighboring regions is used to aid the matching of small regions, but in our algorithm, it is used as a feature for matching of all regions. Details will be covered in Section 3.2.2.3.

3.2.2 FDV and TDV

To match the regions based on their features, the dissimilarity of two regions is measured as follows.

Figure 3.18: Small change in area of the arm

3.2.2.1 FDV

As explained in Section 3.2.1, features of curve length, character points, relations with neighboring regions, area and color are chosen for region matching. *Feature Dissimilarity Value* (*FDV*) is proposed to define how dissimilar two regions are in certain features.

To achieve more reliable comparison, two features, relations with neighboring regions (colored or uncolored) and character points are combined to get one *FDV*. The reason is that character points can divide the contour into stroke segments, reducing possible matching errors. As shown in Figure 3.19, if only the feature of relations with neighboring regions is considered, the stroke *bcd*, which is the shared edge of regions of the hair and the wing in (a) will be matched to the shared edge of two corresponding regions in (b), which is the stroke *BCDE*. However, the stroke *abcde* in (a) should be matched to the

Figure 3.19: Combination of relations with neighboring regions and character points stroke $ABCD$ in (b). With the introduction of character points when computing FDV, the correct matchings of both $abc \leftrightarrow ABC$ and $cde \leftrightarrow CD$ are obtained, because strokes are segmented by character points a , c and e in (a) and also A , C and D in (b).

Hence, all FDVs used in the algorithm are: FDV of area ($FDV_a(i, i')$), FDV of curve length ($FDV_{cl}(i, i')$) and FDV of relations & character points ($FDV_{rc}(i, i')$).

3.2.2.2 FDV of area and curve length

Figure 3.20: Contour detection

Contour detection The contours of the segmented regions are detected to compute the curve length. The image labelled after region segmentation is scanned twice to extract the contour (outer border) and inner regions (regions inside a bigger one) of each region.

Figure 3.20 illustrates the contour detection process. Figure 3.20 (a) is part of an image with each region represented by a different color. During the first scan, for each black pixel (the black pixel with label 0 in (b) is taken as an example), all 8-connected neighbor pixels (pixels pointed by green and blue arrows) are examined. If two or more labels (red and yellow color) can be found, the central black pixel is treated as a border pixel of corresponding regions (regions in red $i1$ and yellow $i2$) and added into each region's set of border pixels ($B(i1)$ and $B(i2)$).

The obtained set of border pixels $B(i)$ for each region i is composed of both outer and inner border pixels, and these border pixels are not sorted. To reorder these border pixels, the second scan is carried out. The reordered set is only composed of outer border pixels and defined as $C(i)$.

Starting from the top left pixel of the region's contour (the black pixel with label 0 in (b) for region in red), each pixel is recorded into the chain and its 8-connected neighbor

pixels are examined to determine the next pixel to be recorded and examined. The 4-connected neighbor pixels (pixels pointed by blue arrows) are examined first. If there is a black pixel in these 4-connected neighbor pixels which has not been recorded into the chain, it is selected as the next pixel to be examined; if there are two such pixels, one is selected arbitrarily. If there are no such pixels, the same examination is applied to the rest of neighbor pixels (pixels pointed by green arrows) to determine the next pixel to be examined. The iteration of finding next pixel to be recorded and examined stops when the starting point is selected again. The whole contour of each region can thus be traced out in order. As illustrated in (b), the label of each pixel represents its index in the chain of the contour.

Assuming that S is the set of a region's inner border pixels, and S' is the set of outer border pixels of the region's inner regions, S is then a subset of S' , as shown in Figure 3.21 (b) and (c). After the chain of each region's contour is constructed, the residual of $B(i) - C(i)$, i.e. S , is compared with other chains, e.g. $C(i')$ as illustrated in Figure 3.21 (d), which is a part of S' . If there are intersections between them, the region i' is an inner region of the region i , as shown in Figure 3.21 (a).

With the detected contour, the curve length $CL(i)$ of region i is computed as the number of outer border pixels.

Image cleaning As mentioned in Chapter 2, rough drawings are traced by animators in cleanup stage of traditional 2D animation production process. Similarly, noises like isolated points, stroke segments, and blocks as mentioned in Section 3.1.2.1, are cleaned up. Furthermore, points, stroke segments and blocks connected with but not on the region borders are also cleaned up, in order to compute the exact area of each region.

After region segmentation, pixels in each region are labelled in a unique number which denotes the region. So an image can be cleaned up according to the labels and the binary values of the pixels.

CHAPTER 3. FEATURE-BASED REGION MATCHING

Figure 3.21: Inner region detection

For each black pixel p , its 8-connected neighboring pixels are examined. Among these neighboring pixels, if non-black pixels exist and are all labelled as i , p is removed and labelled as i . As illustrated in Figure 3.22, the drawing which is cleaned up loses some details of the character, but it still has enough information for region matching. Those removed details are not considered as a feature for matching because of its high uncertainty.

The area of a region, which is defined as the total number of pixels within the region,

Figure 3.22: Image cleaning

can thus be computed accurately. A set of areas in the image I is obtained as

$$A = \{A(i), i = 1, 2, \dots, N\} \quad (\text{Eq. 3.7})$$

where N is the total number of regions and i is the label of a region.

Computing FDV For area and curve length, FDV of i to i' equals to the difference between them divided by i 's value.

$$FDV_a(i, i') = \frac{|A(i) - A(i')|}{A(i)} \quad (\text{Eq. 3.8})$$

$$FDV_{cl}(i, i') = \frac{|CL(i) - CL(i')|}{CL(i)} \quad (\text{Eq. 3.9})$$

where i and i' are respectively the labels of regions in the target frame and the source frame, $A(i)$ and $A(i')$ are the areas of regions i and i' , $CL(i)$ and $CL(i')$ are their curve lengths.

CHAPTER 3. FEATURE-BASED REGION MATCHING

3.2.2.3 FDV of relations & character points

It's more complex to compute the combined FDV of relations & character points than FDV of area or curve length. The steps are shown as follows:

Finding character points As mentioned in Section 3.2.1.1, the k -cosine measure is used to find character points. Here, k is a key element for finding character points. If k is too big, some character points may be ignored and the result θ may be wrong; if k is too small, the value of θ may be bigger than its actual value, also resulting in some loss of character points. In order to obtain a reasonable value of k , we combine the algorithm of determination of region of support in [88] and the algorithm of corner detection in [17].

Figure 3.23: Character points detection

As illustrated in Figure 3.23, for a point p_i , the length of the chord joining the points p_{i-k} and p_{i+k} is defined as $L(k)$. $C_l(k)$, $C_r(k)$ are respectively the contour lengths from the point p_{i-k} to p_i , and p_i to p_{i+k} , $D_l(k)$, $D_r(k)$ are respectively the lengths of the chords from the point p_{i-k} to p_i , and p_i to p_{i+k} . Three conditions are used to determine k :

$$\frac{C_l(k) + C_r(k)}{L(k)} > \frac{C_l(k+1) + C_r(k+1)}{L(k+1)} \quad (\text{Eq. 3.10})$$

$$L(k) < L(k+1) \quad (\text{Eq. 3.11})$$

$$\begin{cases} \frac{C_l(k) - D_l(k)}{C_l(k)} < 0.1 \\ \frac{C_r(k) - D_r(k)}{C_r(k)} < 0.1 \end{cases} \quad (\text{Eq. 3.12})$$

Start with $k = 1$, repeat $k = k + 1$ until neither (Eq. 3.10) nor (Eq. 3.11) is satisfied or (Eq. 3.12) is not satisfied. (Eq. 3.10) and (Eq. 3.11) are conditions from [88], which ensure the value of k is in an appropriate range. (Eq. 3.12) is proposed in [17], which ensures the difference between the arc and the chord is small enough so that the value of angle θ is appropriate. Once k is determined, θ can then be obtained using (Eq. 3.6) to find character points. Through experiments, character points are selected to be the points having θ no bigger than 90° in our approach. Applying the algorithm proposed above, a section of successive character points may be obtained around the sharpest peaks of convex or sharpest pits of concave points, as shown in Figure 3.24. An optimization method is applied to remove redundant character points by selecting the point with the least value of θ as the correct character point. In Figure 3.24, the green point is selected as the character point as its value of θ (68°) is smaller than that of the red one (76°), which is considered as redundant. The other character points of the region in grey are not shown in Figure 3.24 for a clear illustration.

Figure 3.24: Optimization for character points detection

CHAPTER 3. FEATURE-BASED REGION MATCHING

Region quantization Seah [80] groups all regions into BIG and SMALL and different matching approaches are applied to them. Assuming the change in each region's area among frames is always within a limited range, two regions are not considered matched if the difference in their areas is too big. We can extend big and small regions with more scales. In our approach, regions are quantized into several scales so that the comparison is only carried out within the same scale, thus avoiding unnecessary comparison and emphasizing diversity of regions in different scales.

The variation of region area is assumed to be in general no bigger/smaller than half of its original value, which is true in most cases without substantial zooming. As mentioned in Section 3.1.3, a set of region areas can be obtained as $A = \{A(i), i = 1, 2, \dots, N\}$ where N is the total number of regions and i is the label of a region in an image I . Consequently, a quantization method is proposed as follows:

1. Sort $A(i)$ in ascending order of size. Compute the average diversity value $\bar{d}$:

$$d(i) = A(i+1) - A(i) \quad (\text{Eq. 3.13})$$

$$\bar{d} = \frac{\sum_{i=1}^{N-1} d(i)}{N-1} \quad (\text{Eq. 3.14})$$

that is,

$$\bar{d} = \frac{A(N) - A(1)}{N-1} \quad (\text{Eq. 3.15})$$

where $d(i)$ is the value difference of two neighboring areas.

2. Check $A(i)$ and $A(i+1)$ according to:

$$A(i+1) * (1 - \varphi) > A(i) * (1 + \varphi), \quad (\text{Eq. 3.16})$$

$$\begin{cases} d(i) > \bar{d} \\ A(i+1) > \text{Min}(1 + \varphi, \frac{1}{1-\varphi}) * A(i) \end{cases} \quad (\text{Eq. 3.17})$$

where φ is the presumptive maximum variation of region area, which equals to 0.5 in our research.

(Eq. 3.16) examines the situation when region areas change in two directions, where $A(i)$ is assumed to be increasing and $A(i + 1)$ is decreasing. The two corresponding regions are quantized into two different scales when the value of the increased $A(i)$ is less than the that of the decreased $A(i + 1)$. This is quite rigid as the region change among a sequence of frames is usually small or big (usually due to occlusion) but one-directional. (Eq. 3.17) determines the quantization with average diversity value $\bar{d}$ among all areas. Two regions with $d(i)$ bigger than $\bar{d}$ are viewed as having big distance among them. To regulate them according to the assumption for area variation, one-directional change is considered. If $A(i)$ is increasing, $(1 + \varphi) * A(i) < A(i + 1)$ needs to be satisfied; If $A(i + 1)$ is decreasing, $A(i) < (1 - \varphi) * A(i + 1)$ needs to be satisfied. Hence we arrive at $A(i + 1) > \text{Min}(1 + \varphi, \frac{1}{1 - \varphi}) * A(i)$ in (Eq. 3.17), which ensures that at least one of the two above equations is satisfied.

If either (Eq. 3.16) or (Eq. 3.17) is satisfied, a scale point

$$SP(i') = \frac{2 * A(i + 1) * A(i)}{A(i + 1) + A(i)}, i' = 1, 2, \dots, m; i = 1, 2, \dots, N - 1. \quad (\text{Eq. 3.18})$$

is selected, where m is the total number of scale points, N is the total number of regions, i is the label of a region in the image I , and i' is the label of a scale point. Thus, regions in each frame can be quantized into $m + 1$ area scales in the same measure.

Coding relations & character points To compute FDV of relations & character points, a region's neighboring regions and character points at its contour are coded into a string of integers for later comparison by searching along the contour.

Scales of the area, color and character points are coded as integers. The coding rules are given as follows.

CHAPTER 3. FEATURE-BASED REGION MATCHING

- Area: Regions with their areas coded as 1 are the regions in the smallest scale, $(2^{31} - 1)$ in the biggest one, and the background is coded as 0.
- Color: Each integer represents a different color, where 0 for the background and uncolored regions, and $1, 2, \dots, (2^{31} - 1)$ for different region colors.
- Character points: Points coded as 1 are the points which have the biggest values of θ while $(2^{31} - 1)$ the ones with the smallest values of θ . In our research, only 5 scales are used, so character points coded as 5 are those with the smallest values of θ .

A neighboring region or a character point is coded into an element string σ of 2 integers as listed in Table 3.1.

	First Integer(f)	Second Integer(s)
Uncolored Regions	0	$0..(2^{31} - 1)$
Colored Regions	$1..(2^{31} - 1)$	$1..(2^{31} - 1)$
Character Points	-1	1..5

Table 3.1: Coding rules

Then a character string of integers $S = \sigma_1\sigma_2\dots\sigma_n = "f_1s_1f_2s_2\dots f_ns_n"$ along the contour of the region is obtained, representing relations & character points.

Computing FDV The dissimilarity between two regions can be represented by the difference of their character strings. Thus, FDV of relations & character points is defined as:

$$FDV_{rc} = \frac{\gamma(S \rightarrow S')}{\gamma(S \rightarrow \phi)} \quad (\text{Eq. 3.19})$$

where $\gamma(S \rightarrow S')$ represents the least conversion cost from target string S to source string S' , $\gamma(S \rightarrow \phi)$ represents the conversion cost from target string S to an empty string ϕ .

The string comparison algorithm proposed in [59] is used to determine the least conversion cost. The costs of inserting, deleting and substituting a character (an element

string σ in our approach) are written into a matrix and a minimum cost path through the matrix defines the best sequence of transforming the origin string into the target string.

As mentioned in the prior section, there are three types of elements to be compared in the character string. So the determination of conversion cost from one type to another is the key point of string comparison. Assume the cost of deletion or insertion of uncolored regions as V_{ur} , colored regions as V_{cr} and character points as V_{cp} , and V_{cp1} and V_{cp2} as the costs of deletion or insertion of two different character points. Consequently, if two element strings are not identical, the conversion cost can be represented as shown in Table 3.2.

	Uncolored Regions	Colored Regions	Character Points
Uncolored Regions	$2 * V_{ur}$	$V_{ur} + V_{cr}$	$V_{ur} + V_{cp}$
Colored Regions	$V_{cr} + V_{ur}$	$2 * V_{cr}$	$V_{cr} + V_{cp}$
Character Points	$V_{cp} + V_{ur}$	$V_{cp} + V_{cr}$	$ V_{cp1} - V_{cp2} $

Table 3.2: Conversion costs

Based on experiments, we set the conversion costs as $V_{ur} : V_{cr} : V_{cp} = 1 : 5 : 2 * P$, $P = 1, 2, 3, 4, 5$, representing different angle scales of character points. For example, if an element string is "05", $P = 5$.

Direction and the starting point of the string are also considered in our approach to obtain the least conversion cost. So the shorter string $\sigma_1\sigma_2 \dots \sigma_n$ is selected as the origin string, $\sigma_k\sigma_{k+1} \dots \sigma_n\sigma_1 \dots \sigma_{k-1}$ and $\sigma_k\sigma_{k-1} \dots \sigma_1\sigma_n \dots \sigma_{k+1}$, $k = 1, 2, \dots, n$ are then compared with the target string $2n$ times in total, and the least value as the least conversion cost is obtained at the end.

3.2.2.4 TDV

A region has several features, each of which plays different role in region matching. The *weight* (λ) that reflects the relative importance of each feature over the others is an important parameter to find the best-matched region. This is proposed by Seah in

[80]. Therefore, *Total Dissimilarity Value (TDV)* is defined as follows to represent how dissimilar two regions are, according to all the features using the method of least-squares:

$$TDV(i, i') = \lambda_a * FDV_a(i, i')^2 + \lambda_{cl} * FDV_{cl}(i, i')^2 + \lambda_{rc} * FDV_{rc}(i, i')^2 \quad (\text{Eq. 3.20})$$

where i and i' are respectively the labels of regions in the target frame and the source frame, FDV_a , FDV_{cl} and FDV_{rc} are the FDVs of area, curve length, and relations & character points respectively, while λ_a , λ_{cl} , λ_{rc} are their weights.

3.2.3 Region matching

As introduced in Section 1.3, to match regions from one frame (TF) to the other (RF), each region in the prior frame is selected as a TR, and its CRs are selected as all regions in the same area scale in RF. Each TR i is matched with its CRs by computing the $TDV(i, i')$. After the matching, the minimum value $TDV(i, i')$ is obtained, and i is i' 's MR. To avoid two corresponding regions from being wrongly quantized into two different scales, a threshold is empirically set. If $TDV(i, i')$ is bigger than the threshold, the region i and i' are not regarded as correspondent. After the correspondence between i and i' is established, regions i and i' are called *matched regions*, while the remaining regions are *unmatched*.

First, regions in the target frame are bi-directional matched with those in the reference frame and some regions are matched and color is inherited. A region i in the target frame obtains its best-matched region i' in the reference frame, which is recorded as $i \rightarrow i'$. Subsequently each region in the reference frame is matched with its CRs in the target frame. For example, $i' \rightarrow j$. The correspondence between two regions i and i' is established only if i is identical to j . With the bi-directional matching, those regions which are not changing much between frames are matched and colored first, and the color information is inherited according to the correspondence. The inherited color will

be coded in the character string, assisting the following region matching process. Subsequently the remaining unmatched regions in the target frame are matched with those in the reference frame, where the matching is one-directional. After region matching, each region i inherits the color of its MR i' in the reference frame. The character in the target frame is hence painted.

Figure 3.25: Isolated regions

To make use of the color information of those matched regions, the matching steps are proposed as following:

1. Select a region which has the biggest area as TR and match it.
2. New TR is selected as an unmatched region which has most neighboring matched regions, and matched iteratively.
3. Check all regions to find unmatched isolated regions as shown in Figure 3.25, and match these regions according to the order defined in Steps 1 and 2.
4. Match the remaining unmatched regions with the CRs in neighboring area scales.

3.3 Results and Analysis

As introduced in Section 3.2.2.4, a set of weights $\mathcal{L}_t = \{\lambda_a, \lambda_{cl}, \lambda_{rc}\}$ is needed to be set for the computation of TDV.

CHAPTER 3. FEATURE-BASED REGION MATCHING

The set of weights for the various features has three parameters. In order to reduce the combinatorial effect of determining the optimum set of the three parameters, the parameters can be represented by two groups whose weights α and β are to be determined. The first group is shape-independent, which represents λ_a and λ_{cl} . The second group which represents λ_{rc} is shape-dependent. This yields the set of weights $\mathcal{L}_t = \{\alpha, \alpha, \beta\}$.

Figure 3.26: Shape-independent weights versus shape-dependent weight

An experiment is set up to determine the relative weight between the two groups of features. In order to study the impact of the weight, we set $\alpha = 4$ with β varying from 1 to 30. Figure 3.26 depicts the changes in region matching accuracy with respect to increasing β (shape-dependent weight) for several sequences as shown in the later tests. Each frame is matched with other frames in the sequence and an average matching accuracy is obtained as the value shown in Figure 3.26. The four regions representing

the toes of the angel are quite similar as shown in Figure 3.29, the swapping error, as explained later in the analysis for Test 1, is ignored in the test of angel-1, while being considered in angel-2. The overall trends for the accuracy variation are quite similar for all examples except angel-2. After the initial increases, the accuracy values reach their respective plateau at different values of β . As β keeps increasing, the accuracy will drop because the weight of relations & character points begins to overwhelm the others. Judging from the results, $\beta = 16$ gives the optimum result for all the examples. This yields a set of weights $\mathcal{L}_t = \{4, 4, 16\}$.

A second experiment is set up to examine the effect of each feature in the shape-independent group. In this experiment, λ_{rc} is kept constant while λ_a and λ_{cl} are varied.

- Area.

Figure 3.27: Variation of λ_a

CHAPTER 3. FEATURE-BASED REGION MATCHING

λ_a is varied from 1 to 16 while the other two weights remain as $\lambda_{cl} = 4$ and $\lambda_{rc} = 16$. From Figure 3.27 we can see the variation of accuracy is small compared with those in Figure 3.26. The reason is that the feature of relations & character points captures the most important characteristic of the region in most cases. λ_a is selected as 4 accordingly, and $\mathcal{L}_t$ remains as $\{4, 4, 16\}$.

- Curve length.

Figure 3.28: Variation of λ_{cl}

With the fixed values $\lambda_a = 4$ and $\lambda_{rc} = 16$, the accuracy variation due to the change of λ_{cl} is quite small as depicted in Figure 3.28. Correspondingly, the final set is chosen as $\mathcal{L}_t = \{4, 4, 16\}$, which is $\lambda_a : \lambda_{cl} : \lambda_{rc} = 1 : 1 : 4$.

With the chosen weights, several tests are applied for coloring a sequence of frames.

The first frame of each sequence is manually colored, and the regions in each frame are matched with those in the prior frame to maximize the matching accuracy.

Figure 3.29: Test 1: angel (resolution: 208×224 pixels)

Figure 3.29 illustrates the result of our first test. (a) is the first colored frame, (b) and (c) are the two uncolored frames, and (e) and (f) are the coloring results using our feature-based region matching algorithm. The big changes between any two successive frames can be noted, especially the arm and the wing. Part of the matching process from Frame 1 to Frame 2 is analyzed here. All regions in Frames 1 and 2 are first segmented and labelled, as illustrated in Figure 3.30.

The area scale points are then determined using the proposed region quantization method introduced in Section 3.2.2.2. For this example, the values of scale points are

CHAPTER 3. FEATURE-BASED REGION MATCHING

Figure 3.30: Test 1: Region segmentation

computed as 123, 409 and 1014. So the regions are quantized into 4 scales, as illustrated in Table 3.3.

Area	a	b	c	d
Frame 1	2,6,7,8,9,13,14,15,16	5,12	4,10,11	1,3
Frame 2	2,6,7,8,9,13,14,15,16	5,12	3,10,11	1,4

Table 3.3: Test 1: Region quantization

Region 4 is the biggest region in Frame 2. It is first matched with regions 1 and 3 in Frame 1, and bi-directionally matched to region 3. Subsequently region 1 in Frame 2 is bi-directionally matched to region 1 in Frame 1. Then region 3 in Frame 2 is selected as the new TR because it is neighbor to both the matched regions 4 and 1. Figure 3.31 shows the character points detected in the two frames, where the character points labelled in red, green and blue are coded as 01, 02 and 03 respectively. So region 3 in Frame 2 is coded as “3 4 -1 2 0 2 0 1 0 1 0 2 -1 1 0 0 1 4 -1 2 1 4 -1 3 1 4 0 1 0 1 -1 2 0 1 3 4 -1 1 3 4” and matched with regions 4,10,11 in Frame 1. The computed FDVs and TDVs are listed in Table 3.4 and 3.5.

From the tables we can see, region 3 in Frame 2 is bi-directionally matched to region

Figure 3.31: Test 1: Character points

Candidate Region	FDV_a	FDV_{cl}	FDV_{rc}	TDV
4	0.071629	0.016667	0.348485	0.491175
10	0.356742	0.205556	0.848485	3.049224
11	0.422753	0.227778	0.893939	3.427113

Table 3.4: Test 1: Matching region 3 in Frame 2

Candidate Region	FDV_a	FDV_{cl}	FDV_{rc}	TDV
3	0.077156	0.016393	0.553846	1.233204
10	0.236006	0.191257	0.815385	2.751686
11	0.216339	0.278689	0.876923	3.200446

Table 3.5: Test 1: Matching region 4 in Frame 1

4 in Frame 1. A new TR is selected and matched iteratively, and finally all regions are matched and colored.

In Test 1, most of the regions are correctly colored, yet there are still some errors, like the toes with yellow and red swapped in Figure 3.29 (e) and (f), and the color of two feathers swapped in (f). This is because these wrong-colored regions have high similarity in all features. Our algorithm ignores a region's actual coordinates/locations in the frame but considers the topology and relative positions along the contours of regions, in order to be tolerant to the translation and rotation of the character. Such errors are hard to

CHAPTER 3. FEATURE-BASED REGION MATCHING

avoid but can be corrected with minor user-intervention.

Figure 3.32: Test 2: stomach (resolution: 351×401 pixels)

Figure 3.32 shows our second test with some isolated regions in each frame of the sequence. Among a total 75 regions in the three target frames, 71 regions are correctly colored.

Figure 3.33 illustrates the region segmentation results for Frames 2 and 3. Table 3.3 lists the computed FDVs and TDV when matching region 10 in Frame 3 with its candidate regions in Frame 2. From Table 3.3 we can see region 10 in Frame 3 is matched to region 16 in Frame 2 as $TDV(10, 16)$ is the minimum. Subsequently regions 14 and 15 in Frame 3 are both matched to region 19 in Frame 2. The reason is that the artist separates the fingers into different regions in this frame, while in the other frames all fingers are in the same region with the hand. Without a correct corresponding region in

Figure 3.33: Test 2: Region segmentation

Candidate Region	FDV_a	FDV_{cl}	FDV_{rc}	TDV
1	0.834416	0.536232	1.923077	15.776692
2	0.879870	0.536232	1.000000	5.061716
6	0.805195	0.579710	2.384615	23.729965
7	0.779221	0.550725	1.615385	11.348351
11	0.509740	0.217391	2.538461	26.082241
15	0.480519	0.072464	1.076923	4.875204
16	0.412338	0.188406	0.692308	2.122679
19	0.827922	0.579710	1.000000	5.021519
20	0.698052	0.347826	0.769231	2.975123
21	0.844156	0.347826	0.769231	3.200446
22	0.957792	0.710145	0.769231	3.788536
23	0.889610	0.478261	0.769231	3.387004
24	0.840909	0.173913	0.769231	3.104238

Table 3.6: Test 2: Matching region 10 in Frame 3

the prior frame, these regions are matched to others, as shown in Figure 3.32 (f). The region which represents the tongue in Figure 3.32 (g) is wrongly colored due to the same reason. This kind of error can be solved if more reference frames are available, where all regions corresponding to those in the target frame are provided. An alternative way is

CHAPTER 3. FEATURE-BASED REGION MATCHING

to introduce artificial intelligence or image understanding techniques to our algorithm in the future.

Figure 3.34: Test 3: pig (resolution: 517×584 pixels)

Two other tests are depicted in Figure 3.34 and 3.35. As the change among the sequence is small, a high matching accuracy is achieved. The error in Figure 3.35 (f) is due to the high dissimilarity of two regions representing the tail in Frame 1 and 2. The coloring error is propagated to Frame 3 and 4 as shown in (g) and (h) though the regions representing the tail are correctly matched.

3.4 Summary

Most auto coloring algorithms treat the character as a set of regions and try to establish correspondences among them. Accordingly, to achieve the aim of coloring the characters

Figure 3.35: Test 4: rabbit (resolution: 316×486 pixels)

in a sequence of animation frames automatically, a novel region matching algorithm is proposed in this chapter. After an animation frame is scanned into the computer, the character is segmented into a set of regions. Different region features like the area, color, curve length, character points, and relations with neighboring regions are studied. And FDVs of these features are calculated. With the FDVs, TDV is computed to represent how much two regions differ from each other. For a region i , the region i' with the least $TDV(i, i')$ is its best-matched region. According to the TDV, regions in the target frame is first bi-directionally and then one-directionally matched with those in the reference frame. Correspondences among regions in the reference and target frames are hence established, and colors are inherited. The results have shown the robustness of our region

CHAPTER 3. FEATURE-BASED REGION MATCHING

matching algorithm, which provides a basic foundation for the auto coloring approaches which will be introduced in the following chapters.

Chapter 4

Topology Enhanced Auto Coloring

In traditional 2D animation production, each frame is separated into several layers according to the following principles [57]:

- Importance: Main character and subordinate characters should be separated, so should main motion and subordinate motions.
- Rule of movement: Characters, objects or parts of them with different rules of movement should be separated.
- Speed of movement: Characters, objects or parts of them with different speed of movement should be separated.

Accordingly, characters and the background are drawn in different layers because they have different rules or speed of movement. Similarly, objects or parts of the character with different rules or speed of movement are usually drawn in another layer. For example, in a scene that a fox's tail is swaying while the body is still, the tail should be drawn in a different layer. With layer separation, modification is easy without having to redraw the whole frame if the partial motion is not ideal or the speed needs to be adjusted. In commercial production, different animators have their own layer separation methods in different situations. This raises uncertainty of how a character may appear in a single

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

layer. In most cases, if there are more than one main characters in the scene, each one of them is drawn in a separate layer, or even separated to more layers. As illustrated in Figure 4.1, the two characters are separated into four layers.

Figure 4.1: Layer separation (courtesy of Anime International Co., Inc.)

To simplify the problem, our research focuses on coloring characters with an assumption that there is only one character in each layer, and objects which do not appear in master frames are not drawn in the same layer as the character. In some animation,

each region is first painted with a flat color which will subsequently be composed with its highlight and shadow layers. Similarly, each region is assumed to have only one flat color in our research.

In this chapter, our auto coloring approach based on the region matching algorithm proposed in Chapter 3 is introduced in detail. First, the stable topology and a *Hierarchical Region Matching (HRM)* approach are investigated. Subsequently, the whole coloring process is introduced in three stages. At the first stage, regions in all master frames are matched to establish the correspondences among them and extract the stable topology for each master frame. At the second stage, the first frame of a sequence is matched based on the master frames. The correspondence among the first frame and all master frames are hence established, and a stable topology is inherited from the master frames. At the third stage, each frame in a sequence is matched with its prior one based on the HRM approach. The whole sequence is thus colored by inheriting the color of the first frame according to the matching. Some examples are given and analyzed. Based on these examples, a comparison is made to show the advantages of our approach over existing auto coloring algorithms. The last section is the summary of this chapter.

4.1 Stable topology

The rendition of traditional 2D animation is the 2D projection of 3D character which is visualized in the animator's mind. As indicated by E Catmull [23], the lack of explicit 3D information constitutes a major problem in traditional 2D animation. Without 3D information, an animator cannot imagine the character's behavior in 3D space and hence cannot project it onto 2D frames to assure coherence and quality of animation. In current 2D animation production, some animators even use rough 3D models to obtain a clear image of the character's posture and position in certain frames, as shown in

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

Figure 4.2. Similarly, without 3D information, the inbetweening and coloring processes will experience difficulty as well.

Figure 4.2: Using 3D character to assist 2D animation production (courtesy of Mahendra Sidharta Suryadi)

To make use of 3D information, some methods are proposed. For example, 2.5D models are investigated in [37], using the ordering of depth as the supplement of the lost third dimensional information. However, the modeling procedure of a 2.5D model is quite different from the traditional drawing process of a 2D character. It is improper to force animators to abandon their familiar and efficient drawing method and adopt the new modeling method to specify depth for each primitive (open or closed curves as introduced in [37]). Moreover, the depth information is not stable for each primitive. For example, an arm which is in front of the body in one frame may be behind it in another frame, which may cause matching errors of two regions or strokes in these two frames. The same problem exists in other systems using 2.5D models. For example in Inkwell [58], the drawing order for 2D objects is strictly prioritized. The problem mentioned

above is indicated by the author as the future work in [58] since no satisfactory solution is available. Approximate 3D models are utilized in [36] to help animation creation and rendering. The modeling process is similar to that presented in [48] for freeform model creation. In [71, 25], a set of consistent rough 3D models are created for adding shading effects, texture mapping, inbetweening generation etc. However, the approximate or rough 3D models are not suitable for application in our auto coloring purpose. Correspondences among all the 2D frames need to be established before the character model construction in [71, 25], either by user-specified corresponding “feature strokes” [71] or points and paths [25]. But correspondences among regions need to be established automatically in our auto coloring research. Another limitation is that the model can only be constructed very roughly, losing details like the chiffon of the clothes in Figure 4.2 (a) which is not represented in (b). Nevertheless, these details are very important for auto inbetweening and coloring. Moreover, the deformation of rough 3D character models to retain the mentioned details is a very time-consuming task as pointed in [71, 25]. The construction of a precise and detailed 3D character model is also a tedious process. In traditional animation 2D characters may not simply be the projection of 3D character models. Animators, for aesthetic reasons, deliberately break the rules of geometry and physics governing real-world objects to exaggerate reality [73]. So even with an ideal 3D model, matching from the model to 2D frames is still necessary. In summary, existing techniques based on 2.5D or 3D are not suitable to be applied in our auto coloring approach.

An alternative way to make use of 3D information is to extract *stable topology* from 2D frames. The stable topology reflects those corresponding stable relations in a 3D character which is not needed to be constructed. The topology of a 3D character may change with motion or view angle, but part of the topology is always *stable* as illustrated in Figure 4.3, where the relations among head, neck, torso and two arms of the character

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

Figure 4.3: Stable topology

constitute a stable topology. Though the character is rotated in different angles, it is obvious the stable topology is preserved in (a), (c), (d) and (e). The left arm in (b) is occluded and invisible, but it is actually connected to the torso in 3D space. The right arm neighbors the right leg in (a) and (e). But as the right arm in each of the other frames doesn't neighbor the right leg, the relation between them is not considered as stable.

Two related regions i and j are *stably related* if they have a stable relation. The stable relation is denoted as $\langle i - j \rangle$. And a region which has a stable relation with others is defined as a *stable region* in this thesis.

4.2 Hierarchical region matching

Traditional 2D animation hierarchy can be expanded so that the matching process is expanded from the scope of single-level regions, as introduced in Chapter 3, to that of multi-level components.

4.2.1 Hierarchization

4.2.1.1 Traditional 2D animation hierarchy

Figure 4.4: Traditional 2D animation hierarchy

As shown in Figure 4.4, in traditional 2D animation production, after a story is developed, it is split into different scenes, which are generally defined by a certain location and set of characters. Each scene is composed of a sequence of successive frames, and each frame is separated into several layers according to the importance of characters or motions, rule and speed of movement [57]. Accordingly, characters and background are drawn in different layers. Each layer can be segmented into many regions. Regions may be further divided into strokes or even points.

4.2.1.2 Hierarchy expansion of region level

Figure 4.5: Expanded hierarchy of region level

Most existing auto coloring algorithms as reviewed in Section 2.4 handle region matching at the region level. In our algorithm, the region level is expanded into several component levels, as shown in Figure 4.5. The component at each level has one or several children (regions or components), and each child has a parent. A component may contain both regions and sub-components, or only regions while the last/lowest component level contains only region(s). With the expanded hierarchy, matching is carried out from the first to the last component level.

4.2.1.3 Hierarchization

In traditional 2D animation production, animators usually construct a character from simple convex regions, as shown in Figure 4.6. The process of character construction is

Figure 4.6: A character construction process

from simple to complex, coarse to fine, which is similar to our hierarchization process. As indicated in [60], the convex hull of object boundaries is quite useful for object shape recognition. The convex hull has found wide application in pattern recognition and image processing, such as object recognition in [95] and 3D blob detection in [74]. As the convex hull can approximate regions of a character as shown in Figure 4.7, it is applied to hierarchization in our approach.

Figure 4.7: Character approximation

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

The convex hull of a set of points S in n dimensions is the intersection of all convex sets containing S . For N points $p_1, \dots, p_N$, the convex hull CH is then given by the expression

$$CH = \left\{ \sum_{j=1}^N \lambda_j p_j : \lambda_j \geq 0 \text{ for all } j \text{ and } \sum_{j=1}^N \lambda_j = 1 \right\} \quad (\text{Eq. 4.1})$$

In the case of two dimensions it is the smallest convex polygon that includes all the points, as shown in Figure 4.8 [1].

Figure 4.8: Convex hull

To expand a region level into several component levels, some regions are grouped to form components. Two grouping methods can be applied as follows:

Grouping Method 1

- Compute the convex hull $CH(i)$ and its area $A(i)$ for each region i .
- For all pixels $(x_j, y_j), j = 1, \dots, m$ at the border of each $CH(i)$, compute their average coordinate:

$$\bar{x} = \frac{\sum_{j=1}^m x_j}{m} \quad (\text{Eq. 4.2})$$

$$\bar{y} = \frac{\sum_{j=1}^m y_j}{m} \quad (\text{Eq. 4.3})$$

where m is the total number of $CH(i)$'s border pixels. The point at the average coordinate $(\bar{x}, \bar{y})$ is recorded as $\bar{P}_i$.

- For two related regions (one region neighbors to or inside the other), compute the area $A(i, i')$ of the overlapped part of two convex hulls $CH(i)$ and $CH(i')$.
- Compute the ratio of the overlapped part to each convex hull:

$$R(i, i') = A(i, i')/A(i) \quad (\text{Eq. 4.4})$$

$$R(i', i) = A(i, i')/A(i') \quad (\text{Eq. 4.5})$$

- Regions i and i' are grouped together if the following condition is satisfied:

$$\begin{aligned} & ((R(i, i') > 0.5) \cup (R(i', i) > 0.5)) \cap \\ & ((\bar{P}_i \in CH(i')) \cup (\bar{P}_{i'} \in CH(i))) \end{aligned} \quad (\text{Eq. 4.6})$$

where $(\bar{P}_i \in CH(i'))$ indicates that $\bar{P}_i$ is inside $CH(i')$, and 0.5 is the threshold preset empirically.

Using this grouping method, components at the 1st level can be derived. The regions which are grouped to form a component are called the component's *element regions*. For example in Figure 4.9, regions 3, 5, 6, 9, 10, 12, 16 in (a) are the element regions of the component 2 at the 1st level in (b), and regions 5 and 9 in (a) are the element regions of the component 5 at the 2nd level in (c).

After grouping, the shared border of each component's element regions will be removed so that the component can be combined into a new region. Each component or ungrouped region at the 1st level can thus be labelled with a unique number i , as shown in (b). For some components at the 1st level, the corresponding combined regions may have some inner element regions which do not have shared borders with them. For example, regions 5, 6, 9, 10, 12, 16 in (a) do not have shared border with component 4 in (b). Applying the same grouping method to these inner element regions, some components are decomposed into their children (sub-components or regions) at the 2nd level.

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

Figure 4.9: Hierarchization result

As illustrated in Figure 4.10, the component 2 at the 1st level is decomposed into sub-components 5, 6 and regions 3, 8, 9 at the 2nd level. The rest of the components are decomposed into regions at this level. For example, the component 1 at the 1st level is decomposed into regions 1 and 2 at the 2nd level.

The above grouping and decomposition process iterates at the lower level until no components can be found. Regions and components at each level can be obtained after iteration, as illustrated in Figure 4.9 and Figure 4.10.

Grouping Method 2 For two related regions i and i' , the union of their convex hulls $CH(i) \cup CH(i')$ is no bigger than the convex hull of their union $CH(i \cup i')$. The diversity $D_{CH}(i, i')$ between their areas is used as the reference for grouping. The regions with small $D_{CH}(i, i')$ should usually be grouped in the same component, while the regions with

Figure 4.10: Grouping method 1: expanded hierarchy

large $D_{CH}(i, i')$ belong to two different components. This method determines how related two regions are by their geometric relation. As illustrated in Figure 4.11, the region 7 is grouped with 1 as $D_{CH}(7, 1)$ is the minimum. Regions 7 and 1 hence constitute the pincers of the crab, which is physiologically in accordant to the real crab's structure.

$$D_{CH}(i, i') = CH(i \cup i') - CH(i) \cup C(i') \quad (\text{Eq. 4.7})$$

$i : 7, i' : 1, 3$

$D_{CH}(7, 1) = 8110 \text{ pixels}$

$D_{CH}(7, 3) = 11760 \text{ pixels}$

Figure 4.11: Diversity $D_{CH}(i, i')$

For each region i and each of its related regions i' , compute the $D_{CH}(i, i')$. If region

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

i' has the minimum value of $D_{CH}(i, i')$ with region i , i' and i should be grouped into the same component.

Figure 4.12: Hierarchization result

The hierarchization result for the same image in Figure 4.9 (a) is illustrated in Figures 4.12 and 4.13.

No standard has been clearly defined to evaluate the hierarchization methods as far as we know. Compared with method 1, the number of components segmented by the method 2 at the first component level is greatly reduced. And the segmented components by the method 2 is in accordant to the real crab's physiological structure: the body, the pincers and the legs. So we choose method 2 as the hierarchization method for our hierarchical region matching approach.

Figure 4.13: Grouping method 2: expanded hierarchy

4.2.2 Hierarchical region matching

As indicated by H Bunke in [21], graph matching has been widely used in pattern recognition and computer vision, including recognition of graphical symbols, character recognition, shape analysis, 3D object recognition etc. The expanded hierarchy graph as shown in Figure 4.13 can be represented as $g = \{v, w\}$, where v is a set of nodes in the graph and w is the relation among them. As two hierarchy graphs for two frames are usually not isomorphic due to the region variation, they can only be matched by applying error-tolerant graph matching algorithms. However, an optimal error-tolerant graph matching is a NP-complete problem, which requires exponential time and space [21, 27]. So we propose a suboptimal graph matching approach to match the nodes (components and regions) in two hierarchy graphs, from the first to the last component level. For two hierarchy graphs g and g' , a node v_i in g is matched with a set V'_i composed of some nodes in g' , which are selected according to the following rules:

For each v_i ,

- If its parent is matched to a component, V'_i is composed of the component's children;

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

- If its parent is matched to a region, the region is the only element composing V'_i ;
- If its parent is not matched to any component or region, V'_i includes all components and regions at the same level and those regions which have not been matched at the higher level.

The above graph matching method confines the selection of CRs within a range based on the matching result of a TR's parent, matching two graphs hierarchically. It is defined as a *Hierarchical Region Matching (HRM)* approach in this thesis. As it is not an optimal solution, error may arise when a node's parent is matched to a wrong component or region. The node may thus be matched to one of the wrong component's children or the wrong region. This means the region matching error at one level may be propagated to lower levels. However, the error mentioned above may still exist due to the high similarity of wrongly-matched regions if HRM is not applied. With HRM, the region matching accuracy is increased by avoiding the matching with wrong CRs, as analyzed later in Section 4.4.4.

4.2.3 Applicability analysis

To apply the proposed HRM approach, the components segmented at each level should be matchable, especially at the first component level, where the matching results will affect the matching accuracy at lower levels. It means that if several regions are grouped using the hierarchization method in one frame, most of their corresponding regions should be grouped in other frames, which is defined as *coherent*. The ideal situation is that all of their corresponding regions (if exist) are grouped in all other frames.

As introduced in Section 1.1, the master frames are drawn in different view angles, and occlusion is thus introduced. Accordingly, the shapes of regions composing the character are changing greatly among master frames. As our hierarchization method is based on

Figure 4.14: Segmented components for master frames

the convex hull of the region, which is computed based on the region shape, it is hard to obtain a coherent hierarchization result. Figure 4.14 illustrates the hierarchization results for four master frames at the first component level, and the components and regions are hard to be accurately matched by our region matching algorithm proposed in Chapter 3.

As for a target frame, it is also hard to obtain a coherent hierarchization with the master frames, because the posture of the character is usually quite different from the standing pose of master frames. As shown in Figures 4.15 (e) and 4.14, the hierarchization result for the first frame is quite different from those for the master frames in Figure 4.14. And the incoherent hierarchization in master frames may also introduce matching errors.

For a sequence of frames, changes between every two successive frames are either small or partial, which is mostly caused by the motion of a part, for example the legs. It is thus possible to have coherent hierarchization result, which satisfies the HRM approach. As shown in Figure 4.15, the hierarchization results are coherent at the first component level.

Figure 4.15: Segmented components for a sequence of frames

In summary, the HRM approach is not suitable for the matching among master frames or from master frames to a target frame, but is suitable for matching a sequence of frames.

4.3 Auto coloring process

The coloring process consists of three stages. Correspondences among regions in painted master frames are first built with the stable topology extracted. Then the first frame of each sequence is colored based on master frames with the stable topology inherited from master frames. Finally, by referring to the first frame of each sequence and the stable topology, each frame in a sequence is colored according to the color information of its previous frame.

4.3.1 Matching master frames

At the Model Sheet stage in traditional 2D animation production as introduced in Section 2.1, the artist designs master frames by referring to the 3D character in his mind [89]. Accordingly, correspondences of regions among different master frames are first established to extract the stable topology, which is later used to match regions in a target frame.

Each master frame is matched with others by applying the feature-based region matching algorithm introduced in Chapter 3. As all master frames are painted, the color information is used to quantize regions into several scales, each scale with a different color. The matching between regions is thus within each scale, applying the algorithm introduced in Section 3.2.

After all master frames are matched, region correspondences are established among them, and a stable topology for each master frame (if exist) is extracted accordingly.

For two related regions i and j in a master frame, the relation between them is treated as *stable* if and only if the following condition is satisfied:

i and j have at least one pair of corresponding regions i' and j' in one of other master frames; and for each pair of i' and j' in all master frames, i' and j' must be related.

For example, as shown in Figure 4.16, the regions A and B (label in black) in (d) have no corresponding regions in other master frames, so the relation between these two related regions is not stable. The regions A and B (label in white) in (c) are related, but all their corresponding regions A' and B' (label in white) in other master frames are not related, so the relation between them is not stable either. The regions A and B (label in yellow) in (a) and all their corresponding ones A' and B' in other master frames are related, so the relation between them is stable.

After all regions and their related regions are examined, the stable topology is built with all stable relations. Figure 4.17 shows the stable topology composed of extracted

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

Figure 4.16: Stable relation judgment

stable relations.

4.3.2 Coloring the first frame of a sequence

As reviewed in Section 2.4, all existing auto coloring algorithms focus on auto coloring a sequence of frames based on the first frame which is manually colored. To simulate the entire coloring process of traditional 2D animation production, we match the first frame in a sequence based on the color information and the extracted stable topology in master frames. Subsequently, a new subset of the stable topology is extracted from the target frame, by referring to the stable topology of master frames and the established correspondence among the master frames and the target frame.

The matching process is similar to that introduced in Section 3.2.3. As the matching is from several reference frames (the colored master frames) to a target frame, each region

Figure 4.17: Extracted stable topology

i in the target frame is matched with regions in the same area scale in all master frames. The best-matched region i' is hence obtained in one of the master frames. According to the established correspondences among all regions in master frames, i' and all regions corresponded to i' are recorded as matched. So their color information is coded in the character string to assist the subsequent matching process. Finally all regions in the target frame are matched and colors are inherited from the best-matched ones in master frames.

After region matching, the stable topology is inherited from the master frames. For two related regions i and j , if their best-matched regions i' and j' in a master frame are stably related, the relation between i and j is also treated as stable. Figure 4.18 shows the matched first frame of a sequence. The stable topology inherited from master frames in Figure 4.17 is also illustrated.

Figure 4.18: Inherited stable topology

4.3.3 Coloring a sequence of frames

As analyzed in Section 4.2.3, HRM approach is suitable for matching two successive frames in a sequence. So based on the HRM approach and extracted stable topology in the first frame, the coloring process for a sequence of frames is carried out in the following steps:

4.3.3.1 Topology recomposition

To match the components and regions at each component level, the stable topology in the first frame of a sequence is recomposed into those specified for each component level. If the first frame is not colored based on master frames but manually colored, the stable topology can also be manually defined at the character level and then automatically recomposed by the algorithm proposed below.

The recomposition rule is defined as following: For two related components or regions i and j at the same component level, if two of their element regions (or region itself) i_m and j_n is stably related ($\langle i_m - j_n \rangle$), the relation between i and j is stable ($\langle i - j \rangle$). As shown in Figure 4.19, relations $\langle 1, 3 \rangle$, $\langle 2, 3 \rangle$ in (a) are stable because region 3's element region 5 is stably related to regions 8 and 9 in (b), which are respectively element regions of regions 2 and 1 in (a).

Figure 4.19: Recomposition rule

After all stable relations for each level are reorganized, the stable topology for each level is recomposed from them, as illustrated in Figure 4.20.

4.3.3.2 Region matching

After the stable topology is recomposed for each component level, the region matching process is carried out as described by the following three steps:

At the first step, the stable topology in the reference frame is traced to match the regions in the target frame, as shown in Figure 4.21.

During the region matching process, the stable topology is also used to verify whether the *best-Matched Region (MR)* is correct or not. If one region i in the target frame is best-matched to region i' in the reference frame, the correspondence between them is only established when the following condition is satisfied:

If some of i' 's stably related regions have already been matched, their MRs and i in the target frame should have the same topology as their corresponding ones in the reference frame. If i' 's stably related region j' has already been best-matched to j , but j is not related to i in the target frame, i is recorded as a wrongly-matched region, and the correspondence between i' and i is not established. As illustrated in Figure 4.22, regions 1', 2' and 3' are stably related in the reference frame. After regions 1 and 2 in

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

Figure 4.20: Topology recomposition

(b) are respectively matched to regions 1' and 2' in (a), region 3 in (b) is best-matched to region 3' in (a). As regions 3 and 2 are not related, the stable relation $\langle 3' - 2' \rangle$ is not inherited, and the correspondence between 3 and 3' is not established.

At the second step, based on the established correspondences, part of the stable topology is inherited. Tracing the new stable topology, unmatched regions related to each matched region i is matched with their CRs, which are the regions related to i 's corresponding region i' in the reference frame.

At the third step, the remaining unmatched regions in the target frame are selected as TR one by one, in the order determined by their related matched regions as introduced in Section 3.2.3. Each TR is matched with CRs in the same area scale in the reference frame.

By referring to the stable topology of the reference frame, a new subset of stable

Figure 4.21: Region matching process

Figure 4.22: Topology verification

topology is extracted and inherited. The stable topology is thus successively propagated and updated from one frame to the next in a sequence, assisting the matching between every two successive frames. For two related regions i and j in a frame of the sequence, the relation between them is treated as stable only if the best-matched regions i' and j' in the reference frame are stably related.

4.3.4 Postprocessing

After each region is matched and colored, a postprocessing is needed to restore the one-pixel-wide contour skeletons into original strokes. For each colored target frame, all black pixels of its corresponding line drawing are superimposed, as shown in Figure 4.23. The final colored frames are thus obtained and will be used in the composition stage of 2D animation production.

Figure 4.23: Postprocessing

4.4 Results and analysis

Several examples are designed to validate our proposed auto coloring approach. We test the matching accuracy from master frames to a target frame in Tests 1 and 2. In Tests 3 and 4, the advantages of applying stable topology and HRM are shown. Then we apply our approach to color the common human characters in 2D animation in Tests 5 and 6. Finally in Test 7, a Japanese-style animation sequence is used to test the possibility of applying our approach to real-world animation production.

4.4.1 Test 1

In the first test, a comparison of the coloring results based on only one master frame and all master frames is made. Figure 4.24 show the input images. (a)-(c) are three master frames, presenting the left, right and front views of a gorilla, and (d) is a target frame to be colored.

The coloring results are shown in Figure 4.25, where (a)-(c) are the results of matching based on only a single master frame, while (d) is the matching result based on all three master frames.

Some matching errors can be seen in Figures 4.25 (a) and (b). Both results show wrong coloring of the vest and the short pants. Obviously the two master frames of Figures 4.24 (a) and (b) are quite different from the target frame (d). Figure 4.25 (c) shows the result with only the cheek wrongly colored. From Figures 4.24 (c) and (d), we can see the dissimilarity between these two frames is small, hence a good matching can be expected.

Compared with Figures 4.25 (a), (b) and (c), the result (d) is fairly satisfactory when region matching between master frames is done first. Based on all three master frames, our algorithm tries to find the least dissimilar region corresponding to each region of the target frame. Therefore, regions can be best matched across different master frames.

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

Figure 4.24: Test 1: Input images (resolution: 778×700 pixels)

Our result indicates that among a total 27 regions in the target frame, 20 regions are matched to regions in MF3, 4 regions to MF2 and 3 regions to MF1.

From the comparison above we can see the accuracy of region matching is increased with more master frames.

4.4.2 Test 2

Two different target frames are tested in the second example. Large changes in position, shape and relations with neighboring regions can be seen in both target frames compared

Figure 4.25: Test 1: Results

with master frames in Figure 4.26. Despite the big dissimilarity, our algorithm successfully finds the similarity of corresponding regions and colors them correctly, whether the region is a big one (like the wing) or a small one (like the eye), as illustrated in Figure 4.27. This test shows the accuracy of our algorithm and the possibility of automating the whole coloring process in traditional 2D animation production.

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

Figure 4.26: Test 2: Master frames (resolution: 550×1100 pixels)

Figure 4.27: Test 2: Results (resolution: 998×892 and 988×1240 pixels)

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

Figure 4.28: Test 3 (resolution: 870 × 825 pixels)

4.4.3 Test 3

The third test is as shown in Figure 4.28, where (a)-(d) are four master frames, (e)-(h) are a sequence of frames with a total of 38 regions in (e) and 37 regions each in (f) to (h). Large change in the character's positions does not affect the matching because our approach is tolerant to the change of character's position. For some regions, their relations with neighboring regions also vary much. For example, the region of the trunk of the centipede neighbors to the region of the mouth but not to that of the head in Figures 4.28 (f) and (g), while it neighbors to the regions of both the mouth and the head in (e) and (h). Though the relations change, part of them is stable. In the above example, the region of the trunk is always neighboring to that of the mouth, thus the relation between them is stable. Such stable relations are extracted after the matching from master frames to the first target frame (e). It is formed by the stable relations, as illustrated in (i). The stable topology is impervious to large motion such as the character's limbs' movement which can be noted between any two successive frames. Assisted by the stable topology, all regions in each frame of the sequence are colored frame by frame as shown in (n)-(p), and (i)-(l) indicate that the stable topology is propagated and preserved. To test the accuracy of our auto coloring approach, regions are colored in various colors in the first target frame in (q), and the results are shown in (r)-(t). Regions are matched in order by tracing the stable topology, which ensures each region is only matched to the correct corresponding one, as proven by the results.

4.4.4 Test 4

Figure 4.29 shows our fourth test. Despite the large motion of the crab, each region is correctly colored, as illustrated in (o) to (s). The segmented components at the first component level are illustrated in (j) to (n). It can be noted that some regions in (i) may be wrongly colored if hierarchical region matching is not applied because in Figure 4.30

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

Figure 4.29: Test 4 (resolution: 1168 × 826 pixels)

Figure 4.30: Analysis for Test 2

Figure 4.31: Coloring results comparison

region 6 in (b) is more similar to region 18 in (a) than to region 6 or 7 in (a). With hierarchical region matching, regions 7 & 11, 6 & 9, 12&18 in (a), and 6 & 10, 5 & 9, 12&16 in (b) are grouped to form components 7, 6, 10 in (c), and 6, 5, 10 in (d) at the 2nd component level. Components 6, 5 and 10 in (d) are matched to components 7, 6 and 10 in (c) respectively. So region 18 in (a) is not a candidate region of region 6 in (b). Wrong correspondence between them is thus avoided. Finally, all regions at the 3rd level are correctly matched. Figure 4.31 illustrates the coloring result for Frame 5 according to correctly colored Frame 4 using methods without and with HRM. This test shows HRM approach is suitable for matching a sequence of frames, and can increase matching accuracy by avoiding wrong selection of CRs.

4.4.5 Tests 5 & 6

Human or human-like characters have been the most common characters in traditional cel animation. So we use two different human characters to test our auto coloring approach in Test 5 and 6. As shown in Figures 4.32 and 4.33. all regions in the target frames are correctly colored by our auto coloring approach.

Figure 4.32: Test 5 (resolution: 506 × 620 pixels)

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

Figure 4.33: Test 6 (resolution: 504 × 652 pixels)

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

Figure 4.34: Test 7 (resolution: 1500×914 pixels) (courtesy of Anime International Co., Inc.)

Figure 4.35: Error analysis for Test 7

4.4.6 Test 7

To test the possibility of applying our approach in practical animation production, we use a Japanese-style sequence in practical production of the trailer of “Justeen” animation. The 13 frames as shown in Figure 4.34 compose a cut in the trailer, showing a girl’s motion of head turning. It can be noticed that occlusion arises due to the 3D rotation of the character. To minimize the information loss due to occlusion, we select the medium frame, Frame 7, as the starting frame and manually color it. Then the prior and subsequent frames are colored one by one according to the color information of its next and prior frame respectively. From the results we can see most of the regions are correctly colored. The regions wrongly colored are mainly because there are no corresponding regions in the reference frame, like the regions in red circles representing part of the clothes in Figure 4.35 (a), part of the eyebrow in (b) and part of the ear in (c). In (a), the region appears with the 3D rotation. In (b) and (c), regions are occluded and separated into two or more parts. These kind of matching errors are hard to avoid, so we are going to solve it by separating the character into more layers or by user-intervention. Another kind is because of the information loss. As shown in Figure 4.35 (d), the region representing the teeth is wrongly colored as the tongue disappears with the mouth closed. This kind of mouth motion is quite common in 2D animation. One way to solve the problem is

separating the mouth to another layer, which is adapted by many animation companies with the need of lip sync. Another way is introducing the assistance of other information like the expression maps as shown in Figure 4.36. In summary, the coloring accuracy for the total 12 uncolored frames is over 90%, which shows that our approach is applicable to practical animation production. The average time for coloring each frame automatically applying our approach is within 30 seconds using a PC with a CPU of Intel(R) Xeon(TM) 3.20GHz. In practical animation production, the average time using coloring systems like the PaintMan in RETAS! Pro [9] to paint each frame is about several minutes, depending on the complexity of the animation frames. An animation is usually composed of a lot of cuts in practical animation production. To save the time and labor, the auto coloring task can be divided into many sub-tasks according to the division of cuts. Considering these sub-tasks can be done by a lot of computers concurrently, the coloring efficiency is quite high with our approach.

4.5 Comparison with other algorithms

Three auto coloring algorithms are compared with our auto coloring approach. As they are not be able to color the first frame of a sequence based on master frames, the comparison is made on the coloring of a sequence of frames.

The method of JS Madeira et al. [61] only uses the curvature variation of each contour pixel as the feature for similarity matching. As the shape of a region may vary much and it is common to have similar regions in one frame, this method will fail in coloring regions correctly in some cases. For example, if the method is applied to color Figures 4.28 (f) to (h) based on (m), regions representing the nose and the eye may be wrongly corresponded since the shapes of these two kinds of regions are very similar. The introduction of topological region features in our algorithm is able to correctly handle such cases.

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

Figure 4.36: Expression maps (courtesy of Anime International Co., Inc.)

(a) Reference Frame

(b) Target Frame

Figure 4.37: Limitation of the statistical feature

(a) Reference Frame

(b) Target Frame

Figure 4.38: Coloring results using our algorithm

Besides some statistical region features like area, bounding box, density and aspect ratio of the bounding box's length and width, CW Chang and SY Lee [24] also apply topological feature for region matching. The topological feature is similar to our feature of relations with neighboring regions. However, their matching has strict requirements, thus it cannot handle big changes in regions' relations with neighboring regions. For example, as stated in [24], it is quite difficult to find corresponding regions using only

CHAPTER 4. TOPOLOGY ENHANCED AUTO COLORING

statistical features if there is a significant change in the shape of some regions. As an example, A and A' in Figure 4.37 cannot be matched. Suppose Φ is the set of total colors of A 's neighboring regions, which is {color of B , color of C }, and Ψ is the set of total colors of A' 's neighboring regions, which is {color of B' , color of D' }. The set Ψ is not a subset of Φ , so A and A' cannot be matched due to topological feature similarity measurement in [24]. However, A and A' have similar areas, curve lengths, character points and neighboring regions, and the only difference between them is reflected in the character strings since the color codes of C and D' are not the same. With the hierarchical region matching, regions A and B are grouped as a component, so are A' and B' . These two components are matched first. Then regions A and A' can be correctly matched because they have the minimum TDV. The coloring result is shown in Figure 4.38.

Figure 4.39: Coloring results using the algorithm proposed by HS Seah and F Tian

For the algorithm of HS Seah and F Tian [80], as indicated by D Sykora et al. [87], the motion in cartoons seems to be coarse and the structural differences between two successive frames may be so big that it is impossible to track the rapid motion using optical flow estimation. Based on the optical flow, the computed displacement vector in [80] may not be accurate enough to handle big motions. For example, Figure 4.39 shows the results using the algorithm to color Figures 4.29 (f) to (i) based on (o). With big motion, some regions are wrongly colored.

4.6 Summary

In traditional 2D animation production, coloring is a very tedious task with tremendous time and labor being spent in the inking/coloring stage. In this chapter, our auto coloring process is introduced in detail. First, stable topology is investigated. With the stable topology, the stable relation between 2D regions are used to make use of 3D information of the character. A HRM approach is then proposed. The region level is expanded into several component levels by hierarchization. After hierarchization, the matching is carried out from the first to the last level. Subsequently the auto coloring process composed of three stages is introduced. At the first stage, correspondences among regions in all master frames are established and the stable topology for each frame is extracted. At the second stage, regions in the first frame of a sequence are matched with those in all master frames, and the colors are inherited from their best-matched ones. A stable topology is then inherited from the master frames according to the established correspondence with the master frames. The HRM approach is applied at the third stage. Regions in each frame is matched with those in the prior frame, and the stable topology is updated and propagated at each frame based on the matching. Results are then given to show the robustness of our approaches, and a comparison is made to highlight the advantage of our approach over other existing auto coloring algorithms. In general, our approach is robust for auto coloring 2D animation frames for most cases. Further research will be focusing on the auto coloring for some special cases, for example, a sequence with occlusion in some frames. As the first attempt, the next chapter provides a solution for handling occlusion problems with minor user-intervention.

Chapter 5

Auto Coloring with Character Registration

Figure 5.1: Occlusion

As mentioned in Section 4.2.2, the ideal segmentation should assure the coherence of components at each level. The coherence at the first component level is especially important as the matching result at this level will affect the matching accuracy at lower levels. However, without user-intervention, this coherency is hard to be assured, especially when occlusion exists in some frames. As shown in Figure 5.1, part of the torso is occluded by the arm in Frame 2, so the arm is grouped as the same component with the torso. But in Frame 1 the arm and the torso are two separate components. This kind of incoherence indicates possible occlusion. However, as there is no topological information

or prior knowledge on the character involved in the component matching process after segmentation, it is hard to locate and handle occlusion accordingly. To tackle the issue, a character registration approach is proposed in this chapter, aiming at detecting and handling the occlusion with minor user-intervention.

The rest of the chapter is organized as follows. First the character registration process is introduced in detail. Skeletons are extracted, matched and registered. Then the components at the first component level is segmented according to the registration. Subsequently, our occlusion detection and auto coloring process is introduced. Different matching methods are applied to regions in components of two different categories, reducing the influence of region feature change due to occlusion. Experiments are given to test the algorithm with results analyzed. A summary is given at the end of this chapter.

5.1 Character registration

5.1.1 Skeleton extraction

A skeleton is a useful shape abstraction that captures the essential topology of an object in both two and three dimensions [29]. It refers to a thinned version of the original object, which still retains the shape properties of the original object. Skeleton is therefore widely used in many applications for shape modeling and analysis [41], shape matching, motion tracking [35], image segmentation and registration [39]. Moreover, it is adopted to control the movement of characters for 3D articulated figure animation [90]. As for 2D animation, though some information like depth is lost and the extracted skeleton is not as accurate as the real one, the skeleton can preserve the most essential geometric and topological information of the character, which is thus useful for our character registration purpose.

As reviewed in [40], numerous algorithms have been developed for skeleton extraction. Compared with the active contours approach [40], these algorithms are more sensitive to extremes at the boundary or may lose connectivity.

CHAPTER 5. AUTO COLORING WITH CHARACTER REGISTRATION

The leaf nodes of topology are pre-estimated first and the positions of corresponding points at the skeleton are fixed in [40]. According to the fixed topology and estimated positions of leaf nodes, skeleton is extracted by evolving the active contour initialized at the outline, with junction nodes updated at the same time. Consequently, positions of leaf nodes are re-adjusted by comparison of the outline's reconstruction error. To apply the method, prior knowledge of the topology structure is required and the initial estimation of leaf nodes's positions needs to be in the "right segments" of the outline [40]. In traditional 2D animation, as the character is complicated and translation, rotation and occlusion usually influence the positions of the character's leaf nodes in topology graph greatly, the guess on the initial position and orientation of the skeleton based on the bounding box of the silhouette as described in [40] is not always accurate.

We propose an alternative way of extracting skeleton by driving the active contour towards the ridges of the distance map without fixing the leaf nodes. As the potential energy (distance transform in this method) will be minimized when the active contour approximates the ridges of distance map, branches, junction and leaf nodes can be clearly identified.

Active contour is commonly modeled as an elastic band with mass density ρ and elasticity κ moving in the potential energy field, which is parameterized by arc length as $x(\xi) = (x(\xi), y(\xi))$, $0 \leq \xi < 1$. The ridge extraction of an image function can thus be formulated as an energy minimization:

$$\begin{cases} \xi_{total} = \int_0^1 [\rho \varepsilon_p(x(\xi), y(\xi)) + \varepsilon_d(\xi)] d\xi \\ \varepsilon_d(\xi) = \kappa [(\frac{\partial x}{\partial \xi})^2 + (\frac{\partial y}{\partial \xi})^2] \end{cases}, \quad (\text{Eq. 5.1})$$

where $\varepsilon_p(x, y)$ is the potential energy function and $\varepsilon_d(\xi)$ is the energy of the elastic deformation.

The dynamics of the curve is obtained by Euler Lagrange equations:

$$x_t(\xi) = -\rho \nabla \varepsilon_p(x) + \kappa \frac{\partial^2 x}{\partial \xi^2}, \quad (\text{Eq. 5.2})$$

where $\mathbf{x} = (x, y)$. The dynamics of the curve is thus defined by two forces: the gradient of the potential energy and the elastic deformation force.

The evolution of the curve is restricted to the normal direction at every point on the curve by geodesic snakes algorithm [22], so (Eq. 5.2) becomes:

$$\mathbf{x}_t(\xi) = \langle (-\rho \nabla \varepsilon(\mathbf{x}) + \kappa \frac{\partial^2 \mathbf{x}}{\partial \xi^2}) \cdot \mathbf{N}(\xi) \rangle \mathbf{N}(\xi), \quad (\text{Eq. 5.3})$$

where $\langle u \cdot v \rangle$ is an inner product of vectors u and v , and $\mathbf{N}(\xi)$ is a unit length normal pointing inwards. The displacements along the tangent directions are postulated to affect the parametrization of the curve but not the shape, which is therefore excluded from the evolution dynamics.

A discrete approximation of (Eq. 5.3) is:

$$\Delta \mathbf{x}_i^t \triangleq \mathbf{x}_i^{t+1} - \mathbf{x}_i^t = \langle (-\rho \nabla \varepsilon(\mathbf{x}_i^t) + \kappa \sum_{j \in N_i} \mathbf{x}_j^t - \mathbf{x}_i^t) \cdot \mathbf{N}_i^t \rangle \mathbf{N}_i^t, \quad (\text{Eq. 5.4})$$

where $\mathbf{x}_i^t$ is the location of point i on the discrete approximation of the active contour at iteration t and $\mathbf{N}_i^t$ is the normal vector at that location estimated using the point's neighbors.

Using the negative distance transform $D(x, y)$ as the potential energy function, the updating rule of the point position is obtained as:

$$\Delta \mathbf{x}_i^t = \langle (\rho \nabla D(\mathbf{x}_i^t) + \kappa \sum_{j \in N_i} \mathbf{x}_j^t - \mathbf{x}_i^t) \cdot \mathbf{N}_i^t \rangle \mathbf{N}_i^t, \quad (\text{Eq. 5.5})$$

where $\mathbf{x}_i^t$ is a point (x, y) at iteration t on the active contour, $D(\mathbf{x})$ is the negative distance transform, and $\mathbf{N}_i^t$ is the normal vector at the location estimated using the point's neighbors. ρ and κ are the mass density and the elasticity respectively. [40]

The skeleton estimation method is proposed as follows:

- Fill the character with black color (value of 0 in a binary image).

CHAPTER 5. AUTO COLORING WITH CHARACTER REGISTRATION

- Compute the distance transform of the filled character.
- Initialize the active contour at the outline of the character.
- Iteratively evolve the active contour according to (Eq. 5.5).
- Stop the iteration when no point is moving significantly.

Figure 5.2 shows the evolving process of the active contour.

After the skeleton is obtained, it is further thinned under SUSAN thinning rules [82]. With the thinned skeleton, junction and leaf nodes can be easily obtained by examining each skeleton pixel's 8-connected neighbors. Those with only 1 neighboring skeleton pixel are leaf nodes and those with more than 2 neighboring skeleton pixels are junction nodes.

Starting from a leaf node, a branch can be traced by searching along each pixel's 8-connected neighboring pixels, and the tracing ends when reaching a junction node. The branch connecting a leaf node to a junction node is recorded as a *L-branch*. After all *L-branches* are extracted, the *J-branch* which connects two junction nodes is traced in the same way from the rest of unchecked skeleton pixels. As shown in Figure 5.3, junction nodes are illustrated in red, leaf nodes in blue, *L-branches* in green and *J-branches* in black, respectively.

The topology graph is also obtained by connecting the junction and leaf nodes, as illustrated in Figure 5.4. Each branch in the skeleton is associated with a corresponding branch in the topology graph.

5.1.2 Skeleton registration

To represent the character simply and clearly, and to reduce user-intervention, a general topology model is pre-defined for character registration. Figure 5.5 provides some common topology models for characters, where (a), (b) and (c) are for human & human-like

CHAPTER 5. AUTO COLORING WITH CHARACTER REGISTRATION

Figure 5.2: Skeleton estimation ($\rho = 0.8, \kappa = 0.5$)

animals, tailed human-like animals and quadruped respectively. New topology models can be defined, given different kinds of characters.

To assure registration accuracy, a frame is selected from the sequence to be colored as the first reference frame, which contains complete branches of the character as in the general topology model. In animation production, the frame can be selected by the

CHAPTER 5. AUTO COLORING WITH CHARACTER REGISTRATION

Figure 5.3: Extracted branches

Figure 5.4: Extracted topology graph

animator according to the above rule and colored manually.

Each branch of the character skeleton and topology graph in the selected frame is then corresponded to the one in the general model, as illustrated in Figure 5.6.

Figure 5.5: General topology model

Figure 5.6: Skeleton registration

5.1.3 Skeleton matching

Skeleton graphs have been widely used for both 2D and 3D shape matching, recognition and retrieval [47, 81, 86, 20, 45]. Among these algorithms, the matching of skeletons is mainly based on the topology graph or shock graph and some branch features like length or angle. However, the skeleton extracted from a 2D character loses topological stability because of occlusion, artifacts or contour details like high-curvature convex. Matching the two topology graphs only according to topology may give different results. Figures 5.7 (b) and (c) illustrate two matching results which are topologically correct but geometrically wrong. Applying the above algorithms with the branch features like length or angle, the branch of tail (labelled as 6) in (b) has a very high possibility to be matched to the branch of arm (labelled as 4) in (a). For relational indexing/matching [30], which can match the skeletons accurately with proper features, its computational complexity is

CHAPTER 5. AUTO COLORING WITH CHARACTER REGISTRATION

very high because it leads to rather complex search trees. So a new matching algorithm is proposed here, specifically for skeletons of characters in 2D animation, based on both the geometric and topological information of the skeleton.

Figure 5.7: Wrong matching

5.1.3.1 Geometric matching

The change between characters in two frames is often caused by the motion of certain parts. For example, the change of a running character is mainly due to the motion of its moving legs and arms as illustrated in Figure 5.1. Some parts, like the torso of the character in Figure 5.1, are more stable than the moving parts. To make use of the stability, skeletons are first matched based on their geometric information. In our algorithm, both global and local differences are considered for branches in the reference and target frames. Global dissimilarity due to the motion of branches is represented by the global Hausdorff distance $H_g(B_i, B_{i'})$. It is computed using the global coordinate system R_g^2 of the two frames. For two branches B_i and $B_{i'}$, the global Hausdorff distance is computed as:

$$\begin{cases} d(a, b) = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}, (x, y) \in R_g^2 \\ \vec{h}(B_i, B_{i'}) = \sup_{a \in B_i} \inf_{b \in B_{i'}} d(a, b) \\ H_g(B_i, B_{i'}) = \max\{\vec{h}(B_i, B_{i'}), \vec{h}(B_{i'}, B_i)\} \end{cases} \quad (\text{Eq. 5.6})$$

The other measurement of the dissimilarity is the local deformation of each branch, which is represented by the local Hausdorff distance $H_l(B_i, B_{i'})$. To compute the local

Hausdorff distance between branches B_i and $B_{i'}$, they are first transformed to the local coordinate system R_i^2 by the *Principle Components Analysis (PCA)* [42].

$$\begin{cases} B_i^l = T_i * (B_i - M_i) \\ B_{i'}^l = T_{i'} * (B_{i'} - M_{i'}) \end{cases} \quad (\text{Eq. 5.7})$$

where T_i, M_i and $T_{i'}, M_{i'}$ are respectively the PCA transforming matrices for B_i and $B_{i'}$. Subsequently the local Hausdorff distance $H_l(B_i, B_{i'})$ is obtained as:

$$\begin{cases} d(a, b) = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}, (x, y) \in R_i^2 \\ \vec{h}(B_i^l, B_{i'}^l) = \sup_{a \in B_i^l} \inf_{b \in B_{i'}^l} d(a, b) \\ H_l(B_i, B_{i'}) = \max\{\vec{h}(B_i^l, B_{i'}^l), \vec{h}(B_{i'}^l, B_i^l)\} \end{cases} \quad (\text{Eq. 5.8})$$

With the global and local Hausdorff distances between two branches, the dissimilarity value DV is defined as:

$$DV(i, i') = H_g(B_i, B_{i'}) + \min(H_l(B_i, B_{i'}), H_l(B_i, -B_{i'})) \quad (\text{Eq. 5.9})$$

Each branch B_i in the reference frame is matched with all branches in the target frame first. If $DV(i, i')$ is the minimum among all dissimilarity values, a matching ($B_i \rightarrow B_{i'}$) is established. If a bi-directional matching is achieved, $(B_i \rightarrow B_{i'}) \cap (B_{i'} \rightarrow B_i)$, B_i and $B_{i'}$ is corresponded ($B_i \leftrightarrow B_{i'}$), and $B_{i'}$ is registered as the same branch with B_i in the general topology model. The unregistered branches will be re-adjusted in later process.

Figure 5.8: Geometric matching

Figure 5.9: Registration based on geometric matching

The matching result for Frame 2 is shown in Figure 5.8, and Frame 1 is the reference frame. The branches with identical labels in Frames 1 and 2 are matched, and the branches in Frame 2 without labels are unmatched. The registration based on the geometric matching is illustrated in Figure 5.9.

5.1.3.2 Topological re-adjustment

After geometric matching, some branches with big motion or deformation may not be matched and registered, as shown in Figures 5.8 and 5.9. To match and register these branches and detect occlusion, a topological re-adjustment process is proposed as follows.

For a node N_k^t in a topology graph G , its subgraph $G(N_k^t)$ is defined to consist of N_k^t and its connected branches, and its corresponding sub-skeleton $S(N_k^s)$ is defined as $G(N_k^t)$'s the corresponding branches, as illustrated in Figure 5.10, where N_k^s is N_k^t 's corresponding node in the skeleton.

For each unregistered branch in the topology graph of the target frame, compute the ratio λ of registered branches to all branches connected to it. The branch with the maximum λ is selected as the branch to be re-adjusted. After matching the branch, λ of each branch is updated and a new branch is selected with maximum λ .

Figure 5.10: Subgraph and sub-skeleton

To re-adjust each selected branch $B_{i'}^t$ in the target frame, the corresponding skeleton branch $B_{i'}^s$ is matched based on the following steps:

- $B_{i'}^s$ and a registered skeleton branch $B_{j'}^s$, connected to it are grouped as a new skeleton branch $B_{i'j'}^s$.
- Compute the dissimilarity value $DV(j, i'j')$ between the skeleton branch $B_{i'j'}^s$ in the target frame and B_j^s in the reference frame which is matched to $B_{j'}^s$.
- If $DV(j, i'j') < DV(j, j')$, $B_{j'}^s$ is a grouping candidate of $B_{i'}^s$.
- If some grouping candidates are found, group branch $B_{i'}^s$ with $B_{j'}^s$ which has the minimum value of $DV(j, i'j')$, and re-adjust the topology graph accordingly.
- If no grouping candidate is found, apply the geometric matching approach to match unregistered branches in $S(N_{k'}^s)$ and $S(N_{l'}^s)$ with unmatched branches in $S(N_k^s)$ and $S(N_l^s)$, where $N_{k'}$ and $N_{l'}$ are the two nodes of branch $B_{i'}^s$. N_k and N_l are their corresponding skeleton nodes in the reference frame respectively. If all branches in $S(N_k^s)$ and $S(N_l^s)$ are already matched, $B_{i'}^s$ is deleted.

The matching and registration results for Frame 2 are illustrated in Figures 5.11 and 5.12 respectively.

Figure 5.11: Topological re-adjustment

Figure 5.12: Registration based on topological re-adjustment

5.1.4 Component segmentation

To quickly detect and locate the occlusion, characters are segmented into several components corresponding to the branches in the skeleton and topology graph.

During the process of skeletonization, the distance map D of each frame is computed. With the distance map and the skeleton, the shape of the character can be roughly reconstructed by drawing circle at each skeleton point $P(x, y)$ with radius r and compute the envelope, where r equals to the distance transform at $P(x, y)$. Similarly, for each L-branch B_i^t in the topology graph, the area within the reconstructed envelope from B_i^t 's corresponding skeleton branches is defined as B_i^t 's *Explained Area* $EA(B_i^t)$. Figure 5.13

shows $EA(B_1^t)$ in Frame 1, where the skeleton branch is illustrated in blue and the circle at each skeleton point in red.

Figure 5.13: Component segmentation

For each L-branch B_i^t in the topology graph, $EA(B_i^t)$ in the original frame is computed first. Subsequently, regions which overlap with $EA(B_i^t)$ are chosen as the candidate regions for the component corresponding to B_i^t .

The component segmentation method proposed in Section 4.2.1.3 is applied to segment the candidate regions into several candidate components. If only one component is formed by these candidate regions, it is defined as B_i^t 's corresponding component; otherwise the one furthest to node N_i^s is defined as B_i^t 's corresponding component, where N_i^s is the corresponding skeleton node of B_i^t 's junction node N_i^t . The reason is that one L-Branch in a topology graph can be corresponding to several branches in the skeleton graph. As shown in Figure 5.12, the L-Branch labelled as 1 in the topology graph is corresponded to 3 branches in the skeleton graph. To select the component which is nearest to a leaf node of a L-Branch in a topology graph, we select the one furthest to the junction node of its corresponding skeleton branches. As shown in Figure 5.13, candidate regions for B_1^t 's corresponding component form three candidate components,

CHAPTER 5. AUTO COLORING WITH CHARACTER REGISTRATION

and the red component is B_1^t 's corresponding component in Frame 2. For the rest of regions, each region k is classified into the component corresponding to B_i^t with maximum $A(k) \cap EA(B_i^t)$, where k and $A(k)$ is the label and area of the region k respectively, and B_i^t is one of the J-branches in the topology graph.

Figure 5.14: Segmentation results

The segmented components for Frames 1 and 2 are illustrated in Figure 5.14, where corresponding components are in the same color.

5.2 Auto coloring process

Our auto coloring process contains two steps: first the occlusion is detected and located at certain components, and the regions in components with occlusion are classified into one category, and the rest belong to the other category. Two different matching methods are then proposed for components in two categories respectively.

5.2.1 Occlusion detection

Occlusion can break the continuity of a region's contour and greatly change the region's features of area, curve length, character points and relations with neighboring regions. Correspondingly, the change in these features can be used for occlusion detection. The

topology of regions in a component may not be stable. It may change with motion, view angle, or animators' drawing styles even when there is no occlusion. Features like curve length, character points and relations with neighboring regions are thus not suitable for occlusion detection. To detect and locate occlusion in components, we propose a method based on the variation of region areas.

- Each region in the reference frame is quantized and coded as an integer (for example 1) based on its area, by the region quantization method and coding rule proposed in Section 3.2.2.3.
- All the regions in the target frame are quantized and coded based on the scale points computed in the reference frame.
- For regions in the corresponding components C_i and $C_{i'}$ in the reference and target frames respectively, their areas are coded as character strings of integers S_i and $S_{i'}$ by sorting their areas ascendingly, for example "1 1 1 2 2 3". Compute the least conversion cost $\gamma(S_i \rightarrow S_{i'})$ based on the algorithm presented in Section 3.2.2.3 subsequently.
- As animators sometimes break a region or merge some regions together, the total number of regions in corresponding components may be different even without occlusion. As illustrated in Figure 5.15, the hand indicated by the blue circle is represented by two regions while all others indicated by red circles only consist of one region. A threshold is hence empirically defined to tolerate possible change of character strings, which is due to the inconsistency of regions mentioned. If $\gamma(S_i \rightarrow S_{i'})$ is bigger than the threshold, occlusion may be located in the component $C_{i'}$. For animation sequences with different drawing styles, users may adjust the threshold accordingly.

Figure 5.15: Change in regions

5.2.2 Auto coloring

With occlusion detection, regions can be divided into two categories. Regions in the first category are the elements comprising components without occlusion, and those in the second category comprising components with occlusion.

For the first category, the hierarchical feature-based region matching approach as proposed in Section 4.2.2 is applied. Regions in each component in the target frame are matched with those in the corresponding component and components with identical color scheme. For example, regions in the component of a left arm in the target frame can be matched with regions in the components of both arms in the reference frame. This will increase the matching accuracy. As shown in Figure 5.16 (b), the region indicated by the green circle is matched with the one in the green circle in (a), although components to which they belong are not corresponded.

For the second category, the continuity of some region contours is broken because of occlusion. Accordingly, features of these regions such as area, curve length, character points and relations with neighboring regions change greatly. But the rest of the regions still have relatively stable area, curve length and character points features. So a matching method similar to that for the first category is applied to match these regions. Each region is matched with those in corresponding components and unmatched components in the reference frame. The only difference is that the feature of relations with neighboring

(a) Frame 1

(b) Frame 2 (Step 1)

(c) Frame 2 (Step 2)

(d) Frame 2 (Step 3)

Figure 5.16: Auto coloring process

regions is ignored, so that the coded character string is only composed of character points. The result after this step is illustrated in Figure 5.16 (c).

Finally, each of the remaining regions in the target frame inherits the color of the unmatched region that it overlaps mostly in the reference frame. If no such region is obtained, which means the region fully overlaps the background, it is painted in the color which fills the majority of the component (major color) that it belongs to. The reason is that these regions are mostly segmented from big ones due to occlusion. As shown in Figure 5.16 (d), regions indicated by blue circles inherit the overlapping regions' color, and those indicated by green circles are painted in the major color of the component that they belong to.

After skeleton registration for the first reference frame, its prior and subsequent frames

(if exist) are matched separately. New target frame is selected backward for the prior frames, and forward for the subsequent ones. For each selected target frame, its nearest colored frame with complete branches as in the general topology model is selected as the reference frame.

5.3 Results and Analysis

Several examples are applied to test our proposed approach.

Figure 5.17 shows the original sequence, registered topology graphs and skeletons, and segmentation and coloring results. Occlusion can be noticed in Frames 2 and 4, where the arm occludes part of the torso. The general topology model in Figure 5.5 (b) is used for registration, and the coloring results using our auto coloring approach with character registration are illustrated. Frame 1 is selected as the first reference frame, which is colored and registered manually. As occlusion is detected in Frame 2 with one branch of arm occluded, character in Frame 3 is matched with that in Frame 1 as mentioned in Section 5.2.2. In Frame 2, component 3 is not available, and occlusion is detected in components 2 and 4. So regions in component 4 in Frame 2 are matched with those in components 4 and 3 in Frame 1, while regions in component 2 in Frame 2 are matched with those in components 2 and 3 in Frame 1. The occlusion is thus accurately detected, located and handled, as illustrated in Figure 5.16 and Figure 5.17.

In Figure 5.17, most of the regions are correctly colored with our proposed auto coloring approach. Some regions wrongly colored in Frame 3 and Frame 4 are illustrated in Figure 5.18. The region indicated by the blue circle in Frame 3 is a part of an occluded region representing a spot. Due to occlusion, it cannot be matched to any region in the corresponding component, so it inherits the major color of the component. This kind of error can be solved by introducing knowledge-based coloring rules with minor user-intervention. For example, regions with its area smaller than a threshold in the torso

CHAPTER 5. AUTO COLORING WITH CHARACTER REGISTRATION

Figure 5.17: Test 1 (resolution: 768 × 576 pixels)

and limbs can be pre-defined to be painted in the color of the spot, except for those near the leaf nodes of limbs, for these regions are possibly fingers and toes. With character

Figure 5.18: Error analysis for Test 1

registration, knowledge-based coloring rules can be defined to increase the matching accuracy with minor user-intervention. The other error is caused by the feature-based region matching algorithm proposed in Chapter 3. As the region of the toe indicated by the green circle in Frame 3 is similar to the region of sole spot indicated by the green circle in Frame 1 in all features used in the algorithm, it is wrongly matched. Subsequently the error is propagated to Frame 4. It indicates that more features like the convex hull of a region may be needed for our feature-based region matching algorithm.

Figure 5.19 illustrates our second test with more complicated occlusion, while all regions are correctly colored using our proposed auto coloring approach. Occlusion is detected in component 4 in Frames 2, 3 and 4, 5 in Frames 2 and 4, and 2 in Frame 3. For components 1, 3 and 6, no occlusion is detected.

It can be noticed that the hierarchization results in this example are not coherent for each frame due to the complex occlusion. For example, the component 2 in Frames 1 and 2 are the background enclosed by the arm and torso, while their corresponding ones in Frames 3 and 4 are representing the left arm. And as illustrated in Figure 5.20, the region i representing the ball is not correctly grouped in the component representing the left leg because it has smaller diversity with region j than k ($D_{CH}(i, j) < D_{CH}(i, k)$). For more details please refer to Section 4.2.1.

CHAPTER 5. AUTO COLORING WITH CHARACTER REGISTRATION

Figure 5.19: Test 2 (resolution: 484 × 1084 pixels)

CHAPTER 5. AUTO COLORING WITH CHARACTER REGISTRATION

Frame 1

Figure 5.20: Test 2: Component segmentation for the left leg

However, as regions in component 2 of Frame 3 are not matched to any region in component 2 of Frame 2, they inherit the color of the region representing the left arm in Frame 2 according to the overlapping part. The same situation occurs for region representing the left leg in Frames 2 and 4, which are correctly colored though unmatched to any region. So possible hierarchization error introduced by the occlusion does not fail our auto coloring approach.

Complicated occlusion can also be noticed in Figure 5.21. Frame 3 is selected as the first reference frame and the general model in Figure 5.5 (c) is used. Most of regions are correctly colored except for some in Frame 2, as indicated in Figure 5.22. Since these regions of background cannot be distinguished from those representing the character, they inherit the major color of the component of leg. Hard to be solved automatically, this kind of error can be avoided with minor user-intervention by separating the character and the background in the preprocess stage.

Figure 5.21: Test 3 (resolution: 570×410 pixels)

Figure 5.22: Error analysis for Test 3

5.4 Limitation

The character registration approach is proposed as the first attempt to handle occlusion problems with minor-intervention and has achieved some success as proven by the results. Nevertheless it has limitation to handle some cases with occlusion caused by other objects, within a component or information loss.

Occlusion caused by other objects As introduced in the beginning of Chapter 4, our research focuses on coloring characters with an assumption that there is only one character in each layer. It is true in most cases especially for the production in big

CHAPTER 5. AUTO COLORING WITH CHARACTER REGISTRATION

companies, where the layers are separated in a strict standard and a character may even be separated into several layers as shown in Figure 4.1. However, in some small companies, artists may draw several characters, or a character together with other foreground objects in one layer to save the cost, or just because of the personal habit. For the case when a character is separated into several layers, minor user-intervention is needed when registering the obtained skeleton branches. The coloring accuracy is not reduced, sometimes even increased as some components have already been separated into different layers accurately. For the case when different layers or foreground objects are drawn in the same layer, the overlapping among them introduces great difficulty for accurate skeleton extraction and registration. So it is quite hard to obtain a correct registration result, which will fail the following occlusion detection and auto coloring processes. To solve the problem, users may need to separate characters and objects into different layers.

Occlusion within a component Our approach can handle occlusion at the component level, but will fail for more complicated cases where the occlusion appears within a component. As shown in Figure 5.23 (a), the cat character is bowing down and the head and upper torso occlude the lower torso, part of the two arms and left leg, which constitutes a very complicated occlusion. From the obtained skeleton shown in Figure 5.23 (b), we can see the occlusion is within the component of torso, and the skeleton is quite different from the one in Figure 5.3. It is impossible to find a proper registration with the general topology model in Figure 5.5 (b) manually, not to mention the accurate automatic matching with other correctly-registered skeletons. With the occlusion within a component, it is hard to obtain a correct registration result automatically. Users may need to manually register part of the skeleton branches where no occlusion occurs, like the right leg and tail in Figure 5.23. For other parts with complicated occlusion, users may need to color the regions manually to assure the accuracy.

Figure 5.23: Occlusion within a component

Information loss Besides the occlusions mentioned above, information loss can also lead to registration and matching difficulty, as the obtained skeleton is only partial. Figure 5.24 shows several animation frames selected from a sequence. The skeleton in (d) is obtained without losing any information of the character in (a), and it can be registered to the model in Figure 5.5 (b). (b) and (c) are the last two frames in the sequence. As the dog is running away, only part of it is drawn and a lot of information is hence lost. This brings great difficulty to register and match the corresponding skeletons in (e) and (f). Correspondingly, the occlusion detection and auto coloring are not achievable.

The above limitation may be partially solved by introducing more user-intervention to register and match some of the skeleton branches manually. For example, the skeleton branches representing the tail and visible leg in (e) and (f) can be manually registered to the model in Figure 5.5 (b) and assist the later auto coloring process.

5.5 Summary

In this chapter, a character registration approach is proposed to detect and handle occlusion for auto coloring 2D animation characters. With the character registration and component segmentation, occlusion can be detected and located in certain components. Subsequently two different matching methods are applied to regions in components with

CHAPTER 5. AUTO COLORING WITH CHARACTER REGISTRATION

Figure 5.24: Information loss

and without occlusion respectively. Examples have shown that occlusion can be detected and handled with minor user-intervention. Limitation due to the occlusion caused by other objects, within a component and information loss are also analyzed.

With the registration, knowledge-based coloring rules may further be introduced, and expression maps as shown in Figure 4.36 can be used as the reference for coloring the component of head. The registration and segmentation results may also be used to assist inbetweening as mentioned in [26]. Furthermore, a library with occlusion cases may be established for later reference as many characters can be registered with the same topology model.

Chapter 6

Conclusion and Future work

This chapter summaries the work presented in this thesis. The conclusion is made, and future work is discussed.

6.1 Conclusion

Traditionally, 2D animation production has been a labor-intensive artisan process of building up animated sequences all by hand. Among all the tasks in the production, drawing and inking/coloring of the individual animated characters for each frame take up approximately 60% of total labor required [32]. A lot of research work is hence carried out, aiming at obtaining an suitable approach to generate inbetweens or color characters in an animation sequence automatically. In this thesis, we introduced our research work on the computer-assisted auto coloring, in order to save time and labor in the coloring stage of traditional 2D animation production.

In Chapter 2, the traditional 2D animation production process is reviewed. Some existing 2D animation systems are discussed in relation to their coloring modules or functions. It is noted that most of them provide no or very limited auto coloring function. Some image matching techniques and auto coloring algorithms are then investigated. Given the common constraints that the change between the reference and target frames

CHAPTER 6. CONCLUSION AND FUTURE WORK

must be small enough and the first frame of each sequence must still be manually colored, computer-assisted auto coloring remains an open area which needs further investigation.

In 2D animation, each character can be treated as a set of regions which are closed by strokes. Commonly, the coloring work is to find the correct correspondences among regions and fill them in flat colors. Accordingly, a feature-based region matching algorithm is advanced in Chapter 3 to match regions in one target frame with those in one or several reference frames. First, the line drawings are scanned into the computer and preprocessed by noise-reduction, line-thinning and gap-closing algorithms. Subsequently, regions are segmented from each image and labelled, each with a unique number. The region features of color, area, curve length, character points and relations with neighboring regions are then investigated and chosen to be the features for region matching. Feature Dissimilarity Values (FDVs) of one feature or combined features are computed to represent dissimilarities of designated features between regions. Based on these FDVs, Total Dissimilarity Values (TDVs) which represents dissimilarities of all features between regions are computed using the method of least-squares. The correspondence between the region having the least-valued TDV and the target region is hence established. Results have shown that our feature-based region matching algorithm is suitably robust to be the foundation for auto coloring approach.

To cover the entire coloring process in the traditional 2D animation production, our auto coloring approach comprising three stages is proposed in Chapter 4. Stable topology is introduced to make use of some 3D information of a character. A HRM approach is then proposed with its applicability analyzed. At the first stage, master frames are matched to establish the correspondences among their regions, and stable topology is extracted accordingly. At the second stage, regions in the first frame of a sequence are matched with all regions in the master frames to establish the correspondences, and a new subset of stable topology is inherited from the master frames. To match a sequence of

frames, regions in each frame is matched with those in the prior frame at the third stage, based on the proposed HRM approach. The region level of traditional 2D animation is expanded into several component levels, and the stable topology is first recomposed for each component level. Assisted with the recomposed stable topology, components and regions are matched from the first to the last component level, and a new subset of stable topology is then inherited at each frame. The stable topology is hence updated and propagated to the whole sequence, assisting the matching at each frame. Some examples are given and analyzed and a comparison is made with some existing auto coloring algorithms. We show the advantage of our auto coloring approaches and its potential to be applied in practical 2D animation production.

In some cases, occlusion exists in certain frames of a sequence, which will significantly influence the region features and hierarchization results and may lead to wrong region matching. To detect and handle occlusion, character registration is proposed in Chapter 5. Some general topology models are defined first with minor user-intervention. Then the skeleton and topology graph of character in each frame is extracted. A frame with the complete branches as the general topology model is then selected as the first reference frame and manually registered, with each branch corresponded to one in the model. Subsequently, skeletons in the reference frame and a selected target frame are matched based on the geometric and topological information. Referring to the registered skeleton, components are segmented at the first component level, each corresponded to a branch in the general topology model. Occlusion is thus detected and located in certain components. Two kinds of matching methods are advanced for the components with and without occlusion respectively. Inaccurate matching due to the occlusion is hence tackled, making the auto coloring approach more robust.

6.2 Future work

The future work is discussed as follows:

6.2.1 Region feature expansion

As introduced in Chapter 3, the region features chosen for our region matching algorithm are the color, area, curve length, character points and relations with neighboring regions. These region features are sufficient in most cases. While for some cases where two or more similar regions exist in the same frame, it may be difficult to distinguish them according to the features above. As illustrated in Figure 6.1, the matching error for regions of the toes in (e) and (f) and feathers in (f) is due to the similarity in all the proposed features. The same problem arises, wrongly matching the region of sole in Figure 6.2 (a) to that representing the toe in (b), and the error is propagated to the next frame (c). The latter error may be solved if features like the convex hull of the region is added into the region matching algorithm, as the convex hull of the regions representing the sole and the toe differs much. The former error is hard to solve as the feature of region's coordinates/position is not chosen for our region matching algorithm, in order to be tolerant to the translation and rotation of the character. New features associated with the position are needed to solve the problem.

6.2.2 Accuracy quantization

In our region matching approach, each target region i is matched with its candidate regions, and a best-matched one i' is obtained if $TDV(i, i')$ is the minimum and smaller than a threshold which is empirically set. It means that we only use two status to distinguish all the regions: matched or unmatched. For those unmatched regions, we define some rules to inheriting colors from its mostly overlapped regions or the major color of the component it belongs to in Section 5.2.2. In this way, some regions are

Figure 6.1: Test 1 in Chapter 3

Figure 6.2: Test 1 in Chapter 5

correctly colored but some are not. In [19] the author claims that it is easier to color some blanked regions than recolor the wrongly-colored ones. It is true in the sense that

CHAPTER 6. CONCLUSION AND FUTURE WORK

animators may not be as fast enough to detect errors than blanks, as the blanks are more obvious to them. But we also want to keep the unmatched but correctly colored regions to speed the coloring process. To tackle this dilemma, an accuracy quantization method may be helpful. For example, the matching accuracy is quantized from 0 to 1. For those regions with 100% matching accuracy, we keep the color. For the rest of regions, we show them to the user in another frame with the color and matching accuracy displayed. The user is then able to quickly decide to keep or repaint the color according to the display. The method to calculate and quantize the matching accuracy is going to be investigated in the future.

6.2.3 Automatic weight adjustment

As mentioned in Chapter 3, a general weights set is determined as $\lambda_a : \lambda_{cl} : \lambda_{rc} = 1 : 1 : 4$ for all the examples. This general set of weights is chosen based on our experiments and manual adjustment. Though it is correct for most cases, it is not adaptable for all the animation frames. A future work is to research the relations of weights and attributes, trying to find a common rule to build self-adaptive weights. For big regions, the weight of area λ_a may be bigger than that of small regions. For regions having sharp character points which exist in all frames, the weight of relations & character points λ_{rc} may be bigger than normal regions. Based on these knowledge and further analysis, it is expected that the weights be determined automatically, thus making our algorithm applicable to most animation frames if not all.

The learning algorithms, also known as training algorithms, may be of some help to build self-adaptive weights. The learning algorithms are procedures for modifying the weights on the connection links in a Neural Network and can be generally divided into 3 groups: supervised learning, unsupervised learning and reinforcement learning [68]. Providing a set of correct colored frames, the difference between them and the outputs

can be used to modify the weights by supervised learning. As mentioned in Section 6.2.2, with the matching accuracy provided, a threshold can be defined to distinguish ‘good’ and ‘bad’ outputs, and the weights can be adjusted accordingly with reinforcement learning.

After all, these are ideas before further research on neural networks and learning algorithms, the applicability will be judged in later research and other methods possible for construction of self-adaptive weights should be investigated as well.

6.2.4 Character relocation

The auto coloring approach with character registration proposed in Chapter 5 may be challenged if occlusion continues in a sequence of successive frames. In this case, the character’s position in the selected reference frame may differ greatly from that in the target frame, which will influence the matching accuracy of proposed skeleton matching algorithm. A robust relocation method is to be advanced to handle this issue, mimicing the way animators relocate two key frames to draw inbetweens, as illustrated in Figure 6.3. In addition, the translation and rotation of the character can be handled if a robust relocation method is available. Consequently, the coordinates information can also be applied to assist the region matching, in order to handle the problem mentioned in Section 6.2.1.

6.2.5 Further investigation on 2D animation

To apply our auto coloring algorithms to a generic 2D animation system, further investigation on 2D animation is needed.

6.2.5.1 Model sheets

Figure 2.2 in Chapter 2 shows a traditional model sheet. Besides master frames, the expressions of the character are also illustrated from different view angles at the bottom of the model sheet. And there are also expression maps as mentioned in Chapter 5.

CHAPTER 6. CONCLUSION AND FUTURE WORK

Figure 6.3: Conventional relocation method in inbetween drawing

Another type of model sheet, usually contains the character's different motions as the reference for later drawing and coloring, is shown in Figure 6.4. Such expression maps and model sheets can also be very useful information for region matching, which will be investigated and added as the reference for our auto coloring approach in the future.

6.2.5.2 Coloring

In our current research, each region is assumed to have one flat color. To make the characters more lively, regions are colored with different textures or gradients in some animations. Shadow and highlight are drawn in different layer and these layers are composited with the layers of characters to achieve the effect of shading and highlighting. But in some animation production, they are not created in the drawing stage but in the painting stage. For example, the characters in “The Race” animation are colored with shading and highlighting in the painting stage, as shown in Figure 6.5. Accordingly, the study of coloring with textures, gradients, highlighting and shading is one part of our future work. Coons patch [28, 38] may be used for coloring with texture as mentioned in section 2.2 when reviewing Inkwell [58]. The manga colorization approach in [75] may also be applied to segment an image into regions and extract patterns or textures of

each region, and color regions with gradients and shading effects. Using the leak-proof colorization method as mentioned in [75], we may also simplify the preprocessing by removing the gap-closing process.

6.2.5.3 Shot

Different shots such as close-up, medium shot, medium full shot and full shot are often used in film-making and traditional animation [13]. As shown in Figure 6.6, a character's whole body only appears in full shot. Our algorithm currently focuses on the coloring of characters in one shot type, is not able to handle the shot changes. When there is change of shots between frames, some parts of the character are partially drawn and the character is zoomed in or out, which leads to large changes in region features introduced in Chapter 3. This poses much difficulty to our matching algorithm in its current form. Further investigation is needed to color characters with the mentioned shot changes. Some ideas may be feasible such as knowledge-based detection for different shots first and then applying the HRM algorithm proposed in Section 4.2.2.

CHAPTER 6. CONCLUSION AND FUTURE WORK

Figure 6.4: Another type of model sheet (courtesy of Warner Bros.)

Figure 6.5: Different ways of coloring

CHAPTER 6. CONCLUSION AND FUTURE WORK

Figure 6.6: Shot types

References

- [1] <http://mathworld.wolfram.com/ConvexHull.html>.
- [2] <http://plasticanimationpaper.dk/>.
- [3] <http://www.animationstand.com/>.
- [4] http://www.bauhaussoftware.com/products_mirage_studio.php.
- [5] <http://www.cambridgeanimation.com/products/animo.htm>.
- [6] <http://www.celaction.com/>.
- [7] http://www.digicelinc.com/FlipBook_Painter.htm.
- [8] <http://www.macromedia.com/software/flash/>.
- [9] <http://www.retas.com/>.
- [10] <http://www.toonboom.com/products/>.
- [11] <http://www.toonz.com/>.
- [12] <http://www.usanimation.com/>.
- [13] D Ablan. [Digital]Directing - Types of Shots. *New Riders*, 2003.
- [14] F Arrebola, A Bandera, P Camacho, and F Sandoval. Corner detection by local histograms of cotour chain code. *Electronics Letters*, 33(21):1769–1771, 1997.
- [15] F Arrebola, A Bandera, P Camacho, and F Sandoval. Corner detection and curve representation by circular histograms of contour chain code. *Electronics Letters*, 35(13):1065–1067, 1999.

REFERENCES

-
- [16] F Arrebola, A Bandera, P Camacho, and F Sandoval. Corner detection by means of adaptively estimated curvature function. *Electronics Letters*, 36(2):124–126, 2000.
 - [17] F Arrebola, A Bandera, P Camacho, and F Sandoval. Corner detection by means of contour local vectors. *Electronics Letters*, 38(14):699–701, 2002.
 - [18] GW Awcock and R Thomas. *Applied image processing*. McGraw-Hill, Singapore, 1996.
 - [19] H Bezerra, B Feijo, and L Velho. A computer-assisted colorization algorithm based on topological difference. In *Proceedings of 19th Brazilian Symposium on Computer Graphics and Image Processing*, pages 71–77, 2006.
 - [20] A Brennecke and T Isenberg. 3D shape matching using skeleton graphs. In *Simulation and Visualisierung (SimVis)*, SCS European Publishing House, Erlangen, San Diego, pages 299–310, 2004.
 - [21] H Bunke. Graph matching: Theoretical foundations, algorithms, and applications. In *Vision Interface*, pages 82–88, 2000.
 - [22] V Caselles, R Kimmel, and G Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, 1997.
 - [23] E Catmull. The problems of computer-assisted animation. *Computer Graphics (SIGGRAPH '78 Proceedings)*, 12(3):348–353, August 1978.
 - [24] CW Chang and SY Lee. Automatic cel painting in computer-assisted cartoon production using similarity recognition. *The Journal of Visualization and Computer Animation*, 8(3):165–185, 1997.
 - [25] BY Chen, Y Ono, and T Nishita. Character animation creation using hand-drawn sketches. *The Visual Computer*, 21(8-10):551–558, 2005.
 - [26] Q Chen, F Tian, HS Seah, J Qiu, and K Melikhov. Motion estimation based on segmentation for keyframe inbetweening. In *Proceedings of International Workshop on Advanced Imaging Technology*, pages 12–17, Jan 2006.

REFERENCES

- [27] D Conte, P Foggia, C Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. 18(3):265–298, 2004.
- [28] S Coons. Surfaces for computer-aided design of space forms. In *M.I.T. MAC-TR-41*, June 1967.
- [29] ND Cornea, D Silver, X Yuan, and R Balasubramanian. Computing hierarchical curve-skeletons of 3D objects. *The Visual Computer*, 21:945–955, Oct 2005.
- [30] MS Costa and LG Shapiro. 3D object recognition and pose with relational indexing. *Computer Vision and Image Understanding*, 79(3):364–407, 2000.
- [31] M Donias, P Baylou, and N Keskes. Curvature of oriented patterns: 2-d and 3-d estimation from differential geometry. In *1998 International Conference on Image Processing*, volume 1, pages 236–240, October 1998.
- [32] CX Durand. The TOON project: requirement for a computerized 2D animation system. In *Proceedings of ACM SIGGRAPH*, pages 285–295, 1991.
- [33] KB Eom and J Park. Contour models for curvature estimation and shape decomposition. In *the 11th IAPR International Conference on Pattern Recognition*, volume 2, pages 393–396, 1992.
- [34] JD Fekete, E Bizouarn, E Cournarie, T Galas, and F Taillefer. TicTacToon: a paperless system for professional 2D animation. In *Computer Graphics Proceedings, Annual Conference Series*, pages 79–90, 1995.
- [35] H Fijiyoshi and AJ Lipton. Real-time human motion analysis by image skeletonization. In *Proceedings of IEEE Workshop on Application of Computer Vision*, pages 15–21, 1998.
- [36] F Di Fiore and F Van Reeth. Employing approximate 3D models to enrich traditional computer assisted animation. In *Proceedings of Computer Animation*, pages 183–190, June 2002.

REFERENCES

-
- [37] F Di Fiore, P Schaeken, K Elens, and F Van Reeth. Automatic in-betweening in computer assisted animation by exploiting 2.5D modeling techniques. In *Proceedings of Computer Animation*, pages 192–200, 2001.
- [38] A Forrest. On coons and other methods for the representation of curved surfaces. In *Computer Graphics and Image Processing*, volume 1, pages 341–369, 1972.
- [39] DS Fritsch, D Eberly, SM Pizer, and MJ McAuliffe. Stimulated cores and their applications in medical imaging. In *Proceedings of Information Processing in Medical Imaging*, pages 365–368, 1995.
- [40] P Golland and WEL Grimson. Fixed topology skeletons. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 10–17, June 2000.
- [41] P Golland, WEL Grimson, and R Kikinis. Statistical shape analysis using fixed topology skeletons: corpus callosum study. In *Proceedings of Information Processing in Medical Imaging*, pages 382–387, 1999.
- [42] RC Gonzalez and RE Woods. *Digital image processing*. Addison-Wesley Publishing Company, USA, 1992.
- [43] LO Gorman. $k \times k$ thinning. *Computer Vision, Graphics, and Image Processing*, 51:195–215, 1990.
- [44] C Harris and M Stephens. A combined corner and edge detector. In *4th Alvey Vision Conference*, pages 189–192, 1988.
- [45] L He, CY Han, X Wang, X Li, and WG Wee. A skeleton based shape matching and recovery approach. In *Proceedings of International Conference on Image Processing*, pages 789–792, 2002.
- [46] DJ Heeger. Optical flow using spatiotemporal filters. *Computer Vision*, 2:279–302, 1987.
- [47] N Hiransakolwong, K Vu, KA Hua, and SD Lang. Shape recognition based on the medial axis approach. In *IEEE International Conference on Multimedia and Expo*, pages 257–260, 2004.

REFERENCES

- [48] T Igarashi, S Matsuoka, and H Tanaka. Teddy: a sketching interface for 3d freeform design. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 409–416, 1999.
- [49] DHU Kochanek and RH Bartels. Interpolation splines with local tension, continuity, and bias control. *Proceedings of ACM SIGGRAPH*, pages 33–41, 1984.
- [50] A Kort. Computer aided inbetweening. In *Non-Photorealistic Animation and Rendering 2002 (NPAR '02)*, Annecy, France, June 3-5 2002.
- [51] TL Kunii and T Maeda. On the silhouette cartoon animation. In *Computer Animation '96 Proceedings*, pages 110–117, 1996.
- [52] K-K Lai and PSY Wu. Effective edge-corner detection method for defected images. In *the 3rd International Conference on Signal Processing*, volume 2, pages 1151–1154, October 1996.
- [53] JS Lee, YN Sun, and CH Chen. Gray-level-based corner detection by using wavelet transform. In *Computer, Communication, Control and Power Engineering, TENCON '93*, volume 2, pages 970–973, October 1993.
- [54] R Legault and CY Suen. A comparison of methods of extracting curvature features. In *the 11th IAPR International Conference on Pattern Recognition*, volume 3, pages 134–138, 1992.
- [55] M Levoy. A color animation system based on the multiplane technique. *ACM Computer Graphics*, 11:65–71, 1977.
- [56] M Levoy. Area flooding algorithms. New York, NY 10036, USA, July 1982. ACM Press.
- [57] H Li. *Animation Techniques*, volume 1. Wuhan University of Technology Press, China, 2003.
- [58] PC Litwinowicz. Inkwell: a $2\frac{1}{2}D$ animation system. In *Computer Graphics / Proc. SIGGRAPH*, pages 113–122, 1991.

REFERENCES

-
- [59] H-C Liu and MD Srinath. A string descriptor for matching partial shapes. In *Shapiro L, Rosenfeld A (eds) Computer vision and image processing*, pages 575–592, San Diego, 1992. Academic Press.
- [60] S Liu-Yu and M Thonnat. Using apparent boundary and convex hull for the shape characterization of foraminifera images. In *Proceedings of 11th IAPR International Conference on Image, Speech and Signal Analysis*, volume 3, pages 569–572, 1992.
- [61] JS Madeira, A Stork, and MH Grob. An approach to computer-supported cartooning. *The Visual Computer*, 12:1–17, 1996.
- [62] S Milan, H Vaclav, and B Roger. *Image Processing, analysis and machine vision*. Chapman & Hall, London, 1996.
- [63] KT Miura, R Sato, and S Mori. A method of extracting curvature features and its application to handwritten character recognition. In *the Fourth International Conference on Document Analysis and Recognition*, volume 2, pages 450–454, August 1997.
- [64] F Mokhtarian and R Suomela. Curvature scale space for robust image corner detection. In *the 14th International Conference on Pattern Recognition*, volume 2, pages 1819–1821, August 1998.
- [65] F Mokhtarian and R Suomela. Curvature scale space for image point feature detection. In *International Conference on Image Processing and its Applications*, volume 1, pages 206–210, July 1999.
- [66] HH Nagel. On the estimation of optical flow: relations between different approaches and some new results. *Artificial Intell.*, 33:299–324, 1987.
- [67] S Nassif, D Capson, and A Vaz. Robust real-time corner location measurement. In *Sensing, Processing, Networking*, volume 1, pages 106–111, May 1997.
- [68] GW Ng. *Application of Neural Networks to Adaptive Control of Nonlinear Systems*. Research Studies Press LTD., 1997.

REFERENCES

- [69] L O’Gorman. An analysis of feature detectability from curvature estimation. In *Computer Society Conference on Computer Vision and Pattern Recognition*, pages 235–240, June 1988.
- [70] L O’Gorman. Curvilinear feature detection from curvature estimation. In *the 9th International Conference on Pattern Recognition*, volume 2, pages 1116–1119, November 1988.
- [71] Y Ono, BY Chen, and T Nishita. 3D character model creation from cel animation. In *Proceedings of International Conference on Cyberworlds*, pages 210–215, 2004.
- [72] CM Orange and FCA Groen. Model based corner detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1993.
- [73] JW Patterson and PJ Willis. Computer assisted animation: 2D or not 2D? *The Computer Journal*, 37(10):829–839, 1994.
- [74] Y Qing and P Bahram. CHEF: Convex Hull of Elliptic Features for 3D blob detection. In *Proceedings of 16th International Conference on Pattern Recognition*, volume 2, pages 282–285, 2002.
- [75] Y Qu, TT Wong, and PA Heng. Manga colorization. In *Proceedings of ACM SIGGRAPH*, pages 1214–1220, 2006.
- [76] K Rangarajan, M Shah, and D Van Brackle. Optimal corner detector. In *the 2nd International Conference on Computer Vision*, pages 90–94, December 1988.
- [77] A Rattarangsi and RT Chin. Scale-based detection of corners of planar curves. In *IEEE transactions on Pattern Analysis and Machine Intelligence*, volume 14, April 1992.
- [78] DF Rogers. *Procedural elements for computer graphics*. McGraw-Hill, America, 1998.
- [79] HS Seah and BC Chua. A skeletal line-joining algorithm. In *Insight Through Computer Graphics, Proceedings of the Computer Graphics International (CGI’ 94), Melbourne, Australia*, pages 62–73, 1994.

REFERENCES

-
- [80] HS Seah and F Tian. Computer-assisted coloring by matching line drawings. *The Visual Computer*, 16(5):289–304, 2000.
- [81] TB Sebastian and BB Kimia. Curves vs skeletons in object recognition. In *Proceedings of International Conference on Image Processing*, pages 22–25, 2001.
- [82] SM Smith. Edge thinning used in the SUSAN edge detector. *Internal Technical Report TR95SMS5, Defence Research Agency, Chobham Lane, Chertsey, Surrey, UK*, 1995.
- [83] SM Smith and JM Brady. Susan - a new approach to low-level image processing. *International Journal of Computer Vision*, 23(1):45–78, 1997.
- [84] K Sohn, WE Alexander, JH Kim, and WE Snyder. A constrained regularization approach to robust corner detection. In *IEEE Transactions on Systems, Man and Cybernetics*, volume 24, pages 820–828, May 1994.
- [85] E Sojka. A new algorithm for detecting corners in digital images. In *the 18th Spring Conference on Computer Graphics*, pages 55–62, April 2002.
- [86] H Sundar, D Silver, N Gagvani, and S Dickinson. Skeleton based shape matching and retrieval. In *Shape Modeling International*, pages 130–139, 2003.
- [87] D Sykora, J Burianek, and J Zara. Unsupervised Colorization of Black-and-White Cartoons. In *Proceedings of NPAR 2004, the 3rd International Symposium on Non-Photorealistic Animation and Rendering*, pages 121–128, 2004.
- [88] CH Teh and RT Chin. A scale-independent dominant point detection algorithm. *IEEE trans Computer Vision & Pattern Recognition*, pages 229–234, 1988.
- [89] NM Thalmann and D Thalmann. *Computer Animation: Theory and Practice*. Springer-Verlag, 1985.
- [90] L Wade and RE Parent. Fast, fully-automated generation of control skeletons for use in animation. In *Proceedings of Computer Animation*, pages 164–169, 2000.

REFERENCES

- [91] H Wang and M Brady. A practical solution to corner detection. In *International Conference on Image Processing*, volume 1, pages 919–923, 1994.
- [92] AM Waxman. An image flow paradigm. In *Workshop on Computer Vision: Representation and Control*, pages 49–57. IEEE Computer Society Press, 1984.
- [93] J Weng, TS Huang, and N Ahuja. *Motion and Structure from Image Sequences*. Springer Verlag, 1993.
- [94] M Xie. Feature matching and affine transformation for 2d cel animation. *The Visual Computer*, 12:419–428, 1995.
- [95] MP Yu and KC Lo. Object recognition by combining viewpoint invariant fourier descriptor and convex hull. In *Proceedings of Intelligent Multimedia, Video and Speech Processing 2001*, pages 401–404, 2001.
- [96] XN Zhang, RM Haralick, and V Ramesh. Corner detection using the map technique. In *the 12th IAPR International Conference on Pattern Recognition*, volume 1, pages 549–552, October 1994.
- [97] XN Zhang, RM Haralick, V Ramesh, AS Bedekar, and I Phillips. A Bayesian corner detector. In *IEEE International Conference on Image Processing*, volume 2, pages 282–286, 1994.

Publication

1. J Qiu, HS Seah, F Tian, Q Chen Q, and K Melikhov. Computer-assisted auto coloring by region matching. In *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*, 2003; pp. 175-184.
2. Q Chen, F Tian, HS Seah, J Qiu, and K Melikhov. Automatic inbetweening based on feature matching. In *Proceedings of International Workshop on Advanced Image Technology*, 2004; pp. 347-351.
3. J Qiu, HS Seah, F Tian, Z Wu, and Q Chen. Topology enhanced feature-based auto coloring for computer-assisted cel animation. In *Proceedings of the Seventh IASTED International Conference on Computer Graphics and Imaging*, 2004; pp. 26-32.
4. Q Chen, F Tian, HS Seah, J Qiu, and K Melikhov. Initial value specification in motion estimation for keyframe animation. In *Proceedings of the Seventh IASTED International Conference on Computer Graphics and Imaging*, 2004; pp. 33-38.
5. K Melikhov, F Tian, HS Seah, Q Chen, and J Qiu. Frame skeleton based auto-inbetweening in computer-assisted cel animation. In *Proceedings of the Third International Conference on Cyberworlds*, 2004; pp. 216-223.
6. Q Chen, Z Wu, F Tian, Seah H S, J Qiu, and K Melikhov. Vectorization of raster line-drawings in cartoons. In *Proceedings of International Conference on Computer Animation and Social Agents, CASA*, 2005; pp. 151-158.
7. J Qiu, HS Seah, F Tian, Z Wu, and Q Chen. Feature- and region-based auto painting for 2D animation. *The Visual Computer*, 21:928-944, Oct 2005.

8. J Qiu, HS Seah, F Tian, Q Chen, and Z Wu. Enhanced auto coloring with hierarchical region matching. *Computer Animation and Virtual Worlds*, 16(3-4):463-473, 2005.
9. J Qiu, HS Seah, F Tian, Q Chen, and K Melikhov. Topology enhanced component segmentation for 2D animation character. In *Proceedings of International Workshop on Advanced Imaging Technology*, 2006; pp. 30-35.
10. Q Chen, F Tian, HS Seah, J Qiu, and K Melikhov. Motion estimation based on segmentation for keyframe inbetweening. In *Proceedings of International Workshop on Advanced Imaging Technology*, 2006; pp. 12-17.
11. K Melikhov, F Tian, J Qiu, Q Chen, and HS Seah. Grayscale line image vectorization and rendering. In *Proceedings of International Workshop on Advanced Imaging Technology*, 2006; pp. 591-596.
12. K Melikhov, F Tian, J Qiu, Q Chen, and HS Seah. DBSC-based grayscale line image vectorization. *Journal of Computer Science and Technology*, 21(2):244-248, Mar 2006.
13. Q Chen, F Tian, HS Seah, Z Wu, J Qiu, and K Melikhov. DBSC-based animation enhanced with feature and motion. *Computer Animation and Virtual Worlds*, 17(3-4):189-198, 2006.
14. J Qiu, HS Seah, F Tian, Q Chen, Z Wu, and K Melikhov. Auto coloring with character registration, in *Proceedings of the 2006 international conference on Game research and development CyberGames '06*, 2006; pp. 25-32.
15. J Qiu, HS Seah, F Tian, Q Chen, Z Wu, and K Melikhov. Auto coloring with enhanced character registration, *International Journal of Computer Games Technology*, vol. 2008, Article ID 135398, 7 pages, 2008. doi:10.1155/2008/135398.
16. J Qiu, HS Seah, F Tian, Z Wu, and Q Chen. Layer Composition for line drawings in 2D animation. In *Proceedings of International Workshop on Advanced Imaging Technology*, 2008; pp.B4-5 1-6.