LNCS 3031

Andreas Butz
Antonio Krüger
Patrick Olivier (Eds.)

Smart Graphics

4th International Symposium, SG 2004
Banff, Canada, May 2004
Proceedings

Springer

Andreas Butz Antonio Krüger
Patrick Olivier (Eds.)

Smart Graphics

4th International Symposium, SG 2004
Banff, Canada, May 23-25, 2004
Proceedings

Springer

Volume Editors

Andreas Butz
Antonio Krüger
Department of Computer Science, Saarland University
Stuhlsatzenhausweg, Bau 36.1
66123 Saarbrücken, Germany
E-mail: {butz,krueger}@cs.uni-sb.de

Patrick Olivier
Lexicle Ltd.
Innovation Center, York Science Park, York, YO10 5DG, UK
E-mail: plo@lexicle.com

Library of Congress Control Number: 2004105605

CR Subject Classification (1998): I.3, I.2.10, I.2, I.4, I.5, H.5, I.7

ISSN 0302-9743
ISBN 3-540-21977-3 Springer-Verlag Berlin Heidelberg New York

Springer-Verlag is a part of Springer Science+Business Media

springeronline.com

Typesetting: Camera-ready by author, data conversion by Boller Mediendesign
Printed on acid-free paper SPIN: 11005919 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science

For information about Vols. 1–2916

please contact your bookseller or Springer-Verlag

Vol. 3053: J. Davies, D. Fensel, C. Bussler, R. Studer (Eds.), The Semantic Web: Research and Applications. XIII, 490 pages. 2004.

Vol. 3042: N. Mitrou, K. Kontovasilis, G.N. Rouskas, I. Iliadis, L. Merakos (Eds.), NETWORKING 2004, Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications. XXXIII, 1519 pages. 2004.

Vol. 3034: J. Favela, E. Menasalvas, E. Chávez (Eds.), Advances in Web Intelligence. XIII, 227 pages. 2004. (Subseries LNAI).

Vol. 3031: A. Butz, A. Krüger, P. Olivier (Eds.), Smart Graphics. X, 165 pages. 2004.

Vol. 3027: C. Cachin, J. Camenisch (Eds.), Advances in Cryptology - EUROCRYPT 2004. XI, 628 pages. 2004.

Vol. 3026: C. Ramamoorthy, R. Lee, K.W. Lee (Eds.), Software Engineering Research and Applications. XV, 377 pages. 2004.

Vol. 3025: G.A. Vouros, T. Panayiotopoulos (Eds.), Methods and Applications of Artificial Intelligence. XV, 546 pages. 2004. (Subseries LNAI).

Vol. 3024: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 621 pages. 2004.

Vol. 3023: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 611 pages. 2004.

Vol. 3022: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 621 pages. 2004.

Vol. 3021: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 633 pages. 2004.

Vol. 3019: R. Wyrzykowski, J. Dongarra, M. Paprzycki, J. Wasniewski (Eds.), Parallel Processing and Applied Mathematics. XIX, 1174 pages. 2004.

Vol. 3015: C. Barakat, I. Pratt (Eds.), Passive and Active Network Measurement. XI, 300 pages. 2004.

Vol. 3012: K. Kurumatani, S.-H. Chen, A. Ohuchi (Eds.), Multi-Agnets for Mass User Support. X, 217 pages. 2004. (Subseries LNAI).

Vol. 3011: J.-C. Régin, M. Rueher (Eds.), Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. XI, 415 pages. 2004.

Vol. 3010: K.R. Apt, F. Fages, F. Rossi, P. Szeredi, J. Váncza (Eds.), Recent Advances in Constraints. VIII, 285 pages. 2004. (Subseries LNAI).

Vol. 3009: F. Bomarius, H. Iida (Eds.), Product Focused Software Process Improvement. XIV, 584 pages. 2004.

Vol. 3007: J.X. Yu, X. Lin, H. Lu, Y. Zhang (Eds.), Advanced Web Technologies and Applications. XXII, 936 pages. 2004.

Vol. 3006: M. Matsui, R. Zuccherato (Eds.), Selected Areas in Cryptography. XI, 361 pages. 2004.

Vol. 3005: G.R. Raidl, S. Cagnoni, J. Branke, D.W. Corne, R. Drechsler, Y. Jin, C.G. Johnson, P. Machado, E. Marchiori, F. Rothlauf, G.D. Smith, G. Squillero (Eds.), Applications of Evolutionary Computing. XVII, 562 pages. 2004.

Vol. 3004: J. Gottlieb, G.R. Raidl (Eds.), Evolutionary Computation in Combinatorial Optimization. X, 241 pages. 2004.

Vol. 3003: M. Keijzer, U.-M. O'Reilly, S.M. Lucas, E. Costa, T. Soule (Eds.), Genetic Programming. XI, 410 pages. 2004.

Vol. 3002: D.L. Hicks (Ed.), Metainformatics. X, 213 pages. 2004.

Vol. 3001: A. Ferscha, F. Mattern (Eds.), Pervasive Computing. XVII, 358 pages. 2004.

Vol. 2999: E.A. Boiten, J. Derrick, G. Smith (Eds.), Integrated Formal Methods. XI, 541 pages. 2004.

Vol. 2998: Y. Kameyama, P.J. Stuckey (Eds.), Functional and Logic Programming. X, 307 pages. 2004.

Vol. 2997: S. McDonald, J. Tait (Eds.), Advances in Information Retrieval. XIII, 427 pages. 2004.

Vol. 2996: V. Diekert, M. Habib (Eds.), STACS 2004. XVI, 658 pages. 2004.

Vol. 2995: C. Jensen, S. Poslad, T. Dimitrakos (Eds.), Trust Management. XIII, 377 pages. 2004.

Vol. 2994: E. Rahm (Ed.), Data Integration in the Life Sciences. X, 221 pages. 2004. (Subseries LNBI).

Vol. 2993: R. Alur, G.J. Pappas (Eds.), Hybrid Systems: Computation and Control. XII, 674 pages. 2004.

Vol. 2992: E. Bertino, S. Christodoulakis, D. Plexousakis, V. Christophides, M. Koubarakis, K. Böhm, E. Ferrari (Eds.), Advances in Database Technology - EDBT 2004. XVIII, 877 pages. 2004.

Vol. 2991: R. Alt, A. Frommer, R.B. Kearfott, W. Luther (Eds.), Numerical Software with Result Verification. X, 315 pages. 2004.

Vol. 2989: S. Graf, L. Mounier (Eds.), Model Checking Software. X, 309 pages. 2004.

Vol. 2988: K. Jensen, A. Podelski (Eds.), Tools and Algorithms for the Construction and Analysis of Systems. XIV, 608 pages. 2004.

Vol. 2987: I. Walukiewicz (Ed.), Foundations of Software Science and Computation Structures. XIII, 529 pages. 2004.

Vol. 2986: D. Schmidt (Ed.), Programming Languages and Systems. XII, 417 pages. 2004.

Vol. 2985: E. Duesterwald (Ed.), Compiler Construction. X, 313 pages. 2004.

Vol. 2984: M. Wermelinger, T. Margaria-Steffen (Eds.), Fundamental Approaches to Software Engineering. XII, 389 pages. 2004.

Vol. 2983: S. Istrail, M.S. Waterman, A. Clark (Eds.), Computational Methods for SNPs and Haplotype Inference. IX, 153 pages. 2004. (Subseries LNBI).

Vol. 2982: N. Wakamiya, M. Solarski, J. Sterbenz (Eds.), Active Networks. XI, 308 pages. 2004.

Vol. 2981: C. Müller-Schloer, T. Ungerer, B. Bauer (Eds.), Organic and Pervasive Computing – ARCS 2004. XI, 339 pages. 2004.

Vol. 2980: A. Blackwell, K. Marriott, A. Shimojima (Eds.), Diagrammatic Representation and Inference. XV, 448 pages. 2004. (Subseries LNAI).

Vol. 2979: I. Stoica, Stateless Core: A Scalable Approach for Quality of Service in the Internet. XVI, 219 pages. 2004.

Vol. 2978: R. Groz, R.M. Hierons (Eds.), Testing of Communicating Systems. XII, 225 pages. 2004.

Vol. 2977: G. Di Marzo Serugendo, A. Karageorgos, O.F. Rana, F. Zambonelli (Eds.), Engineering Self-Organising Systems. X, 299 pages. 2004. (Subseries LNAI).

Vol. 2976: M. Farach-Colton (Ed.), LATIN 2004: Theoretical Informatics. XV, 626 pages. 2004.

Vol. 2973: Y. Lee, J. Li, K.-Y. Whang, D. Lee (Eds.), Database Systems for Advanced Applications. XXIV, 925 pages. 2004.

Vol. 2972: R. Monroy, G. Arroyo-Figueroa, L.E. Sucar, H. Sossa (Eds.), MICAI 2004: Advances in Artificial Intelligence. XVII, 923 pages. 2004. (Subseries LNAI).

Vol. 2971: J.I. Lim, D.H. Lee (Eds.), Information Security and Cryptology -ICISC 2003. XI, 458 pages. 2004.

Vol. 2970: F. Fernández Rivera, M. Bubak, A. Gómez Tato, R. Doallo (Eds.), Grid Computing. XI, 328 pages. 2004.

Vol. 2968: J. Chen, S. Hong (Eds.), Real-Time and Embedded Computing Systems and Applications. XIV, 620 pages. 2004.

Vol. 2967: S. Melnik, Generic Model Management. XX, 238 pages. 2004.

Vol. 2966: F.B. Sachse, Computational Cardiology. XVIII, 322 pages. 2004.

Vol. 2965: M.C. Calzarossa, E. Gelenbe, Performance Tools and Applications to Networked Systems. VIII, 385 pages. 2004.

Vol. 2964: T. Okamoto (Ed.), Topics in Cryptology – CT-RSA 2004. XI, 387 pages. 2004.

Vol. 2963: R. Sharp, Higher Level Hardware Synthesis. XVI, 195 pages. 2004.

Vol. 2962: S. Bistarelli, Semirings for Soft Constraint Solving and Programming. XII, 279 pages. 2004.

Vol. 2961: P. Eklund (Ed.), Concept Lattices. IX, 411 pages. 2004. (Subseries LNAI).

Vol. 2960: P.D. Mosses (Ed.), CASL Reference Manual. XVII, 528 pages. 2004.

Vol. 2958: L. Rauchwerger (Ed.), Languages and Compilers for Parallel Computing. XI, 556 pages. 2004.

Vol. 2957: P. Langendoerfer, M. Liu, I. Matta, V. Tsaoussidis (Eds.), Wired/Wireless Internet Communications. XI, 307 pages. 2004.

Vol. 2956: A. Dengel, M. Junker, A. Weisbecker (Eds.), Reading and Learning. XII, 355 pages. 2004.

Vol. 2954: F. Crestani, M. Dunlop, S. Mizzaro (Eds.), Mobile and Ubiquitous Information Access. X, 299 pages. 2004.

Vol. 2953: K. Konrad, Model Generation for Natural Language Interpretation and Analysis. XIII, 166 pages. 2004. (Subseries LNAI).

Vol. 2952: N. Guelfi, E. Astesiano, G. Reggio (Eds.), Scientific Engineering of Distributed Java Applications. X, 157 pages. 2004.

Vol. 2951: M. Naor (Ed.), Theory of Cryptography. XI, 523 pages. 2004.

Vol. 2949: R. De Nicola, G. Ferrari, G. Meredith (Eds.), Coordination Models and Languages. X, 323 pages. 2004.

Vol. 2948: G.L. Mullen, A. Poli, H. Stichtenoth (Eds.), Finite Fields and Applications. VIII, 263 pages. 2004.

Vol. 2947: F. Bao, R. Deng, J. Zhou (Eds.), Public Key Cryptography – PKC 2004. XI, 455 pages. 2004.

Vol. 2946: R. Focardi, R. Gorrieri (Eds.), Foundations of Security Analysis and Design II. VII, 267 pages. 2004.

Vol. 2943: J. Chen, J. Reif (Eds.), DNA Computing. X, 225 pages. 2004.

Vol. 2941: M. Wirsing, A. Knapp, S. Balsamo (Eds.), Radical Innovations of Software and Systems Engineering in the Future. X, 359 pages. 2004.

Vol. 2940: C. Lucena, A. Garcia, A. Romanovsky, J. Castro, P.S. Alencar (Eds.), Software Engineering for Multi-Agent Systems II. XII, 279 pages. 2004.

Vol. 2939: T. Kalker, I.J. Cox, Y.M. Ro (Eds.), Digital Watermarking. XII, 602 pages. 2004.

Vol. 2937: B. Steffen, G. Levi (Eds.), Verification, Model Checking, and Abstract Interpretation. XI, 325 pages. 2004.

Vol. 2936: P. Liardet, P. Collet, C. Fonlupt, E. Lutton, M. Schoenauer (Eds.), Artificial Evolution. XIV, 410 pages. 2004.

Vol. 2934: G. Lindemann, D. Moldt, M. Paolucci (Eds.), Regulated Agent-Based Social Systems. X, 301 pages. 2004. (Subseries LNAI).

Vol. 2930: F. Winkler (Ed.), Automated Deduction in Geometry. VII, 231 pages. 2004. (Subseries LNAI).

Vol. 2929: H. de Swart, E. Orlowska, G. Schmidt, M. Roubens (Eds.), Theory and Applications of Relational Structures as Knowledge Instruments. VII, 273 pages. 2003.

Vol. 2926: L. van Elst, V. Dignum, A. Abecker (Eds.), Agent-Mediated Knowledge Management. XI, 428 pages. 2004. (Subseries LNAI).

Vol. 2923: V. Lifschitz, I. Niemelä (Eds.), Logic Programming and Nonmonotonic Reasoning. IX, 365 pages. 2004. (Subseries LNAI).

Vol. 2919: E. Giunchiglia, A. Tacchella (Eds.), Theory and Applications of Satisfiability Testing. XI, 530 pages. 2004.

Vol. 2917: E. Quintarelli, Model-Checking Based Data Retrieval. XVI, 134 pages. 2004.

Preface

The International Symposium on Smart Graphics 2004 was held on May 23–25, 2004 in Banff, Canada. It was the fifth event in a series which originally started in 2000 as a AAAI Spring Symposium. In response to the overwhelming success of the 2000 symposium, its organizers decided to turn it into a self-contained event in 2001. With the support of IBM, the first two International Symposia on Smart Graphics were held at the T.J. Watson Research Center in Hawthorne, NY in 2001 and 2002. The 2003 symposium moved to the European Media Lab in Heidelberg to underline the international character of the Smart Graphics enterprise and its community. The 2004 symposium particularly emphasized the contribution of arts and design to the interdisciplinary field of Smart Graphics and was therefore held at the Banff Centre in Alberta, Canada, an internationally recognized center of creative excellence.

The core idea behind these symposia is to bring together researchers and practitioners from the field of computer graphics, artificial intelligence, cognitive psychology and the fine arts. Each of these disciplines contributes to what we mean by the term "Smart Graphics": the intelligent process of creating expressive and esthetic graphical presentations. While artists and designers have been creating communicative graphics for centuries, artificial intelligence focuses on automating this process by means of the computer. While computer graphics provides the tools for creating graphical presentations in the first place, cognitive sciences contribute the rules and models of perception necessary for the design of effective graphics. The exchange of ideas between these four disciplines has led to many exciting and fruitful discussions and the Smart Graphics symposia draw their liveliness from a spirit of open minds and the willingness to learn from and share with other disciplines.

We would like to thank all authors for the effort that went into their submissions, the program committee for their work in selecting and ordering contributions for the final program, the Banff Centre and our local organizers for providing space and time for hosting the event, and Springer-Verlag Heidelberg for publishing the proceedings in their Lecture Notes in Computer Science.

March 2004

Andreas Butz
Antonio Krüger
Patrick Olivier

Organization

Organizing Committee

Conference Chairs	Andreas Butz (Saarland University, Germany)
	Antonio Krüger (Saarland University, Germany)
	Patrick Olivier (Lexicle Limited, UK)
Local Organization	Sara Diamond (Banff Centre, Canada)
	Brian Fisher (University of British Columbia, Canada)

Program Committee

Maneesh Agrawala (Microsoft Research, USA)
Elisabeth André (University of Augsburg, Germany)
Steven Feiner (Columbia University, New York, USA)
Sidney Fels (University of British Columbia, Canada)
Mark Hansen (University of California, Los Angeles, USA)
Knut Hartmann (University of Magdeburg, Germany)
Takeo Igarashi (University of Tokyo, Japan)
Rainer Malaka (European Media Lab, Germany)
W. Bradford Paley (Digital Image Design, USA)
Bernhard Preim (University of Magdeburg, Germany)
Thomas Rist (DFKI Saarbrücken, Germany)
Andrew Salway (University of Surrey, UK)
Stefan Schlechtweg (University of Magdeburg, Germany)
Sha Xinwei (Georgia Institute of Technology, USA)
Massimo Zancanaro (ITC-IRST Trento, Italy)
Michelle Zhou (IBM T.J. Watson Research Center, USA)

Secondary Reviewers

Blaine Bell, Columbia University
Hrvoje Benko, Columbia University
James T. Klosowski, IBM

Sponsoring Institutions

The 4th International Symposium on Smart Graphics was hosted at the Banff Centre in Alberta, Canada. Organizational support was given by the Banff New Media Institute (BNMI).

Author Index

Table of Contents

Virtual Characters and Environments

Tangible and Hybrid Interfaces

Graphical Interfaces

Poster Presentations

Animating 2D Digital Puppets with Limited Autonomy

Erin Shaw, Catherine LaBore, Yuan-Chun Chiu, and W. Lewis Johnson

Center for Advanced Research in Technology for Education
4676 Admiralty Way, Marina del Rey, 90292, USA
{shaw, labore, joechiu, johnson}@isi.edu
http://www.isi.edu/isd/carte/index.html

Abstract. Digital puppets are animated personas that augment online educational materials with commentary and summaries. Their animations are generated dynamically based on user-authored text, contextual hints, and domain goals, allowing the puppet to act with limited autonomy within specific domains. In this paper we describe the graphical realization and authoring of a 2D digital puppet. We present a *build-once, use-forever* production path that allows us to quickly create new character behaviors, and makes the production of 2D personas feasible. We describe two digital puppet applications and explain how the animation capability is supported across domains.

1 Introduction

Animated characters are becoming increasingly popular for use in conversational interfaces, and as presentation and pedagogical agents. In this paper we describe the graphical realization and authoring of a animated character, or Digital Puppet, which can act with limited autonomy in its particular domain. Digital Puppets augment online educational materials with commentary and summaries, and have been used in both the tutoring and the presentation domain. The work presented here is builds upon our experiences creating pedagogical animated characters, or *guidebots* [19][20][24][28][12][15]. Like guidebots, digital puppets interact with learners via a combination of speech and gestures, making it possible to more accurately model the kinds of dialogs and interactions that occur during apprenticeship learning and one-on-one tutoring. They personify the interface, building upon people's natural tendency to interact socially with computers [23], and can express both thoughts and emotions, in order to portray enthusiasm and empathy. Like real expert tutors that attend to both motivational and cognitive factors, animated characters have the potential to increase learner curiosity and interest, and to offer help and reassurance when they encounter difficulties [16].

The digital puppet system, and in particular, its animation and authoring, is motivated by two drawbacks to developing agents in a research environment: limited budgets and the prototype development cycle. First, we have found limited-animation 2D characters particularly useful for some pedagogical applications; cartoon 2D for younger children and realistic 2D for adult storytelling. However,

A. Butz et al. (Eds.): SG 2004, LNCS 3031, pp. 1–10, 2004.

production demands for animated characters frequently exceed limited research budgets. This is especially true for complex characters, like the puppets, that are designed to speak and gesture in parallel. To make the production of 2D personas feasible, we have developed a *build-once, use-forever* production path that allows us to quickly create new behaviors for a given character [set]. Second, we have found that the prototype iteration cycle for agents precludes a broad focus: Only after completion of the prototype is thought given to re-purposing the agent. XML has mitigated the problem, but hasn't removed the challenge of layperson authoring. Digital puppets were designed to have an integrated user-friendly authoring environment. We wished to separate the content authoring and the context authoring, and did not want to distract users with low-level animation details. Puppets speak exactly the lines they are authored to speak but their gestures and facial animation are generated automatically from contextual hints; thus, the puppets act with limited autonomy.

2 Related Work

With respect to authoring, animated characters fall into three categories, sequentially scripted characters, limited autonomous characters, and autonomous planning agents. Scriptable online characters such as those from Oddcast, Haptek, Microsoft have become increasingly common in the commercial domain where they act as social interfaces or guides [4][13][21]. Because each utterance and gesture must be specified, and because the associated graphics must be kept simple, these interface aids are of limited general use. More sophisticated commercial applications such as those by Conversive and EDrama apply templates and rules to alleviate repetitiveness and provide a degree of autonomy [9][11]. Whereas commercial applications of animated characters assume simple scripting and graphics, research applications explore multimedia presentation and discourse planning, and the animation of complex behaviors[3][17][10].

Digital puppets are limited autonomous characters. They are scripted and sequenced, much like in AutoTutor 2 [10][22], though the puppet-scripting model allows for greater flexibility, supporting parallel actions, behavior choices and behavior probabilities. The main difference, however, is the use of presentation goals to create autonomous gestures. In this respect our puppet is similar to Andre and Rist's PPP Persona, a multipurpose presentation agent [2][3]. However, puppets do not plan, and thus are not intelligent or autonomous agents. Though they are less powerful within any particular domain, it is fairly efficient to transfer puppets to new domains.

The work presented here is compatible with work on automated behavior expression, especially the BodyChat and BEAT systems [6][7]. In BodyChat, gaze behaviors, for example, are fully automated for turn-taking in multi-participant settings. Though the Digital Puppets' behavioral domain is not social communication per se, some of its behaviors are similarly automated, and could be further developed using BEAT. Gestures signifying attentiveness to user actions, deictic gestures indicating on-screen objects such as lesson pages or simulation interfaces, and idle behaviors such as weight and gaze shifts, blinks and head movements are automated. These automatic behaviors may be overridden by authored behaviors, e.g., those which are attached to content-free initiators and responses, in the case of the tutorial agent, or gestures chosen to accompany presentation material, in the case of the presentation agent.

3 2D Character Creation and Animation

3.1 Graphical Realization

The prototypes for the character animation system were Adele [25], the tutor for our case-based clinical practice training simulation, and Carmen and Gina, characters in the interactive drama, Carmen's Bright Ideas [19], which teaches problem-solving skills to mothers of pediatric cancer patients. Adele uses classic limited animation techniques - 2D sequences of frames that always return to a default position. Adele's persona was rendered directly by an artist using a commercial paint program. Carmen's Bright Ideas uses traditional hand drawn character animation, realized as 2D computer graphics using Macromedia's Flash. The agent-controlled characters, Carmen and Gina, use composite Flash libraries for gestures, head and eye positions, and facial expressions.

The 2D behavior-library approach is useful in a number of contexts. Many agent domains are limited in their requirements for visual behavior, relying on synthetic speech to provide necessary depth and breadth. Applications that use a 2D interface look flat in the context of a 3D character, and real time rendering to line-and-fill art is technically challenging. The limited-animation, 2D cartoon character as the basis for a persona seems useful and viable for pedagogical applications, particularly for young learners. Such libraries are most useful when we can load and unload components as needed. In the case of hand-drawn art, if the agent's domain changes, or requires additional behaviors, the same artist must be re-engaged, and spend many more hours in the labor intensive task of hand drawing the new frames.

Research budgets and circumstances make this approach less than ideal. To make our 2D personas feasible, we opted for a *build-once, use-forever* production path. The character for the first Digital Puppet system, Skip, was built using Alias Wavefront's Maya, a commercial 3D animation program. Any competent character animator can create new behaviors for Skip, and rendered output will always match the original art, so libraries of behavior can be built up over time. In addition, a procedural system such as Maya allows for the generalization of behaviors across characters.

To maximize behavioral capability, we render and animate the character in layers, allowing us to achieve independent control of important features. Eyes, brows, mouth and body can each be rendered separately so that the character can raise its brows while speaking, speak while gesturing, shift its gaze and weight when attending to user actions, etc. This capability introduces a new problem - registering the 2D layers, in particular, the head and body. In 3D, as the gesticulating body rotates around, the 2D perspective projection of the head-body registration point changes. In 2D, the separately-rendered head positions remain stationary. Because the camera is fixed for all animations, we determine a projection matrix, apply it to the 3D registration point, and then compute a translation that aligns the 2D registration point on the head with the one on the body.

3.2 Creating Behaviors

Behaviors can thus be created to meet the evolving needs of the agent. We begin with a basic library that makes the character come alive: weight and gaze shifts, saccades, blinks and head movements. In the case of the presentation agent, Skip, behaviors specific to expository speaking and the use of visual aids are added on top of these. Maya animation sequences are created and rendered by Cambridge Animation System's Swiffworks shader at base body orientations (e.g., toward the user, toward the screen) to Macromedia's Flash format, which provides indexing and editing capabilities. For additional flexibility, we use scalable vector graphics (SVG) as our target display format. This allows us to scale the images to the optimal size for the screen layout and employ a variety of cameras, from full body shots for gestures to closeups for conversational behaviors. To avoid the latency currently associated with parsing a document object model, we convert the final SVG files to the portable network graphics format, gaining speed but losing the ability to further scale animations.

Beyond the basic behavior library, the animators must create behaviors related to a particular domain that will be used by the character. We collected and studied video footage of human presenters, and human tutors working with users of computer applications. The reference footage permits the animators to anticipate what kinds of moves will be indispensable. In the presentation domain deictic gestures combined with beats are the most common gestures. Deictics are easy to define procedurally and generically useful. Future additions to a presentation puppet might include tutorial gestures related to orienting the user and using the application interface.

The image libraries are loaded into the puppet animation engine. XML descriptions of gestures, comprised of sequenced and procedural frames, are generated by a graphical animation editing tool. Table 1 shows the output of the editor. Background animations are performed automatically when the engine is idling, and sending a text string automatically results in lip synced text-to-speech synthesis at any of several different head positions.

4 Animation Authoring

Digital puppets require several types of input: Users must supply text for the puppet to speak and environments must supply contextual hints that enables an inferencing module to select appropriate behaviors. The selection is informed by video analysis, presentation techniques [1], and facial communication displays

Table 1. A viseme sequence and a body gesture example in Widget.xml.

```
<head>
    <!-- Frames for viseme sequence for position 000 -->
    <frame filename="head/head-000-phonemes{0001..0012}.svgz">
        <mount name="body" coordinate="0, 25.632, 0.351" />
    </frame> ...
</head>
<body>
    <!-- Frames for body gesture weight shift -->
    <gesture name="weight-shift">
        <sequence subtype="middle-to-left-to-right"
            progression="weight-shift0001..0009" />
    </gesture>
</body>
```

[8]. Based on the input, the inferencing module generates a runtime script and sends it to the animation engine for display.

We use contextual hints to select behaviors in two ways. In every domain, there are naturally occurring high-concept goals. For example, in the presentation domain there are the components of the presentation task, *introduce, explain, etc.* and in the tutoring domain there is the flow of the application task. The markup for scripting the puppet's behavior at this level is shown in Table 2. The *PreIntro* is the initial context; in this state the camera might zoom in and the puppet might look around and speak. When the viewer begins the presentation, the puppet moves to the *Introduction* state, which specifies that the puppet first glance at the button, and then orient itself with its body facing toward the viewer before speaking. There are idle and end states as well. A type of *choices* will choose one [group] of the gestures, while a type of *gspeak* will specify a facial animation or gesture to perform while speaking.

The more interesting mapping of behaviors occurs at lower levels, where gestures are mapped to contexts and goals, for example, *elaborate* in the presentation domain and *increase politeness* in the tutoring domain, as well as to sentence fragments, words, and patterns, such as exclamations and numbers. This intention-based mapping is similar to the approach of Andre and Rist [3], who use presentation goals to select their animation schema. A gesture is executed randomly and probabilistically after the sentence starts. Several examples are shown in Table 3.

The animation engine executes the script commands asynchronously and in parallel, depending on the command. The script uses a special-purpose XML schema that is narrowly defined for digital puppets and was not created as a general standard. Though the 2D puppet schema is compatible with other standards, most animation languages – Affective Presentation Markup Language(APML), Multimodal Utterance Representation Markup Language (MURML), Rich Representation Language(RRL), Virtual Human Markup Language(VHML) – control 3D avatar movement and communication in virtual environments. Similarly, there are 3D facial animation standards that focus on realistic control.

Table 2. The mapping of high-level contextual information to gestures.

⟨context name=PreIntro⟩ ⟨s type=camera name=closer⟩ ⟨s type=gspeak text="Come {%talkGesture :default%} in and sit down."⟩ ⟨s type=wait for=second value=5⟩ ⟨s type=gspeak text="We {%turnhead:044%} are waiting for people to arrive."⟩ ⟨s type=choices⟩ ⟨c type=pose gesture=talkGesture subtype=default⟩ ⟨c type=nothing⟩ ⟨c type=group⟩ ⟨g type=turnhead to=044 wait=true⟩ ⟨g type=turnhead to=000 wait=true⟩ ⟨/c⟩ ⟨/s⟩ ⟨s type=wait for=second value=5⟩ ⟨s type=jump context=PreIntro⟩ ⟨/context⟩	⟨context name=Intro⟩ ⟨s type=turnhead to=044 wait=true⟩ ⟨s type=wait for=second value=1⟩ ⟨s type=turnhead to=000 wait=true⟩ ⟨s type=turnbody to=000 wait=true⟩ ⟨insert_point⟩ *insert authored text* ⟨/insert_point⟩ ⟨/context⟩

Table 3. The mapping of low-level text and contextual information to gestures.

⟨word⟩ ⟨entry word = "this" gesture = "lifthandpoint" subtype = "default" probability = "100" /⟩ ⟨entry word = "but|but," gesture = "shiftweightright" subtype = "default" probability = "50" /⟩ ⟨/word⟩ ⟨sentence⟩ ⟨entry fragment = "in fact" gesture = "talkgesture" subtype = "default" probability = "100" /⟩ ⟨/sentence⟩	⟨context⟩ ⟨entry context = "Introduction" gesture = "talkgesture1" subtype = "default" probability = "50" /⟩ ⟨/context⟩ ⟨goal⟩ ⟨entry goal = "clain" gesture = "lookaround" subtype = "default" probability = "50" /⟩ ⟨/goal⟩

5 Puppet Applications

5.1 Puppets for Developing Oral Presentation Skills

Digital puppets were created as part of an application to teach fourth and fifth grade students how to construct oral presentations. The goal of the project was to improve science learning through the use of digital puppets in peer teaching and collaborative learning settings. Because of the user base, a user-friendly WYSIWYG puppet authoring tool, shown in Figure 1 on the left, was developed in parallel with the animation engine, instead of as an afterthought. The tool enables users to annotate Web pages and use the annotations to generate commentary and animation.

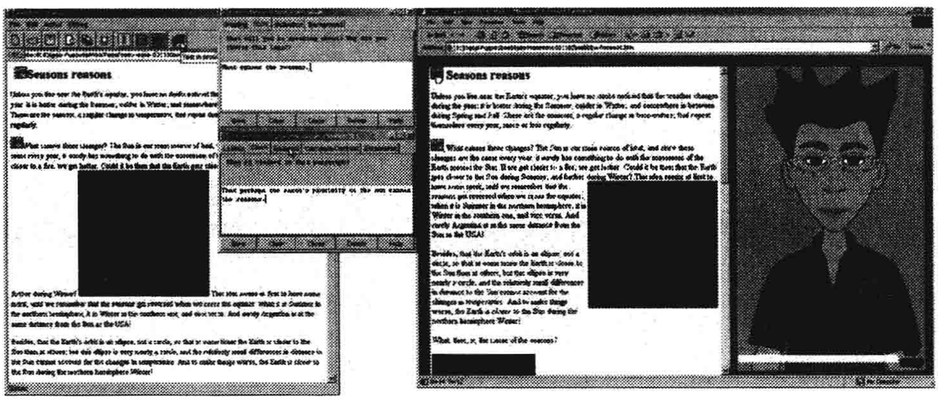

Fig. 1. On the left is the WYSIWYG presentation authoring tool with a Web page and two authoring boxes displayed. On the right is the resulting browser-based puppet presentation. Interactive buttons have been inserted in the Web page.

Our approach to authoring the puppets was based on Rhetorical Structure Theory (RST) [18] and then generalized to accommodate any presentation goal. By constraining the domain, we can reasonably infer the intent of the author with respect to the authored commentary. For example, in the introduction, a user must provide text to motivate the topic. Based on the context and goal, the inferencing module might select behaviors approprate to motivation and introductions, such as face viewer, perform inviting gesture, and amplify voice.

Figure 1, left, shows the digital puppet user authoring tool, which contains a Web page about the topic to be presented. Clicking on a paragraph of text brings up a small authoring windows, shown at center, that displays the particular concepts that the students must address in their explanations. These goals (e.g., *lesson, claim, evidence*) can be tailored for different contexts (i.e., *introduction, explanation, and conclusion.*) Students provide an introduction, a conclusion and