CharToon: a system to animate 2D cartoon faces

Paul ten Hagen, Han Noot, Zsófia Ruttkay

CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands
e-mail: {paulh, han, zsofi}@cwi.nl

Abstract

We discuss the CharToon system by which one can interactively construct parametrized 2D drawings and a set of time curves to animate the drawings. The system has been implemented in JAVA and used to produce and animate 2D cartoon faces. The variety and potential usage of such faces is to illustrated by a series of demos.

Keywords: animation, facial modeling, JAVA, constraints.

1. Introduction

Computer facial animation has been a flourishing research topic for more than 25 years, aiming at building models which can be animated and used to (re-)produce facial expressions reflecting emotions and mouth movements for spoken text[7]. The bulk of the efforts has been spent on producing 3D facial models, which both in appearance and in facial movements match faithfully the real human face. In spite of the extensive research, it is still an unsolved problem to achieve full realism on synthetic faces. Besides difficulties with modeling details such as hair and skin texture, it has been empirically proven that facial expressions are very complex, both with regard to co-articulation of several muscles and the evolvement of expressions in time[2,3]. If realism is the goal, the animator has to orchestrate dozens of muscles of the synthetic face. This task, partly because of the lack of sufficient knowledge on the time evolvement of facial expressions, is often solved by using face tracker hardware and software to drive a face model with real performer data[10]. The technological and time requirements are thus still a main, often forbidding factor in using realistic 3D facial animation.

On the other hand, there are several application domains such as avatars and social user interfaces, where it is sufficient to have an expressive and appealing human face, which can be animated real time using limited computer resources. These requirements can be met by dropping realism as aim, and using stylized, cartoon-like 2D faces[1,6,9] . At the same time the application of 2D cartoon faces does not imply decrease in expressiveness or appeal. Just the opposite: the user notices that he is confronted with an 'unrealistic' face, so he will not be disturbed by the lack of faithful details. Moreover, in the cartoon world it is not only possible but fully accepted to apply exaggerated, non-realistic effects which can be used very well to express emotions and state of mind. Last but not least, cartoon-like faces can be artistic and pleasing, besides just being informative.

The above considerations motivated us in the framework of the "Facial Analysis and Synthesis of Expressions" project[4] to develop the CharToon system, a package to define 2D cartoon-like faces and to animate those.

2. Architecture of CharToon

CharToon is a collection of JAVA programs by which one can interactively construct parametrized 2D drawings and a set of time curves to animate the drawings. CharToon consists of 3 components: An animation parameter editor (Animation Editor), a 2D drawing package (Face Editor) and a movieplayer (Face Player). These programs exchange data with each other and possibly with other programs outside CharToon (Figure 1).

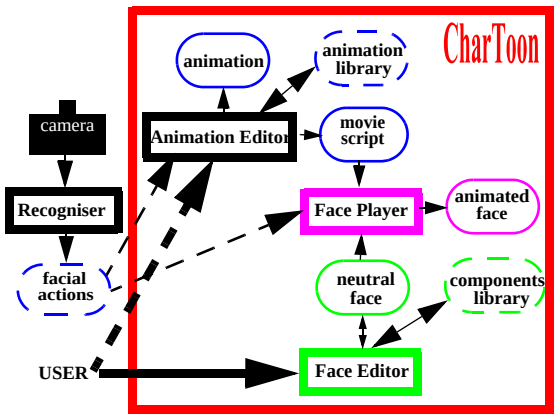

Figure 1: *Architecture of CharToon*

The steps to create an animation in CharToon are:

1) Design the drawing to be animated, using Face Editor. The result is the description of the geometry of the face and an animation parameter description file.
2) Use Animation Editor to specify the time-behavior of the drawing's animation parameters as given by the parameter description file. Save the animation as a script (for later re-use), and make also a movie script.
3) Use Face Player to animate the drawing, with the movie script and the face description as input.

2.1 The face editor

Face Editor is a drawing program for the construction of 2D animated drawings. The program provides a hierarchy of drawing components. Basic building blocks are open or closed polylines and smooth curves which either remain fixed or can be animated via control points. Compound building blocks are objects composed of animated polygons and curves which change shape according to some general principles when the former do. These components can simulate objects with a moving skeleton or things like a human mouth where the outer mouth contour is explicitly animated and the inner contour follows the outer one. At the high end there are ready-made complex objects like an eye with pupil and eyelids. One constructs a drawing by selecting and dropping components. Every component can be edited by, for instance, adding, deleting or displacing points, setting the range of control points etc. Furthermore, the user can create his own library of compound objects. Components may have a scale and visibility parameter which too can be dynamically changed. There is also the possibility to include images into the drawing. In Test mode one can see the effect of dragging control points.

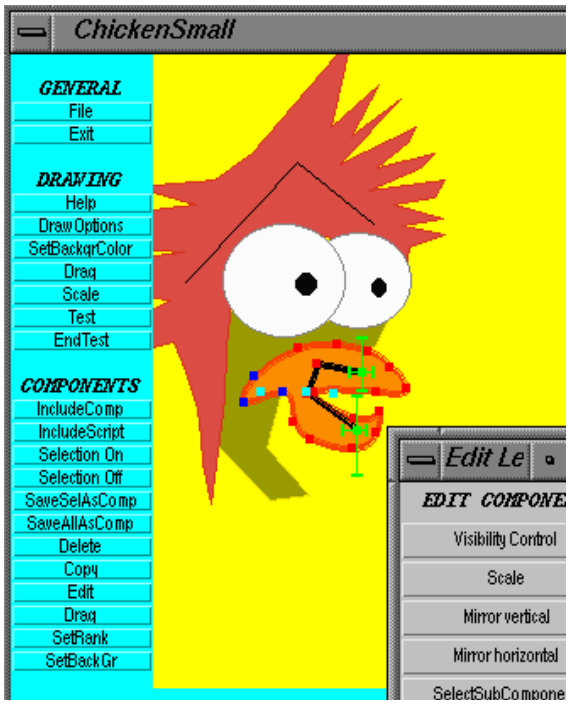

Figure 2: *The Face Editor window, while editing a face. A component, the beek is selected, and part of the pop-up window showing editing menuse applicable to the selection is shown.*

2.2 The animation editor

Animation Editor is a graphical editor for the specification and modification of the values of animation parameters for computer facial (or other) models. Those models must be prepared to communicate the ranges of their parameters and precision to Animation Editor and to respond to the parameter values produced through it. Particularly, Face Editor generates this information when a face is produced.

Animation Editor operates on a window which looks like a musical score (Figure 4). There is a 'staff' for every animation parameter; the lines on each staff reflect the values the parameter can take. The behavior in time of an animation parameter is specified by placing points on its staff. They are inserted, moved and deleted by mouse-operations. One can do cut and paste operations and time- and value scaling on portions of curves and on sets of them. Different views (zoom, hide, overview) of the staves help to focus on controlling certain features. The editor can produce animation scripts (movies) in ascii format by sampling the curves at a rate which is set by the user.

There also is a facility to switch on and off an arbitrary number of audio channels. If the audio is first annotated with (ascii) labels, (e.g. using a program like SGI's soundtrack) Animation Editor will display these labels at their proper place in time. Thereby one can synchronize the audio with the animation parameters.

2.3 The face player

Face Player is a movieplayer to play movies constructed by Animation Editor for a face produced by Face Editor. Face Player takes ascii files as input to generate the pictures with the animated face. Hence it is possible to animate a face via an ascii file real-time.

Figure 3: *The Face Player window with an expression of the Chicken shown to the left.*

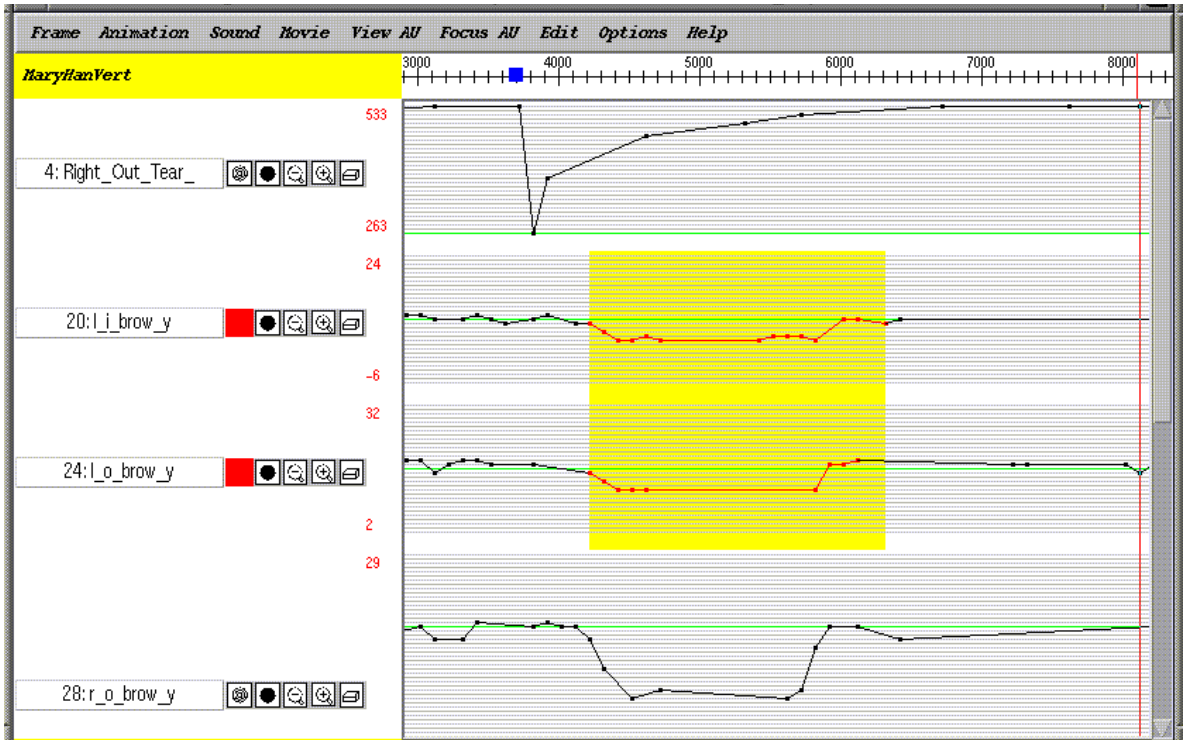

Figure 4: *The Animation Editor window, showing 4 animation parameter staves, the top one with artificial data, the 3 below with captured data. Part of the left eyebrow movements is selected.*

Face Player can be activated as stand alone, or from Animation Editor to be used to test the animation being made, by showing snapshots or playing selected parts. Face Player can dump the images which can be used as input to make and postprocess traditional movie files. It is also possible to play movies accompanied by sound, according to the play instructions for sound channels specified in Animation Editor.

In Face Player every facial component may run in a separate thread, to make it possible to control different parts of a drawing by different sources. This may be useful for special applications such as videoconferencing, and also to exploit parallelism to gain speed. Face Player can be also coupled by socket communication directly with a facial data tracker system.

2.4 Implementation

The first version, with the above explained features, is fully implemented in JAVA, which makes it platform-independent and suitable for Web applications. Our system has been running on Sun and SGI machines and on Windows and MacOS PCs. For small drawings, one can (just) achieve a frame-rate of 25 frames/sec on an otherwise empty Sun Sparc Ultra. On high-end Windows platforms speed is better.

For the next version being developed, constraint-based facilities will be added to Animation Editor[8], allowing the user to define building blocks and express requirements in the form of constraints and helping him to fulfill the current set of requirements during editing.

We plan to improve the audio facilities and extend the system to generate/edit synthetic, visual speech.

3. Demos and applications

The system has been used by three groups of people: the system developers, professional animators and researchers in human ergonomics at a third party. The results of try-outs include the following movies (most of them to be demonstrated during the talk):

Lily is animation of a single subtle drawn female face, to demonstrate how expressions can be achieved by exaggerated and non-realistic features. The artist wanted to have subtle control of the deformation of the features, which was achieved by using 93 control parametes.

NineFaces is a collection of 9 simple human and non-human faces, animated to exhibit some basic expressions and talking. The faces have 6-12 control parameters each. The goal was to show that with simple control (often only scaling and replacement of features) attractive and expressive simple avatar faces can be made. Such faces could be used on web pages, as simple representatives of users in multi-user environments, or in applications for kids.

Mooze is an animal cartoon face, reacting with emotional expressions to a short story. With this animation it is demonstrated how human expressions can be mapped on a non-human face. Also, experiments have been made to fake 3D deformations of the face in 2D.

Han is a face made on the basis of a photograph, but with non-realistic animation.

These animations were all made from scratch, and demonstrate different benefits of faces made by CharToon: expressiveness, ease of control, funny or artistic look. The first three movies have sound, which demonstrates the added value of sonic effects.

CharToon has also been used to make animations on the basis of performer data. Blue dots on a real moving face were traced and 2D coordinates of them were mapped to parameters of the CharToon face to be driven. The blue dots may corrsepond to feature points used in ISO MPEG4 coding[5], but arbitrary set of feature points can be dealt with. The performer data can be used directly as input to Face Player to drive a face, or can be saved as an animation, which can be post-processed in Animation Editor to gain an improved animation. Effects beyond the capability of the face tracker can be achieved by authoring animation for non-tracked features (e.g. eye gaze, tears) or exaggerating the deformation of some features beyond realism. Tracked data has been used in both ways to drive a 'realistic' 3D as well as several cartoon-like 2D faces. One of the cartoon faces has been designed such that it can be animated according to the ISO MPEG standard.

Further examples of possible applications are: animated characters (also on web pages) that react to dynamic data; simple teleconferencing applications; talking faces with speech synthesis or text in speech bubbles.

Acknowledgment

The FASE project is sponsored by STW under nr. CW166.4088. We thank Zs. Paál, B. Kiers, K. Thórisson, M. ten Hagen and F. de Haan for their contributions to the demos. We are indebted to Chris Thórisson for the original idea of CharToon[9] and for the many useful comments on its evolution.

References

1. Brennan, S.: Caricature Generator: The dynamic exaggeration of faces by computer, *LEONARDO*, 18(3), pp.170-178. 1985.
2. Ekman, P., Friesen, W.: *Manual for the Facial Action Coding System*, Consulting Psychologists Press, Inc., Palo Alto, CA, 1978.
3. Essa, I., Basu, S., Darrel, T, Pentlands, A.: Modeling, tracking and interactive animation of faces and heads using input from video, *Proc. of Computer Animation'96*, pp. 68-79.1996.
4. *FASE home page*: http://www.cwi.nl/FASE/
5. *Information Technology - Coding of audio-visual objects: visual*, ISO/IEC 14496-2 Committee Draft, Tokyo, 1998.
6. Martin, D., Torres, J. C.: Alhambra: a system for producing 2D animation, *Proc. of Computer Animation'99*, 1999.
7. Parke, F., Waters, K.: *Computer Facial Animation*, Peters, Wellesley, Massachusetts, 1996.
8. Ruttkay, Zs.: Constraint-based facial animation, *CWI Report INS R9907*, 1999. Also available from ftp://ftp.cwi.nl/pub/CWIreports/INS/INS-R9907.ps.Z.
9. Thórisson, K.R.: ToonFace: A System for Creating and Animating Interactive Cartoon Faces, *M.I.T. Media Laboratory technical report 96-01*, April 1996.
10. Williams, L.: Performance-driven facial animation, *Proc. of SIGGRAPH'90*, pp. 235-242.1990.