Advances in Intelligent Systems and Computing

Volume 770

Series editor
Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland
e-mail: kacprzyk@ibspan.waw.pl
The series “Advances in Intelligent Systems and Computing” contains publications on theory, applications, and design methods of Intelligent Systems and Intelligent Computing. Virtually all disciplines such as engineering, natural sciences, computer and information science, ICT, economics, business, e-commerce, environment, healthcare, life science are covered. The list of topics spans all the areas of modern intelligent systems and computing such as: computational intelligence, soft computing including neural networks, fuzzy systems, evolutionary computing and the fusion of these paradigms, social intelligence, ambient intelligence, computational neuroscience, artificial life, virtual worlds and society, cognitive science and systems, Perception and Vision, DNA and immune based systems, self-organizing and adaptive systems, e-Learning and teaching, human-centered and human-centric computing, recommender systems, intelligent control, robotics and mechatronics including human-machine teaming, knowledge-based paradigms, learning paradigms, machine ethics, intelligent data analysis, knowledge management, intelligent agents, intelligent decision making and support, intelligent network security, trust management, interactive entertainment, Web intelligence and multimedia.

The publications within “Advances in Intelligent Systems and Computing” are primarily proceedings of important conferences, symposia and congresses. They cover significant recent developments in the field, both of a foundational and applicable character. An important characteristic feature of the series is the short publication time and world-wide distribution. This permits a rapid and broad dissemination of research results.

Advisory Board
Chairman
Nikhil R. Pal, Indian Statistical Institute, Kolkata, India
e-mail: nikhil@isical.ac.in

Members
Rafael Bello Perez, Universidad Central “Marta Abreu” de Las Villas, Santa Clara, Cuba
e-mail: rbellop@uclv.edu.cu

Emilio S. Corchado, University of Salamanca, Salamanca, Spain
e-mail: escorchado@usal.es

Hani Hagras, University of Essex, Colchester, UK
e-mail: hani@essex.ac.uk

László T. Kóczy, Széchenyi István University, Győr, Hungary
e-mail: koczy@sze.hu

Vladik Kreinovich, University of Texas at El Paso, El Paso, USA
e-mail: vladik@utep.edu

Chin-Teng Lin, National Chiao Tung University, Hsinchu, Taiwan
e-mail: ctlin@mail.nctu.edu.tw

Jie Lu, University of Technology, Sydney, Australia
e-mail: Jie.Lu@uts.edu.au

Patricia Melin, Tijuana Institute of Technology, Tijuana, Mexico
e-mail: cpmelin@hafsamx.org

Nadia Nedjah, State University of Rio de Janeiro, Rio de Janeiro, Brazil
e-mail: nadia@eng.uerj.br

Ngoc Thanh Nguyen, Wroclaw University of Technology, Wroclaw, Poland
e-mail: Ngoc-Thanh.Nguyen@pwr.edu.pl

Jun Wang, The Chinese University of Hong Kong, Shatin, Hong Kong
e-mail: jwang@mae.cuhk.edu.hk

More information about this series at http://www.springer.com/series/11156
Foreword

It is a great pleasure to invite you to the Fourth International Conference on Big Data Applications and Services (BigDAS 2017), which will be held in Tashkent, Uzbekistan on August 15–18, 2017. BIGDAS 2017 is hosted by National University of Uzbekistan and Korea Big Data Service Society.

The main topic of BigDAS 2017 is “Power to Change the World, Big Data”. The Big Data has become a core technology to provide innovative solutions in the many fields such as health care, manufacturing, social life, etc. The aim of BigDAS 2017 is to present the innovative results, encourage academic and industrial interaction, and promote collaborative research in Big Data and digital information management worldwide. We expect that publications of our conference will be a cornerstone for the further related research and technology improvements in the field of Big Data.

For BigDAS 2017, we accepted many high-quality papers, which will be presented in oral and poster sessions. The organizing committee of BigDAS 2017 has prepared 11 technical sessions. The topics of technical sessions are

- Big Data Models and Algorithms,
- Big Data Architectures,
- Big Data Applications,
- Big Data Mining and Analysis,
- Big Data Security,
- Big Data Visualization,
- Social Network Analysis,
- Internet of Things (IoT) and Health care,
- Big Data in Industry,
- Big Data in Government, and
- Big Data in Business.
We hope you will find the results presented during the conference useful and inspiring for your future research.

We would like to express our sincere thanks to invited speakers, organizing committee, and authors for their valuable contributions to the conference.

Suwon, South Korea

Joo-Yeon Lee
General Co-chair of BigDAS 2017
Ajou University

Tashkent, Uzbekistan

Avazjon Marakhimov
General Co-chair of BigDAS 2017
National University of Uzbekistan

August 2017
Preface

The International Conference on Big Data Applications and Services (BigDAS) aims to address the need of the academic community and industry about Big Data. It encourages academic and industrial interaction and promotes collaborative research in Big Data applications and services by bringing together academics, government, and industry professionals to discuss recent progress and challenges in Big Data applications and services. Moreover, BigDAS also serves as a platform for theoreticians and practitioners to exchange their original research ideas on academic or industrial aspects of Big Data applications and services, present their new findings or innovative results on theoretical or practical aspects of Big Data, share their experiences on integrating new technologies into products and applications, discuss their work on performing Big Data applications and services in real-life situations, describe their development and operations of challenging Big Data related systems, and identify unsolved challenges.

Since the First International Conference on Big Data Applications and Services (1st BigDAS 2015), three BigDAS conferences have been held in the following venues:

- **1st BigDAS 2015**: Seogwipo KAL Hotel, Jeju Island, South Korea, on October 20–23, 2015;
- **2nd BigDAS 2016**: Korea Software HRD Center, Phnom Penh, Cambodia, on January 22–27, 2016; and

With their success, the Fourth International Conference on Big Data Applications and Services (4th BigDAS 2017) is held in the following venues:

- **4th BigDAS 2017**: National University of Uzbekistan, Tashkent, Uzbekistan, on August 15–18, 2017.

The BigDAS 2017 was held in Tashkent, Uzbekistan (which is located in the center of Great Silk Road). The conference was hosted by (i) The Korea Big Data Service Society and (ii) National University of Uzbekistan. It was organized by (i) Bigdata
Research Institute, Chungbuk National University; (ii) New Industrial Convergence R&D Center, Ajou University; (iii) Inha University in Tashkent; and (iv) Korea China Yeouido Leaders Forum. It was sponsored by (i) Electronics and Telecommunications Research Institute (ETRI); (ii) UNISEM Co., Ltd.; and (iii) WIZCORE, Inc. The program consists of the following events:

- Four keynote speeches on big data in industry and government:
  - “The Fourth Industrial Revolution and New Industry Development Directions” by Marn-ki Jeong (ex-First Vice Minister of the Ministry of Trade, Industry and Energy, South Korea)
  - “Prediction Markets a Computational Mechanism for Aggregating Information” by Sarvar Abdullaev (Inha University in Tashkent, Uzbekistan)
  - “Big Data Approach and Challenges in Government” by Myoung Hee Kim (President of National Computing and Information Service (NCIS), Ministry of Interior, South Korea)
  - “Initiation of an Innovation Ecosystem: Uzbekistan’s Road Towards Building Silicon Valley” by Bokhodir Ayupov (Ministry of Information Technologies and Communications, Uzbekistan)

- eight technical sessions with 54 regular and short paper presentations on the following topics: (i) Big Data models and algorithms, (ii) Big Data architectures, (iii) Big Data applications, (iv) Big Data mining and analysis, (v) Big Data security, (vi) Big Data visualization, (vii) social network analysis, and (viii) Internet of things (IoT) and health care;

- three poster sessions with eight poster paper presentations on the following topics: (i) Big Data applications, (ii) Big Data in industry, and (iii) Big Data in business; and

- two workshops: (i) E-government and Big Data, and (ii) Big Data composition.

For BigDAS 2017, we have recruited many international experts in Big Data applications and services to join our team of international committee. As a result, our Committee consists of professionals from different parts of the world such as Bangladesh, Canada, China, Japan, South Korea, and Uzbekistan. This committee has done an excellent job in organizing the conference and selecting a collection of 54 high-quality regular and short papers, as well as 8 poster papers. Among these papers, a small number of them were selected and revised to be included in the current volume.

BigDAS 2017 would not have been possible without the help and effort of many people and organizations. We thank BigDAS 2017 Organizing Committee members, especially Honorary Co-chairs (S. H. Han and N. Yusupbekov), General Co-chairs (J.-Y. Lee and A. R. Marakhimov), and Organizing Co-chairs (K.-H. Yoo, W.-S. Cho, and Y.-S. Kim) for their valuable advice and suggestions toward the conference. We are grateful to other members for their professionalism and dedication in different aspects of this conference, including the selection of papers presented at the conference and the further selection of papers published in the
current volume. We also express our thanks to authors and non-author participants of this conference. We also thank A. Nasridinov for local arrangement. Last but not least, we thank the staff at Springer for their help in publishing the current volume.

Incheon, South Korea

Winnipeg, Canada
August 2017

Wookey Lee  
Program Co-chair of BigDAS 2017

Carson K. Leung  
Program Co-chair of BigDAS 2017
Organization

Honorary Co-chairs
Sun Hwa Han, KISTI, South Korea
Nadirbek Yusupbekov, Tashkent State Technical University, Uzbekistan

General Co-chairs
Joo-Yeon Lee, Ajou University, South Korea
Avazjon R. Marakhimov, National University of Uzbekistan, Uzbekistan

Program Co-chairs
Wookey Lee, Inha University, South Korea
Carson K. Leung, University of Manitoba, Canada

Organizing Co-chairs
Kwan-Hee Yoo, Chungbuk National University, South Korea
Wan-Sup Cho, Chungbuk National University, South Korea
Yoo-Sung Kim, Inha University, South Korea

Local Arrangement Co-chairs
Aziz Nasridinov, Chungbuk National University, South Korea
Sarvar Abdullaev, Inha University in Tashkent, Uzbekistan
Azizbek Marakhimov, Dongguk University, South Korea

Publicity Co-chairs
Sang-Hyun Choi, Chungbuk National University, South Korea
Seon-Phill Jeong, United International College, China

Web Co-chairs
Young-Koo Lee, Kyung Hee University, South Korea
Phyoung-Jung Kim, Chungbuk Provincial College, South Korea
Industry Co-chairs
Tae-Sung Kim, Chungbuk National University, South Korea
Soo-young Chae, National Security Research Institute (NSRI), South Korea

Industry Liaison Co-chairs
Duck-Keun Park, Wizcore, South Korea
Jung-Tae Kang, YURA, South Korea

Demo/Poster Co-chairs
Su-young Chi, ETRI, South Korea
Nakhoon Baek, Kyungpook National University, South Korea

Registration Co-chairs
Young-Ho Park, Sookmyung Women’s University, South Korea
Yong-Ik Yoon, Sookmyung Women’s University, South Korea

Tutorial Co-chairs
Jin-Ho Kim, Kangwon National University, South Korea
Supratip Ghose, UITS, Bangladesh

Special Session Co-chairs
Eun-mi Choi, Kookmin University, South Korea
Jin-young Choi, Ajou University, South Korea
Takeshi Kurata, AIST, Japan
Tianhong Pan, Jiangsu University, China
Masaki Aono, Tohohashi University of Technology, Japan

Sponsor Co-chairs
Seong Joon Yoo, Sejong University, South Korea
Jong-Seok Myong, Oracle Inc., South Korea

Secretariat
Jong-Min Yoon, KITECH, South Korea

Co-hosts

Korea Big Data Service Society

NATIONAL UNIVERSITY OF UZBEKISTAN

hbbs01@hanmail.net
Organizers

Sponsors

hbbs01@hanmail.net
Contents

3D Volume Visualization System Based on GPUs for Medical Big Data ................................................................. 1
Joseph Yoon, Oh-Soek Kown, Jae-Gyun Cheong, Keon-Kuk Park
and Young-Bong Kim

A Fire Flame Simulation Scheme with Massively Parallel Processing ................................................................. 9
Byeonguk Im and Nakhoon Baek

A Framework for Calculating Damages of Personal Information Leakage Accidents .............................................. 17
Dong-Hee Lee and Tae-Sung Kim

A Hierarchical Structure for Representing 3D Respiration Organ Models ............................................................... 23
Eath Manith, Chan Park and Kwan-Hee Yoo

An Effective Method for Detecting Outlying Regions in a 2-Dimensional Array .................................................. 37
Ki Yong Lee and Young-Kyoon Suh

An Effective Recall-Oriented Information Retrieval System Evaluation ................................................................. 43
Justin JongSu Song, Jiyoung Lim, Chris Soo-Hyun Eom and Wookey Lee

Chentry: Automated Evaluation of Students’ Learning Progress for Entry Education Software .......................... 51
Jeong-Hun Kim, Jong-Hyeok Choi, Uygun Shadikhatovaev,
Aziz Nasridinov and Ki-Sang Song

Constrained Big Data Mining in an Edge Computing Environment ................................................................. 61
Carson K. Leung, Deyu Deng, Calvin S. H. Hoi and Wookey Lee
Constrained Frequent Pattern Mining from Big Data Via Crowdsourcing ........................................... 69
Calvin S. H. Hoi, Dantiyal Khowaja and Carson K. Leung

Data-Driven Prediction of Ship Destinations in the Harbor Area Using Deep Learning .......................... 81
Kwang Il Kim and Keon Myung Lee

Design and Implementation of a Sunshine Duration Calculation System with Massively Parallel Processing ........................................... 91
Woosuk Shin and Nakhoon Baek

Developing 3D Annotation Features for 3D Digital Textbooks .............. 99
Jin-Uk Jeong, Gae-A Ryu, Mihye Kim and Kwan-Hee Yoo

Efficient Mining of Time Interval-Based Association Rules .................. 121
Ki Yong Lee and Young-Kyoon Suh

Investigating the Role of Musical Genre in Human Perception of Music Stretching Resistance ................. 127
Jun Chen and Chaokun Wang

Keyword-Based Metadata Modeling for Experimental Omics Data Dissemination ........................................... 139
Kyoung Soon Hwang and Keon Myung Lee

Nonlinear Time-series Mining of Social Influence ......................... 151
Thinh Minh Do, Yasuko Matsubara and Yasushi Sakurai

PEGASEF: A Provenance-Based Big Data Service Framework for Efficient Simulation Execution on Shared Computing Clusters ........... 175
Young-Kyoon Suh, Ki Yong Lee and Nakhoon Baek

Preference-Aware Music Recommendation Using Song Lyrics ............ 183
Sein Jang, Battulga Lkhagvadorj and Aziz Nasridinov

Real-Time Smart Safe-Return-Home Service Based on Big Data Analytics .......................................................................................... 197
Gae-A Ryu, Jae-Won Lee, Ji-Sung Jeong, Mihye Kim and Kwan-Hee Yoo

Author Index .............................................................................. 211

Subject Index ........................................................................... 213
Developing 3D Annotation Features for 3D Digital Textbooks

Jin-Uk Jeong, Gae-A Ryu, Mihye Kim and Kwan-Hee Yoo

Abstract Digital textbooks (DTs) have been established as the major media for future and smart education in South Korea. In general, DTs are implemented using two-dimensional (2D) web-based platforms with embedded 3D content, including images, motion graphics, animations, and video and audio. Recently, these 2D-based DTs have evolved into a 3D-based interface by adopting various types of 3D virtual environments. Accordingly, a number of 3D DTs have been developed; however, these are mainly focused on the implementation of the basic features of DTs, including displaying text and images, viewing pages, zooming in and out, indexing, moving to a certain page, and finding a specific text or object. Furthermore, these have not yet been comprehensively implemented, and further development is required to provide more diverse input and annotation features to facilitate better dynamic interaction between students and DTs. Here, we introduce 3D annotation features that were designed and implemented to enable users to freely annotate on 3D DTs with various forms of input, facilitating dynamic user interactions. These include stylus writing, underlining, highlighting, drawing lines and shapes, and the insertion of textboxes, sound and video files, and voice memos. Students may highlight text or place notes to give commentary on the content of DTs using stylus writings, lines, shapes, text, multimedia files, and voice memos to aid their studying. The 3D annotation features were developed using the eXtensible 3D (X3D) standard, which is an XML-based international standard file format for representing scenes and 3D objects in computer.

J.-U. Jeong · M. Kim
Department of Computer Science Education, Catholic University of Daegu, Gyeongsansi, Gyeongbuk, South Korea
e-mail: usmlike@nate.com

M. Kim
e-mail: mihyekim@cu.ac.kr

G.-A. Ryu · K.-H. Yoo (✉)
Department of Digital Informatics and Convergence, Chungbuk National University,
410 Seongbongro, Heungukgu, Cheongju, Chungbuk, South Korea
e-mail: khyoo@chungbuk.ac.kr

G.-A. Ryu
e-mail: reflexive@gmail.com

© Springer Nature Singapore Pte Ltd. 2019

hbbs01@hanmail.net
graphics, extended from the virtual reality modeling language (VRML), along with the Java Applet and JavaScript. We aim to enhance student engagement in the learning process by supporting various forms of dynamic interaction features via a natural user interface. We believe that the approaches we describe provide a viable solution to enable 3D annotation on DTs.

Keywords Digital textbook · 3D digital textbook · X3D-based 3D digital textbook · 3D annotation

1 Introduction

With the continuing development of information technology (IT), including pervasive computing [1], cloud computing [2], and smart media devices [3, 4], educational systems are increasingly attracting innovative media for education and smart learning in the future, leading to an intelligent and adaptive learning system that will enhance learner efficiency and engagement [5]. Accordingly, in July 2011, the South Korean government established the Smart Educational Strategy Action Plan for a twenty-first century smart education system, and has launched a project to convert all existing printed textbooks (PTs) used in elementary, middle, and high schools into digital textbooks (DTs) suitable for smart learning environments by 2015 [5–8]. DTs are curriculum-based electronic textbooks that contain educational material used for teaching and learning in schools [9].

DTs are typically implemented with two-dimensional (2D) platforms and include all the functionality of PTs while providing a wider range of learning activities and environments by including various types of digital media features. With the adoption of 3D virtual environments in education systems, DTs are currently evolving to a 3D interface [10]. As a result, 3D DTs have been developed using the extensible 3D (X3D) standard by Hong et al. [10, 11]. The principle advantage of 3D DTs is the ability to realize a more realistic virtual environment, thus enabling users to perform various additional activities. For example, users can place virtual notes on a 3D object, as in temporarily attaching post-it notes. In addition, a 3D DT can present textbook content naturally without distortion of text or images because rotation, transition, and zoom operations can be freely applied with the 3D interface. Furthermore, dynamic interactions between users and 3D DTs are possible as both the content of DTs and objects embedded in the DTs exist in the same 3D space. Thus, a more realistic environment can be provided to students, enhancing their interest and engagement in the learning process [11]. The 3D DT developed by Hong et al. [10, 11] was for the subject of general computing in high schools in South Korea. The main features implemented in the 3D DT included displaying text and images, viewing pages, zooming in and out, indexing, moving to a particular page, finding specific text or objects, and embedding and playing 3D motion graphics, animations, and audio. The 3D DT was valuable in that the DT prepared guidance and directions for the development of a X3D-based DT. However, the 3D DT functionality demonstrated
to date has focused on implementing the basic features of a DT in a 3D interface. Thus, further developments are required, particularly regarding annotations, to enable students to interact with DTs in a more fully interactive manner.

In this paper, we introduce 3D annotation features that were developed to allow users to freely annotate on 3D DTs via various forms of interaction. These include stylus writing, underlining, highlighting, drawing shapes, and the insertion of textboxs, multimedia files, and voice memos. Specially, head-mounted display (HMD) devices can be used to give interaction between users and the 3D DT. The content of this paper is based on the work described in [12] and the concept of the proposed approach was partially introduced in [13].

The remainder of this paper is organized as follows. In Sect. 2, we review the existing literature on 3D DTs and 3D annotation techniques. In Sect. 3, we present the basic architectures of the 3D DT for the process of 3D annotations, in addition to describing the implementation strategies of 3D annotations in relation to X3D nodes. In Sect. 4, we present 3D annotation features that were implemented, and Sect. 5 summarizes the paper and outlines directions for future work.

2 Related Works

2.1 3D DTs

3D content can enhance students’ learning effectiveness by overcoming the limitations of the user experience in a 2D flat space, and also by facilitating learning engagement and motivation [11]. For these reasons, 3D content has been used in various ways in the field of education. In 2007, An and Kim developed a 3D web-based courseware for textbook content that enabled students to observe transportation routes used in the real-world and to experience the changes of civilizations using the Virtual Reality Modeling Language (VRML) and 3D Max [14]. In 2008, Yang et al. developed a science DT that shows the cut planes of multi-layer structures of 3D objects in real time as they are cut [15]. As mentioned in the previous section, Hong et al. also developed a 3D DT with various forms of 3D objects using X3D features for the subject of general computing [10, 11]. These 3D-based DTs can enhance students’ motivation for learning by supporting dynamic user interaction with 3D objects [11]. However, these DTs focused on the implementation of the basic features of a DT in a 3D-based interface. Thus, further development is required, particularly for annotations to support more realistic user interactions.

Table 1 shows the 3D DT features implemented in the DT developed by Hong et al. [10, 11]. These are based on existing 2D DT features [7, 8]. The 3D DT features were implemented using the X3D functionalities.
<table>
<thead>
<tr>
<th>Features</th>
<th>2D DT Features</th>
<th>3D DT features based on X3D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display</td>
<td>Display texts and images</td>
<td>• Implement 3D DT texts and images in the same way as a PT using the Text Node and ImageTexture Node of X3D</td>
</tr>
<tr>
<td></td>
<td>Single/double-page view</td>
<td>• Display DT content in single-page or double-page view using the settings feature of the ViewPoint Node</td>
</tr>
<tr>
<td></td>
<td>Zoom in and out</td>
<td>• Using the mouse scroll wheel</td>
</tr>
<tr>
<td></td>
<td>Fit a page to the screen</td>
<td>• Using the X3D viewpoint initialization key (&quot;Esc&quot; key)</td>
</tr>
<tr>
<td></td>
<td>Fit a page to width/height</td>
<td>• Using the settings feature of the ViewPoint Node</td>
</tr>
<tr>
<td></td>
<td>Page scroll</td>
<td>• Using the movement and viewpoint of the DT screen</td>
</tr>
<tr>
<td></td>
<td>Indicate page thickness</td>
<td>• Indicate page thickness with an overlapped page view</td>
</tr>
<tr>
<td></td>
<td>Display page number</td>
<td>• Show page thickness changes whenever a page is turned</td>
</tr>
<tr>
<td></td>
<td>Text hiding</td>
<td>• By adjusting the transparency using the ScalarInterpolator Node</td>
</tr>
<tr>
<td>Input</td>
<td>Writing</td>
<td>• String input using the StringSensor Node</td>
</tr>
<tr>
<td></td>
<td>Memo</td>
<td>• Using the JavaScript memo function</td>
</tr>
<tr>
<td></td>
<td>View memo</td>
<td>• Display memos on a web browser</td>
</tr>
<tr>
<td></td>
<td>Enter memo</td>
<td>• Input through a prompt dialog box</td>
</tr>
<tr>
<td></td>
<td>Auto-save memo</td>
<td>• Save memos using web browser cookies</td>
</tr>
<tr>
<td></td>
<td>Open memo</td>
<td>• Using the &quot;get cookie&quot; function of a web browser</td>
</tr>
<tr>
<td></td>
<td>Delete memo</td>
<td>• Delete memos using web browser cookies</td>
</tr>
<tr>
<td></td>
<td>Create a table of contents (TOC) for memos</td>
<td>• By saving in the form of a list of memos</td>
</tr>
<tr>
<td></td>
<td>Adjust memo window</td>
<td>• By adjusting the size, move, and position of a memo window</td>
</tr>
<tr>
<td>Move</td>
<td>Move via TOC</td>
<td>• Move between Sections using the Anchor Node</td>
</tr>
<tr>
<td></td>
<td>Move via the previous and next page buttons</td>
<td>• Page turning using the OrientationInterpolator Node</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Page connection by the Inline Node</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Movement button connection using the TouchSensor Node</td>
</tr>
<tr>
<td></td>
<td>Page tuning</td>
<td>• Display the page turning motion</td>
</tr>
<tr>
<td>Search</td>
<td>Search content</td>
<td>• By linking to general search engines using the Anchor Node</td>
</tr>
<tr>
<td></td>
<td>Sound effect</td>
<td>• Using the AudioClip Node</td>
</tr>
</tbody>
</table>

(continued)
2.2 **3D Annotation**

3D virtual spaces are used in a number of fields, including games, politics, and society. Even an election campaign can be represented via a virtual reality service in Second Life. However, further development of 3D virtual technologies is also required for 3D object manipulation and collaboration in 3D spaces, as the number of applications of 3D spaces grows [16]. One of the major requirements for further development is that of 3D annotation techniques on 3D objects or 3D virtual spaces.

Annotation techniques in 3D virtual spaces have primarily been developed using 3D sketches to support collaboration between coworkers, especially in the field of design engineering. Bourguignon et al. developed an interactive sketch annotation system for 3D spaces for applications in education, design, architecture, and fashion, where 3D sketches were used to annotate 3D models [17]. Do and Gross developed a sketch-based interaction system for 3D domains using the Space Pen, which supports user annotation in the form of sketching written in Java 3D [18]. This system enables designers to mark or to add drawings on 3D models. Boujut and Dugdale developed a 3D annotation tool that supports collaborative design between designers by allowing them to leave comments on CAD models [16]. The tool was developed using Java 3D and SolidWorks, a CAD tool. Sookmyung Women’s University has also performed studies on sketched-based annotation techniques for 3D object manipulation and cooperative work in 3D virtual spaces by resolving the recognition and phase difference of 3D objects [19]. Sin et al. improved sketch-based annotation techniques by exploring a sketch-box annotation technique suitable for 3D virtual spaces [20]. All of these reports focused on the development of more effective annotation techniques for 3D models using a sketch-based interface to support collaboration between designers in engineering fields. Hence, the annotation features of these sketch-based approaches were limited, and it is not straightforward to apply them to other fields.

Here, we report a different annotation approach for X3D-based DTs to allow students to freely annotate DTs using various types of input via X3D. Here, the term
annotation can be defined as notes, memos, and drawings attached or inserted on the
content of 3D DTs. For example, highlighted text or virtual post-it notes to provide a
commentary on the content of DTs using lines, circles, texts, and voice with a view
to facilitating study are included in the annotation features in our approach.

X3D is an XML-based international standard file format for representing 3D
computer graphics objects in VRML [21, 22]. There exist a number of 3D standard
file formats, including Universal 3D (U3D), JAVA 3D, and Coin3D. Among these,
only X3D is an XML-based file format comparable with existing XML DT formats
with a rich variety of functionalities [11]. Moreover, it is comparable with VRML 97
and supports a dedicated platform independent web browser (i.e., BS Contact) with
capability for integration with external programs. For these reasons, we used X3D as
the file format for 3D, which was also used for the work reports in Refs. [10, 11]. The
X3D feature set includes 3D graphics, 2D graphics, animation, audio and video, user
interaction, navigation, user-defined objects, scripting, and networking, in addition
to physical simulation, geospatial positioning, CAD geometry, layering, support for
programmable shaders, and particle systems [22]. A more detailed description of
X3D can be found at the Web3D Consortium [21]. The X3D features are provided
in the object hierarchy, with nodes grouped into components with the related objects
[22].

3  Design of 3D Annotation Features for 3D DTs Using X3D

3.1  Main Features of 3D Annotations for X3D-Based 3D DTs

We determined the 3D annotation features for inclusion in DTs based on the input
and multimedia functions of 2D-based DTs from the Korea Education and Research
Information Service (KERIS) [7, 8]. We then designed the annotation features using
X3D nodes on the X3D-based DT developed by Hong et al. [10, 11]. Table 2 shows
the 3D annotation features we designed for the X3D-based DTs and the X3D nodes
used for the implementation of each feature. The memo annotation feature was
implemented with a memo pad [10, 11] and, in addition, we support a voice memo
feature.

3.2  X3D Nodes for 3D Annotation Features

To realize 3D annotations in an X3D-based DT, we utilized the TouchSensor, Script,
and Transform nodes of X3D and extended them appropriately for each annotation
feature.

- **TouchSensor.** The interactions of X3D consist of sensing user actions and prompt-
ing appropriate responses. We used the TouchSensor node of X3D to recognize
Table 2  3D annotation features for 3D DTs and the corresponding X3D nodes

<table>
<thead>
<tr>
<th>Features</th>
<th>2D DT features</th>
<th>3D nodes for each 3D annotation feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stylus writing</td>
<td>Writing on 3D DTs using a stylus pen or mouse</td>
<td>LineSet</td>
</tr>
<tr>
<td>Highlight</td>
<td>Highlighting using a highlight pen</td>
<td>TriangleStripSet</td>
</tr>
<tr>
<td>Textbox</td>
<td>Input strings using a keyboard</td>
<td>Text</td>
</tr>
<tr>
<td>Underline or line</td>
<td>Underlining using a stylus pen or mouse</td>
<td>LineSet</td>
</tr>
<tr>
<td>Shapes</td>
<td>Creating and drawing various shapes on 3D DTs</td>
<td>LineSet</td>
</tr>
<tr>
<td>Multimedia</td>
<td>Insert sound and video files on 3D DTs</td>
<td>MovieTexture</td>
</tr>
<tr>
<td>Voice memo</td>
<td>Insert voice memos (voice recording) using a microphone</td>
<td>Java Applet</td>
</tr>
</tbody>
</table>

Table 3  TouchSensor node used to obtain 3D coordinates

```
<Transform DEF="touchPlane">
  <Shape>
    <Box size="8 12 0.0015"/>
    <Appearance>
      <Material transparency="1"/>
    </Appearance>
  </Shape>
  <TouchSensor DEF="Input" enabled="false"/>
</Transform>
```

user action and movements on a 3D space using a mouse. To obtain 3D coordinate values for the annotations, we added a touch sensor (i.e., touch panel) over the 3D DT, and defined the touch sensor in the Transform node, as shown in Table 3. For keyboard events, we used the StringSensor node.

- **Script.** Application programmers can be used to define additional fields and features when instantiating the Script node within a scene in X3D. That is, programmers may define the interface and behavior of a Script node. In X3D, sensors detect various kinds of user interaction and produce events to ROUTE within a scene. Here, programmers define how the events describing user interaction are handled using a programming language that the X3D browser supports. We created Script nodes for each annotation feature, defining the interface and behavior of the Script nodes to handle incoming events using JavaScript.

- **Menu.** We implemented seven annotation features, as listed in Table 2, which were provided via a menu interface. Thus, users must select an annotation type from the given menu to make an annotation.
3.3 3D Annotation Process

Figure 1 shows the overview of our 3D annotation process for X3D-based DTs. First, a user should select the annotation type (icon) from the annotation menu. The system then initializes the TouchSensor node and the respective Script node corresponding to the annotation type, and produces events to ROUTE within a scene. When a user clicks on a starting position on the 3D DT using a mouse, TouchSensor reads the 3D coordinate values of the start position and executes the respective Script node. Then, the Script creates an X3D node corresponding to the selected annotation type. For example, when a user selects a stylus writing in the annotation menu, the Script will create a LineSet node, as described in Table 2. The Script then assigns these coordinate values to the property values of the respective X3D node in real time according to the movements of the mouse, drawing and displaying the annotation results on the 3D DT. Here, moving the mouse while the sensor is activated (i.e., mouse dragging motion) will cause continuous generation of output events. Then, the Script adds the created X3D node (e.g., LineSet, TriangleStripSet, Text, MovieTexture nodes) to the Transform node of X3D. The X3D nodes encoded in the Transform node for annotations will be viewed on the DT, where the annotation was inserted. That is, X3D will display the annotations on the DT, along with the origin contents.

When a keyboard event occurs, the StringSensor node is activated. To change a line thickness, we use the PlaneSensor node of X3D. The X3D nodes registered in the Transform node can be deleted and the annotation color and thickness for annotations may also be altered.

![Diagram](image)

**Fig. 1** Overall process of 3D annotations on an X3D-based DT
3.4 Design for 3D Annotation Features

- **Stylus Writing, Underline, and Shapes.** We designed these annotations using the LineSet node of X3D. Thus, the internal processes of these annotations are similar. The differences lie in the control of mouse clicks. In the case of stylus writing, all the 3D coordinates followed by the mouse moving through a 3D space are continuously transmitted to the LineSet node created by the respective Script node, because all the data for the trajectory of the mouse are required. For an underline or a line, two 3D coordinate points, one from where the mouse was clicked for the first time and the other from where the mouse was released, will be transmitted to draw a line. In the case of shape drawing, whenever a vertex of a shape is clicked, the 3D coordinate values of the vertex will be transmitted to the corresponding LineSet node. When a vertex input from a mouse click is completed, a shape will be drawn by connecting the vertices. If a point is clicked within a predefined arbitrary range around the first clicked point, the point will be recognized as a signal of the final vertex input and will be identified with the first clicked vertex. For example, if the vertices of a shape are clicked from 1 to 5 in order, and then point 6 is clicked, the annotation system will identify point 6 with point 1 and will draw a pentagram by connecting points from 1 to 5 sequentially, as shown in Fig. 2a.

- **Highlight.** To represent lines, X3D uses vector formats that cannot control line thickness. Thus, we process a highlight from a rectangle using the TriangleStripSet node of X3D. The TriangleStripSet node creates a rectangle by connecting the three coordinate points that were taken most recently. However, the coordinate points followed by the mouse moving for a highlight cannot generate a rectangle. Accordingly, we use additional points to generate a rectangle in the direction that the mouse moves, as shown in Fig. 2b, c. The vertical extent of the rectangle will be determined by the thickness of the highlight.

- **Textbox.** We designed textbox input using the Text node of X3D. A textbox is created at a 3D point where the mouse was clicked, obtained using the TouchSensor node. Strings are then entered in the textbox using the StringSensor node that provides a string-based interface from the user’s keyboard. Each key press is collected until the enter key is pressed, which is used to signal the end of the input.
• **Multimedia.** Users can annotate on the content of the DT by inserting audio or video files. The location where a multimedia file is inserted is obtained using the TouchSensor node (i.e., from a mouse click) and the file path from the StringSensor node. The multimedia file is inserted to the DT using the MovieTexture node of the texture component of X3D. To play multimedia files, the startTime and stopTime properties of the texture component are used.

• **Voice Memo.** To record a voice memo, hardware control is necessary to control a microphone and a sound card. However, X3D does not support hardware control features. For this reason, we developed a voice memo using the java.applet and linked it to the DT using the Anchor node of X3D. We implemented a voice recording feature using the AudioFormat and DataLine classes, and a playback feature using the AudioClip class in Java. Java applets are set to disable the hardware control because of security issues. Thus, using the keytool and jarsigner utilities included in the Java Development Kit, we convert the applet implemented for voice memos to a Signed applet that allows users to control hardware.

## 4 Implementation Results of 3D Annotation Features

We implemented the 3D annotation features listed in Table 2 using the 3D DT developed by Hong et al. [10, 11]. The annotation system was developed using a Windows 7 platform with the X3D browser (BS Contact) and X3D Edit for the X3D nodes, Java Applet, and JavaScript.

Figure 3 shows the screen configuration of the 3D DT, including the Table of Contents (TOC) on the left side of the screen, textbook contents in the double-page view, and the menu for 3D annotations. We added the annotation menu to the existing DT using the Inline node of X3D. The annotation menu at the bottom of the screen includes a palette for selecting colors, a scroll bar for selecting the thickness, an eraser for deleting annotations, a pen for stylus writing, and a highlighter pen for highlighting, as well as icons for textboxes, lines, shapes, and video and video file inputs, as indicated in the figure. A feature icon for voice memos was included, as shown in the top right of the screen, shown in Fig. 3.

• **Stylus Writing and Highlighting.** The most frequently used annotation features in a DT are expected to be stylus writing and highlighting. Figure 4a shows examples of stylus writing and Fig. 4b shows examples of highlighting. Table 4 lists the JavaScript source code used to obtain a block of an X3D node when the mouse is clicked. The received block is added to X3D using the Browser.createVrmFromString() function and the variable btnClick is set to true to maintain the status of mouse click. If the mouse button is released, then coordNo is set to 0. Table 5 lists a function that transfers the 3D coordinates to an X3D node for stylus writing.

Users may change the highlighting color and thickness. Table 6 shows the function used to generate 3D coordinates for highlighting representation. To represent
Fig. 3  Screen configuration of the 3D DT

Fig. 4  Examples of (a) stylus writings and (b) highlighting

Table 4  JavaScript source code to handle mouse clicks

```javascript
function isActive(value, timestamp) {
  if (value == true) {
    AnnotationNode.addChildren = Browser.createVrmlFromString(getBLOCK());
    btnClick = true;
    nodeNo = nodeNo + 1;
  } else {
    btnClick = false;
    coordNo = 0;
  }
}
```

highlighting, the function obtains and generates 3D coordinates based on mouse movement in real-time, checking the color information, and transmits these data to the TriangleStripSet node. If the mouse is moved to the right, then right = true; otherwise, right = false.
Table 5  Function to transfer 3D coordinates to an X3D node for stylus writing

```javascript
function freeLineCoordChange(value, timestamp) {
  if (btnClick == true) {
    AnnotationNode.children[nodeNo1].children[0].geometry.coord.point[coordNo] = new SFVec3f(value.x, value.y, value.z-0.0001);
    AnnotationNode.children[nodeNo-1].children[0].geometry.color.color[coordNo] = selectedColor;
    AnnotationNode.children[nodeNo-1].children[0].geometry.vertexCount[0] = coordNo + 1;
    coordNo = coordNo + 1;
  }
}
```

Table 6  The function used to generate 3D coordinates for the highlight representation

```javascript
function fluCoordChange(value, timestamp) {
  if (btnClick == true) {
    AnnotationNode.children[nodeNo-1].children[0].geometry.coord.point[coordNo] = new SFVec3f(value.x, value.y, value.z-0.0001);
    AnnotationNode.children[nodeNo-1].children[0].geometry.color.color[coordNo] = selectedColor;
    if (AnnotationNode.children[nodeNo-1].children[0].geometry.coord.point[coordNo].x - AnnotationNode.children[nodeNo-1].children[0].geometry.coord.point[coordNo-2].x > 0) {
      right = true;
    } else if (AnnotationNode.children[nodeNo-1].children[0].geometry.coord.point[coordNo].x - AnnotationNode.children[nodeNo-1].children[0].geometry.coord.point[coordNo-2].x < 0) {
      right = false;
    }
  } else if (right == true) {
    AnnotationNode.children[nodeNo-1].children[0].geometry.coord.point[coordNo+1] = new SFVec3f(value.x, value.y-thick, value.z-0.0001);
    AnnotationNode.children[nodeNo-1].children[0].geometry.color.color[coordNo+1] = selectedColor;
    AnnotationNode.children[nodeNo-1].children[0].geometry.stripCount[0] = coordNo + 2;
  } else {
    AnnotationNode.children[nodeNo-1].children[0].geometry.coord.point[coordNo+1] = new SFVec3f(value.x, value.y+thick, value.z-0.0001);
    AnnotationNode.children[nodeNo-1].children[0].geometry.color.color[coordNo+1] = selectedColor;
    AnnotationNode.children[nodeNo-1].children[0].geometry.stripCount[0] = coordNo + 2;
  }
  coordNo = coordNo + 2;
}
```

- **Textbox.** Users may also enter text (i.e., strings) via the keyboard. If the textbox icon from the annotation menu is selected, a textbox will be created at the position where the mouse was clicked, which is determined using the TouchSensor node. Strings can then be entered in the textbox using the keyboard via the StringSensor node. Each character key press is collected until the enter key is pressed. Figure 5 shows examples of textbox inputs and Table 7 lists the functions implemented to handle the textbox input.
Fig. 5 Examples of textbox inputs with different colors and sizes

Table 7 The functions used to process a textbox input

<table>
<thead>
<tr>
<th>Function</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>function textboxActive()</code></td>
<td><code>function textboxActive(value, timestamp) {</code></td>
</tr>
<tr>
<td></td>
<td><code>AnnotationNode.addChildren_BROWSER.createVRMLFromString(getBLOCK());</code></td>
</tr>
<tr>
<td></td>
<td><code>nodeNo = nodeNo + 1;</code></td>
</tr>
<tr>
<td></td>
<td><code>}</code></td>
</tr>
<tr>
<td><code>function textboxCoordChange()</code></td>
<td><code>function textboxCoordChange(value, timestamp) {</code></td>
</tr>
<tr>
<td></td>
<td><code>BtnLoc = value;</code></td>
</tr>
<tr>
<td></td>
<td><code>}</code></td>
</tr>
<tr>
<td><code>function textboxInput(value)</code></td>
<td><code>function textboxInput(value) {</code></td>
</tr>
<tr>
<td></td>
<td><code>AnnotationNode.children[nodeNo-1].children[0].children[0].geometry.string = new</code></td>
</tr>
<tr>
<td></td>
<td><code>MFString(value);</code></td>
</tr>
<tr>
<td></td>
<td><code>}</code></td>
</tr>
</tbody>
</table>

Fig. 6 Examples of a lines and b shapes drawn using a combination of stylus inputs and lines

- **Line and Shape.** Users can underline text where they wish to emphasize something, or may draw a line to connect two arbitrary points on the 3D DT. After selecting the line icon from the annotation menu, the user may click to define a starting point of a line using the mouse, and then drag the mouse until the endpoint of the line and release the mouse button. A line will then be drawn by connecting the starting and end points. The left screen of Fig. 6 shows examples of lines and Table 8 lists the functions used to implement line annotation using JavaScript.

Users may also draw a variety of shapes on the 3D DT. The right screen of Fig. 6 shows examples of shapes drawn using a combination of stylus and line input. Table 9 lists the function used to implement drawing shapes. When the mouse is clicked for the first time, the variable firstClick will be set to false and the first clicked point will be assigned to the variable StartPoint. If the assigned StartPoint exists within a
Table 8  The function used for drawing lines

```javascript
function lineCoordChange(value, timestamp) {
    if (btnClick === true) {
        AnnotationNode.children[nodeNo-1].children[0].geometry.coord[coordNo]
            = new SFVec3f(value.x, value.y, value.z-0.0001);
        AnnotationNode.children[nodeNo-1].children[0].geometry.color.color[coordNo] = selectedColor;
        AnnotationNode.children[nodeNo-1].children[0].geometry.vertexCount[0] = coordNo + 1;
        coordNo = 1;
    }
}
```

Fig. 7  Examples of a inserting a sound file and b inserting a video file

predefined arbitrary range, the point will be recognized as a signal for the final vertex input and will be identified with the first clicked point of the shape.

- **Audio and Video.** Users can also insert sound, music, or video files into the 3D DT. These features were implemented using the MovieTexture node of X3D. A user may click a point at which they want to place a multimedia file after selecting the sound or video icons from the annotation menu, and a URL input screen will appear on the DT. When the file path for the multimedia file is entered, the system will add play and pause buttons at the point where the mouse was clicked. In the case of a video file, the system will also add a video screen, as shown in the bottom right screen in Fig. 7. Tables 10 and 11 show the functions implemented to insert sound and video files into the DT, respectively.

- **Voice Memo.** X3D does not support hardware control features, such as a microphone and sound card, so we recorded voice memos using a Java applet and linked
Table 9  Function to process the shape annotation on the 3D DT

```javascript
function polyCoordChange(value, timestamp) {
  if (btnClick === true) {
    AnnotationNode.children[nodeNo-1].children[0].geometry.coord.point[coordNo] = new SFVec3f(value.x, value.y, value.z-0.0001);
    AnnotationNode.children[nodeNo-1].children[0].geometry.color.color[coordNo] = selectedColor;
    AnnotationNode.children[nodeNo-1].children[0].geometry.vertexCount[0] = coordNo + 1;
    if (firstClick === true) {
      startPoint = value;
      firstClick = false;
    }
    else {
      if ((startPoint.x+rad > value.x && startPoint.x-rad < value.x) && (startPoint.y+rad > value.y && startPoint.y-rad < value.y)) {
        AnnotationNode.children[nodeNo-1].children[0].geometry.coord.point[coordNo] = startPoint;
        AnnotationNode.children[nodeNo-1].children[0].geometry.color.color[coordNo] = selectedColor;
        AnnotationNode.children[nodeNo-1].children[0].geometry.vertexCount[0] = coordNo + 1;
        firstClick = true;
        coordNo = 0;
      }
    }
    btnClick = false;
  }
  else {
    if(firstClick === false) {
      AnnotationNode.children[nodeNo-1].children[0].geometry.coord.point[coordNo+1] = new SFVec3f(value.x, value.y, value.z-0.0001);
      AnnotationNode.children[nodeNo-1].children[0].geometry.color.color[coordNo+1] = selectedColor;
      AnnotationNode.children[nodeNo-1].children[0].geometry.vertexCount[0] = coordNo + 2;
    }
  }
}
```

them to the 3D DT using the Anchor node of X3D. Figure 8 shows screen captures for recording a voice memo. If the icon for the voice memo shown in the top right of the DT screen is selected, an applet window will open, as shown in Fig. 8. First, the user should enter a file name in the bottom of the window to save the audio recording, and then start recording the voice memo by pressing the Capture button. During recording, the Stop button will become enabled, while the Capture and Play buttons become disabled. Pressing the Stop button terminates the recording, and the voice memo will then be saved using the file name that was supplied earlier, and will be displayed in the memo list shown in the middle-left of the window. A voice memo can be played using the Play button after selecting it from the memo list.
Table 10 Function to insert sound files to the 3D DT

```javascript
function getMUSIC() {
    var addfile = 'Transform {
        children [
            Transform {
                translation 6.9 6.8 0.065 children [ DEF MovieStart+'+MnodeNo+'' Shape {
                    geometry Box { size 0 0 0 }
                    appearance Appearance {
                        texture DEF Movie'+MnodeNo+'' MovieTexture {
                            repeatS FALSE repeatT FALSE url{" +fileLoc=" } } } ] } ]
            Transform {
                translation'+BtnLoc.x+' '+BtnLoc.y+' '+BtnLoc.z+0.1+' children [Shape {
                    geometry Box { size 0.4 0.4 0 }
                    appearance Appearance {
                        texture ImageTexture {
                            url ["../images/Play.png"] } } ]
                DEF MovieStart+'+MnodeNo+'' TouchSensor {} ] }]
            Transform {
                translation '+BtnLoc.x+0.6+' '+BtnLoc.y+0.6+' '+BtnLoc.z+0.1+' children [Shape {
                    geometry Box { size 0.4 0.4 0 }
                    appearance Appearance {
                        texture DEF btn stop'+MnodeNo+'' ImageTexture {
                            url ["../images/Stop.png"] } } ]
                DEF MovieStop+'+MnodeNo+'' TouchSensor {} ] }
            ]
            ROUTE MovieStart+'+MnodeNo+''.touchTime TO Movie+'+MnodeNo+''startTime
            ROUTE MovieStop+'+MnodeNo+''.touchTime TO Movie+'+MnodeNo+''stopTime
        };
    return addfile;
}
function mediaCoordChange(value, timestamp) {
    BtnLoc = value;
}
function keyboard(value) {
    if (!value.equals("in")) {
        KeyString[0] = 'URL : ' + value;
        fileLoc = value;
    }
}
function multimediaIsActive(value,timestamp){
    if (value == true){
        StEnable = true;
        KeyString[0] = 'URL : ';
    }
}
```
Table 11  The function used to insert video files to the 3D DT

```javascript
function getMOVIE() {
  var addfile
  = `Transform {
       children [
         Transform {
           translation "+(BtnLoc.x-2.9)+" +(BtnLoc.y-0.4)+"+(BtnLoc.z+1)"
           children [ DEF MovieStart+MnodeNo+"
             Shape {  
               geometry Box { size 5 3.75 0 }
               appearance Appearance {
                 texture DEF Movie+MnodeNo+"
               MovieTexture { 
                 repeatS FALSE repeatT FALSE url["+fileLoc+" ] } } ] } ]
        Transform {
          translation +(BtnLoc.x+"+(BtnLoc.y+"+(BtnLoc.z+0.1)+"
          children [  
            Shape {  
              geometry Box { size 0.4 0.4 0 }
              appearance Appearance {
                texture ImageTexture {url ["../images/btn start.jpg"]
          } } ]
        DEF MovieStart+MnodeNo+` TouchSensor { }
     ] }]
      Transform {
        translation +(BtnLoc.x+"+(BtnLoc.y-1)+"+(BtnLoc.z+0.1)+"
        children [  
          Shape {  
            geometry Box { size 0.4 0.4 0 }
            appearance Appearance {
              texture ImageTexture {url ["../images/btn stop.jpg"]
            } } ]
          DEF MovieStop+MnodeNo+` TouchSensor { }
       ] }]
    ROUTE MovieStart+MnodeNo+`touchTime TO Movie+MnodeNo+`.startTime
    ROUTE MovieStop+MnodeNo+`touchTime TO Movie+MnodeNo+`.stopTime
  });
  return addfile;
}
```

Fig. 8  An example recording a voice memo
Table 12 lists the functions associated with the voice memo annotation. The captureAudio() function records audio by acquiring signals from a microphone using the getAudioFormat() function and saves them into the given file. The playAudio() function is for playing a saved voice memo.

Figure 9 shows examples of 3D objects in the DT annotated using stylus writing and lines, viewed from different angles. Such annotated features can provide a more realistic learning environment for students, enhancing their interest and motivation in the learning process.

5 Conclusion

The principle advantage of 3D annotations is the ability to realize a more realistic virtual education environment, thus enabling students to perform various additional learning activities on 3D DTs. We have described the design and implementation of 3D annotations on X3D-based DTs. The objective was to develop 3D annotation features that can enable students to freely place notes into 3D DTs via various forms of interactions. The aim was to enhance student interest and engagement in the learning process by supporting a more realistic environment and various forms of
Fig. 9 3D annotation examples on the 3D DT

dynamic interaction between users and DTs. The developed 3D annotation features include stylus writing, highlighting, underlining, drawing shapes, and the insertion of textboxes, multimedia files, and voice memos. Students can freely highlight text or place notes to give commentary on the content of DTs using text, stylus writings, lines, circles, shapes, multimedia files, or voice to aid their studying.

Although the development of these 3D annotations are valuable and may provide some guidance and directions for 3D interactions between users, there are a number of fundamental issues that require further development. First, X3D cannot control hardware because it is itself configured as a web-based 3D environment. As a result, X3D is not able to save annotation data directly to disk. To handle this issue, an additional interface is required, which has issues for portability. Second, in X3D, the perception of an object has a higher priority than the recognition of a sensor. Thus, when a node is generated for an annotation and overlaps with a sensor, abnormalities may occur in displaying the contents. Third, there exist issues in handling 2D-based input devices, such as the mouse and stylus writing pen, whereas 3D annotation is made in a 3D space. Thus, the development of input devices suitable for 3D environments is desirable. However, this study is a significant step towards the development of 3D annotation techniques for 3D, providing an intuitive user interface, which may facilitate an enhanced smart learning environment.

Acknowledgements This research was financially supported by the ICT R&D program of MSIP/IITP [2016001210021001, Standard Development of HMD Based VR Service Framework] and by the “Standard Technology Improvement Program” through the Ministry of Trade, Industry and Energy (No. 10053638).
References
