Journal of Zhejiang University SCIENCE A
ISSN 1009-3095
http://www.zju.edu.cn/jzus
E-mail: jzus@zju.edu.cn

Reuse of clips in cartoon animation based on language instructions*

WEI Bao-gang (魏宝刚), ZHU Wen-hao (朱文浩)†, YU Jin-hui (于金辉)

(*State Key Laboratory of CAD & CG, School of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China*)

†E-mail: nixon3103@163.com

Received Mar. 10, 2005; revision accepted Oct. 15, 2005

Abstract: This paper describes a new framework for reusing hand-drawn cartoon clips based on language understanding approach. Our framework involves two stages: a preprocessing phase, in which a hand-drawn clip library with mixed architecture is constructed, and the on-line phase, in which the domain dependent language instructions parsing is carried out and clips in the clip library are matched by use of some matching values calculated from the information derived from instruction parsing. An important feature of our approach is its ability to preserve the artistic quality of clips in the produced cartoon animations.

Key words: Language instruction, Reuse of clips, Content management
doi:10.1631/jzus.2006.A0123 **Document code:** A **CLC number:** TP391

INTRODUCTION

Traditional animated cartoons have highly exaggerated drawings and motions. These animations are usually created by highly trained animators who use animation principles to create motion that is expressive and stylized, so that traditional animation is a time consuming and expensive industry.

Much research effort has been devoted to the reduction of the work involved in different cartoon production phases, such as automatic in-betweening, automatic coloring, automatic inking as well as management aspects including the uniformity of processing, cross-referencing the various stages, database techniques for frames and sequences, the ability to capture and reuse movements, the separation of visual appearance from movement and sound, and sharing of load (both human and computational) (Patterson and Willis, 1994). Although many automatic in-betweening techniques have been proposed during the past 30 years (for a comprehensive review of automatic in-betweening methods, please refer to (Yu and Patterson, 1997; Gotsman and Surazhsky, 2001)), their use in computer aided systems is somehow limited to dealing with local movements only (Yu and Patterson, 1997), because exaggerated drawings used in cartoon animations follow animation principles and do not respect the geometry (Patterson and Willis, 1994) that is difficult for handling in-betweeners, so drawing in cartoon production still heavily relies on skilled animators.

These days, the main market for animation is television. Animations shown on television are usually produced in series to present many episodes which usually tell a whole story, in which same characters may act in the same place or in a different place, while drawing clips representing their actions may look similar, it is therefore possible for us to reuse these drawings with different background added in the new episodes.

Currently the reuse of drawing clips is achieved mainly by hand, that is, animators put drawings into the database and retrieve them manually for the new

* Project supported by the National Natural Science Foundation of China (No. 60373037), the Hi-Tech Research and Development Program (863) of China (No. 2004AA119060), Natural Science Foundation of Zhejiang Province (No. M603228), Zhejiang Science and Technology Plan Project, and Ningbo Science Foundation for Doctor, China

episode according to the storyboard. From the reuse point of view, we expect more clips can be put into the database, while a big clip source may lead to a tedious searching process if clip retrieving is done manually.

This paper presents a system called RUCLI (Reuse of Clips by Language Instruction) that automates this process. At a high level, our system proceeds in four steps as shown in Fig.1.

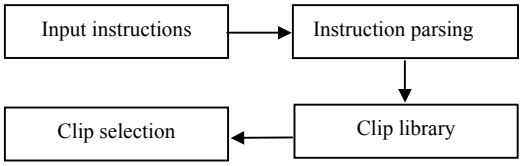


Fig.1 Overview of our approach

First, during off-line preprocessing, we construct a clip library containing hand-drawn cartoon clips with some attributes; next, during subsequent on-line sessions, language instructions are parsed to get some token values with which the system searches for suitable matches in the clip library; finally, the matched clips are played back for users to select and put it to the animation.

The remainder of the papers is organized as follows. In Section 2 we review some related work, Sections 3~5 describe clip library construction, language instruction parsing and matching process, Section 6 shows examples and Section 7 concludes with potential application and further work.

RELATED WORK

In this section we review some research that our work draws on. Current AI supported animation techniques can be classified into following categories:

(1) Animation from instructions, which focuses on command-driven intelligent agent animation creation (Webber, 1998; Bindiganavale *et al*., 2000). The AnimNL (Animation from Natural Language Instructions) project of Pennsylvania University expands the creation features to facial expression, gesture and speech synthesis.

(2) Knowledge-driven animation generation, this part of the work aims mainly at human action simulation (Vosinakis and Panayiotopoulos, 2001). The knowledge here includes the relationship of action parameters, physical information on track motion and action synchronization data.

(3) Autonomous communicative characters, with the integration of further understandings on linguistics, dynamic dialog knowledge base and multi-model IO support, the character is now communicative (Vilhjalmsson and Cassell, 1998; Cassell *et al*., 2001; 2000).

(4) Story-driven animation, a translator (Noma *et al*., 1992) from natural language to computer animation includes story formulation, action creation, environment generation and virtual camera calibration.

(5) Interactive storyboard, the main idea of storyboard provides an easy to use the system for the director, playwright and producer as a forepart visual design tool (Girard, 1987). Users can choose character, action, direction, background, image, etc. from the system library to assemble as a rudiment for later design.

Most of existing AI supported animation techniques focus on 3D realistic animations. The mechanism used in them to generate realistic movements of virtual characters is however unsuitable for generating exaggerated movements for cartoon animations.

In this work we investigate an AI supported approach for reusing hand-drawn cartoon clips in the cartoon series, an important feature of our approach is its ability to preserve the artistic quality of clips in the produced cartoon animations. The RUCLI architecture enables us to parse language instructions and transfer them into parameters that are used to match cartoon clips in the clip library. The three main issues we address are: (1) clip library construction, (2) language instruction parsing, and (3) matching process. We describe them in detail in the following three sections.

CLIP LIBRARY CONSTRUCTION

Construction of the clip library involves video classification and content management. There are two widely accepted approaches to characterize video in the database: shot-based and object-based. Benitez *et al*.(2000) employed conceptual entities and semantic and perceptual relationships among concepts to con-

struct a knowledge representation framework using multimedia content for representing semantic and perceptual information. By integrating conceptual and perceptual representations of knowledge, the retrieval effectiveness is improved. Fan *et al*.(2004) shortened the semantic gap between the low-level visual features and the high-level semantic visual concepts by using a hierarchical tree structure.

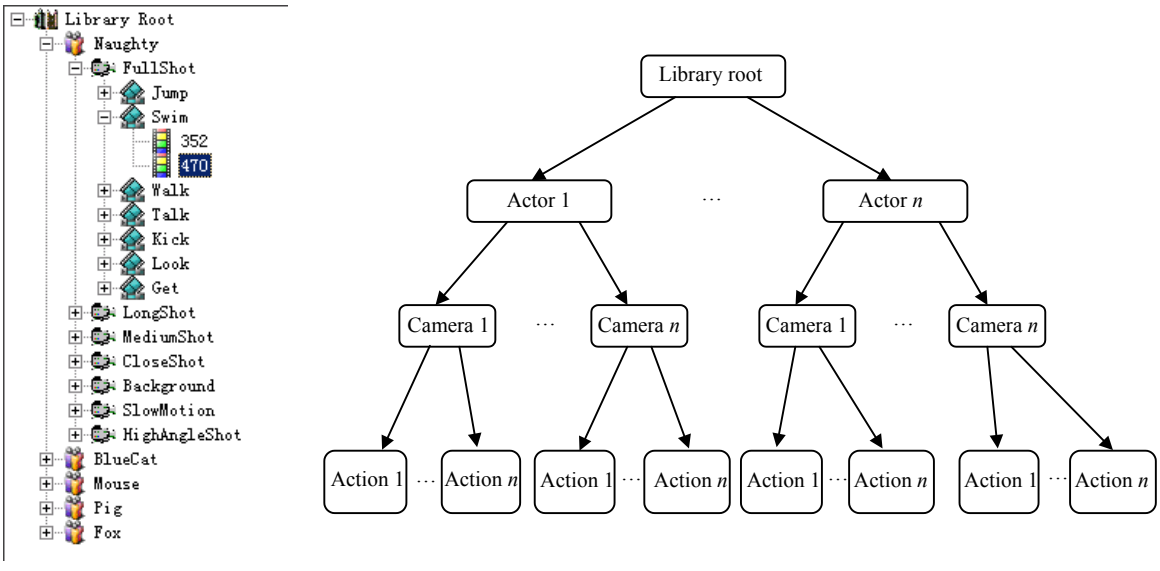
In RUCLI we adopt a mixed video content management technique. Each clip in the library is treated as an object. Attributes associated with an object include high-level semantic visual concepts, low-level visual features and context features.

High-level semantic visual concepts include character, action, shot, etc. Obviously, this information is kernel variable in clip matching. By establishing a tree structure as shown in Fig.2b, concept-oriented hierarchical video database browsing can be supported. Furthermore, the hierarchical structure of the clip library allows matching process at particular level by giving some matching weights. The low-level visual features include colors used to paint big areas on character's body, say, CColor for clothes and FColor for character's face. Context features include clip order in different episodes; we record the clip order in the completed episodes and use the clip order information to get better match.

The action attribute of the clip object in the clip library is written in the form of etymon for verbs. Assume we have two characters called Naughty and Blue cat, actions of Naughty are always written as "Naughty does something or somebody". For example, "Naughty eats something" or "Naughty chases Blue cat". With this form we avoid using multiple action attributes such as "Naughty ate/eats/is eating/will eat something" for the same action. The background information is not included in the action attribute because the same action may take place in different places.

Fig.2a is an actual library screen shot showing how the conceptual features are managed with the tree structure in Fig.2b. The number in Fig.2a indicates the clip ID, clips with different ID represent similar actions under the directory "swim", say, clip 352 represents Naughty's swimming seen from profile and clip 470 presents Naughty's swimming with his face facing the screen, etc. Clips under the same action directory are called sister clips, and methods for determining a particular clip among them are given in Section 5. Additionally, the exclusive clip ID connects the tree structure to a clip database, which maintains both high-level and low-level information for the specific clip.

The clip library is extensible. A new clip may be drawn to be added to the library for future use.



(a) (b)

Fig.2 The tree structure of the clip library

LANGUAGE INSTRUCTION PARSING

The ideal way of cartoon creation by reuse of the clip is to import the original playbook and then get a sample movie, which only needs little optimization manually. Natural language processing (NLP) is required to turn the language instructions into clip search results. Note, here we deal with the language instruction as the first step to the ideal method above. Since NLP is usually subject to ambiguous terms, domain knowledge, syntax and grammar restrictions, the parsing work is really difficult and unmanageable, especially for some complex syntax.

Fortunately, attributes associated with clips in the clip library allow high level retrieving in terms of character and action and shot, NLP in our system thus can be simplified on the base of these high level attributes. Accordingly we use simple language instructions instead as the input that need to be parsed for matching clips.

In RUCLI we only require language instructions be written with simple patterns such as Noun Phrases (NP)+Verb Phrase (VP) (e.g. "Naughty chases Blue cat") which can be parsed by use of Context Free Grammar (CFG) algorithm (Earley, 1970). For ordinary parsing purposes "Naughty chases Blue cat" is a wrong sentence, while in RUCLI it is acceptable because verbs in all action attributes of clips in the clip library are written in their etymon form. After parsing input instructions to get VP by use of CFG, we transform VP into its etymon by adding some rules to get transformed instruction that are finally used for matching.

With transformed instructions, RUCLI retrieves clips using the following procedure (written in pseudo codes):

```
GetTokens (streamText, listToken) {
  for each clip object in clip library {
    for each word in streamText {
      Create a token object obj and set
          obj.name = word
        for each attr of the clip object {
          if clip.GetAttribute(attr) == word
              obj.SetAttribute(attr, true)
        }
    if obj.HaveAttribute
      listToken.Add(obj)
    }
  }
}
```

where StreamText is the input text, listToken is the resultant token list, attr is the attribute name (e.g. shot, character, action, object, …), function GetAttribute gets the attribute value of the clip object and SetAttribute sets the corresponding attribute value (only true or false for token object) of the token object.

Since shot information such as full shot (fs), medium shot (ms), close shot (cs), etc., is given in the storyboard and independent of actions, our interface allows users to specify shot information according to the storyboard.

Table 1 shows results of processing instruction "Naughty chases Blue cat" and shot information.

Table 1 Results of processing a transformed instruction

Naughty	Chase	Blue cat	Shot
Actor: true Object: true	Action: true	Actor: true Object: true	Fs: true

Although high-level semantic visual concepts are important elements for matching, we may get multiple candidate clips by just using high-level semantic visual concepts to match clips. For instance, one character may do the same action in different clothes. In these cases, clips representing the same action for a character in different clothes have the same matching degree if only high level semantic visual concepts are used, which result in ambiguous matches.

In order to get more accurate match, we take some additional information such as low level visual information as well as context information into account during the matching phase, the detailed matching process is described next.

MATCHING PROCESS

In this section we first address low-level visual information and context information and then proceed to describe the matching process.

Low level visual information

In cartoon animation, two big areas on a char-

acter are usually on body and face. Colors painted on these areas can be used as low-level visual features. Since different clothes on characters body are usually painted with different colors, we take cloth color CColor, and the face color FColor as low-level visual features. Our interface allows users to specify these color features. In case users think it is not necessary to change the color information, colors are inherited from the previous clip.

It should be pointed out that color information is not a vital factor for clip matching, because color painted on character body may change in animation. For instance, a character's face may turn red or the whole body's color changes with sunset. Actually, users can find clips by specifying color information they remember, and repaint colors using image-editing systems in those cases.

Context information

Besides low-level visual information, context information such as clip orders in completed episodes are also taken into account to achieve a better match, because in completed episodes some action, say action A, may happen together with other actions, say action B and action C occur before and after action A, respectively, and the probability that action A occurs together with actions B and C is bigger than other actions. The clip order, corresponding to actions B-A-C thus reflects the context information in some completed episodes.

Note that in different episodes action A may occur simultaneously with different actions, say action J and K before and after action A, respectively, which form a new clip order corresponding to actions J-A-K. Our system records all clip orders in completed episodes and uses weighted statistics derived from clip orders to help users to get better matches. In the case of two candidate clip orders having the same statistical probability, we let users make the decision.

Matching function

RUCLI begins the matching process by looking up the clip library and getting the matching degree value by an object matching function, this function decides the matching result based on three factors: high-level semantic concepts, low-level visual features and context information.

The match value is calculated by use of Eq.(1):

$$V = H \times W_H + L \times W_L + C \times W_C, \tag{1}$$

where H stands for the conceptual attributes matching value, C for context information and L for low-level visual attributes. The matching value of H and L is calculated as the matched number. For example, the value of H is 3 if "actor", "action" and "shot" are matched and "object" is unmatched, and the value of L is 1 if CColor is matched. C is the probability of a candidate clip (suggested by high level semantic visual concepts parsing) appearing in completed episodes. W_H, W_L, W_C are weights assigned to H, L and C, respectively.

Since conceptual information is much more important than low-level visual features, we set W_H 10 times bigger than W_L in our implementation. The value of W_C is related to the number of completed episodes that containing the context relevant to the current episode, the default ratio of W_C:W_L is set as 1:10, users may change the ratio if necessary, say, 1:5 for W_C:W_L if they want to take more historical information into account.

The final matching degree is defined by Eq.(2), which gives users a numerical evaluation for the candidate clip:

$$D = \frac{V}{N_H \times W_H + N_L \times W_L + N_C \times W_C}, \tag{2}$$

where N_H and N_L are the total compared attributes number and N_C is the total number of completed episodes. In case that everything is matched, say, N_H=H, N_L=L and N_C=C, the value of D is 1.

For sister clips that are matched under the same action director, RUCLI first selects one clip randomly or, alternatively, users may selects one if they are not satisfied with the clip selected by the system.

RESULTS

Fig.3 shows the interface of RUCLI. With the input information shown at the bottom part of Fig.3, RUCLI finds two matches with high matching degree, 78% and 76%, respectively. The difference in matching degree is caused by different statistically based context information. A poor match of matching degree of 49% is also found, in the actor name. In this

example the background picture is added through the interface.

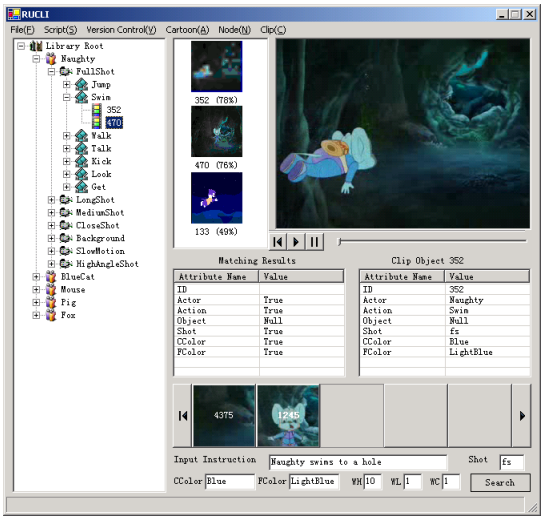


Fig.3 Searching result by input instruction "Naughty swims to a hole"

Fig.4 shows another example in which the language instruction is "Fox is astonished." In this example no context information is considered, and with the input instruction and color information only we get two matches with matching degree 100% (Fox is seen in front and profile, respectively), and the clip with ID 56 is finally selected according to storyboard. The third match (A girl called Feifei is astonished) just has matching degree of 63% because both actor name and color information are unmatched.

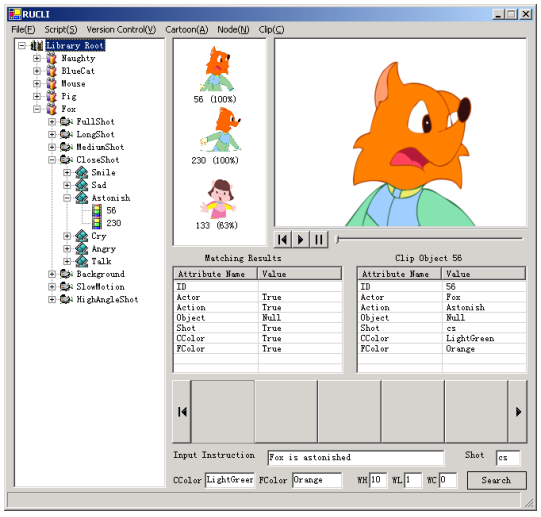


Fig.4 Searching result by input instruction "Fox is astonished"

CONCLUSION AND FUTURE WORK

In this paper we propose an object-oriented framework for making series cartoon with language instructions. With RUCLI users can reuse hand-drawn cartoon clips in the new episodes thus reducing the load of skilled animators. Moreover, by employing a domain-dependent language parsing method and a mixed video content management architecture, the system can be used in the network, which allows many users to access the clip library and reuse the clips with language instructions.

It should be noted that RUCLI aims at reusing hand-drawn cartoon clips when necessary. For some actions that RUCLI cannot find best matches in the clip library, or matched clips are unsatisfactory, animators are required to draw new clips according to the storyboard. The new clips can be put into the clip library for possible reuse in the future. With the growth of the clip library, we can expect more and more clips can be reused.

The work suggests a number of areas for future investigations:

(1) Reuse of old cartoon clips. In the history of cartoon-making, it is possible to discover really valuable and artistically advanced work, which are outstanding worldwide in modern cartoon production. Unfortunately, these old cartoons are only available in the form of movies. We can reuse actions in old cartoons if we extract moving characters from the backgrounds.

(2) Better representation for color information. In the current implemented RUCLI we use words to specify color features, a drawback of this method is that words are inadequate to represent rich colors that could be used to paint clothes and faces of cartoon characters. We need to find a better representation for color information specification for matching purpose.

(3) Automatic transition between similar actions. In our clip library there are several clips under the same action directory, these sister clips usually represent similar actions viewed from different angles that could be used in combination with each other in animation. Smooth transition between them can be achieved by abstracting their structural information, say skeletons of these actions, and connecting the two with minimal distance between corresponding skeletons. In case that users are not satisfied with the tran-

sition between two clips decided by the system, they can draw a few additional frames to ensure smooth transition.

References

Bindiganavale, R., Schuler, W., Llbeck, J.M., Badler, N., Joshi, A.K, Palmer, M., 2000. Dynamically Altering Agent Behaviors Using Natural Language Instructions. Proc. the Fourth International Conference on Autonomous Agents, p.293-300.

Benitez, A.B., Smith, J.R., Chang, S.F., 2000. MediaNet: A Multimedia Information Network for Knowledge Representation. Proc. Conference on IS&T/SPIE.

Cassell, J., Cipolla, R., Pentland, A.(Eds.), 2000. A Framework for Gesture Generation and Interpretation, Computer Vision in Human-Machine Interaction. Cambridge University Press.

Cassell, J., Vilhjalmsson, H., Bickmore, T., 2001. BEAT: the Behavior Expression Animation Toolkit. Proc. SIGGRAPH'01, p.477-486.

Earley, J., 1970. An efficient context-free parsing algorithm. *Communications of the Association for Computing Machinery*, **13**(2):94-102.

Fan, J.P., Elmagarmid, A.K., Zhu, X.Q., Aref, W.G., Wu, L., 2004. ClassView: hierarchical video shot classification, indexing, and accessing. *IEEE Trans. on Multimedia*, **6**(1):70-86. [doi:10.1109/TMM.2003.819583]

Girard, M., 1987. Interactive design of 3-D computer animated legged animal motion. *IEEE Computer Graphics and Applications*, **7**(6):39-51.

Gotsman, C., Surazhsky, V., 2001. Guaranteed intersection-free polygon morphing. *Computers and Graphics*, **25**(1):67-75. [doi:10.1016/ S0097-8493(00)00108-4]

Noma, T., Kai, K., Nakamura, J., Okada, N., 1992. Translating from Natural Language Story to Computer Animation. Proc. SPICIS'92, p.475-480.

Patterson, J.W., Willis, P.J., 1994. Computer assisted animation: 2D or not 2D? *The Computer Journal*, **37**(10):829-839. [doi:10.1093/comjnl/37.10.829]

Vilhjalmsson, H., Cassell, J., 1998. BodyChat: Autonomous Communicative Behaviors in Avatars. Proc. the 2nd Annual ACM International Conference on Autonomous Agents, p.269-276.

Vosinakis, S., Panayiotopoulos, T., 2001. SimHuman: A Platform for Real-time Virtual Agents with Planning Capabilities. Proc. IVA'01, p.210-223.

Webber, B., 1998. Instructing Animated Agents: Viewing Language in Behavioral Terms. NCS 1374, Springer-Verlag, Berlin.

Yu, J.H., Patterson, J.W., 1997. Assessment Criteria for 2D Shape Transformations in Animation. Proc. Computer Animation'97, p.103-112.