Research and Practice on 3D Model Retrieval and Visualization Technology in Institutional Repositories: Postprint

Authors: Wu Zhiqiang, Zhongming Zhu, Liu Wei, Zhang Wangqiang, Yao Xiaona

Date: 2017-11-08T00:00:00+00:00

Abstract

Purpose: To address the extended functional requirements of institutional repositories, this study investigates 3D model retrieval and display technologies, designing and implementing content-based 3D model retrieval and Web3D display and interaction capabilities. Method: Building upon the open-source 3D model retrieval algorithm from National Taiwan University, we employ off-screen rendering to obtain orthogonal projections of models, extract 3D model features, generate 3D model thumbnails using Java3D, and implement Web-based online display and interaction of 3D models using Three.js. Result: Users can retrieve 3D models by submitting a URL or uploading, and on the 3D model browsing page, they can rotate and zoom the model using the mouse for detailed examination. Limitation: While retrieval results and efficiency can meet current basic requirements, the recall and precision rates of the model need further improvement, necessitating continuous attention to 3D model retrieval technologies and optimization of retrieval functions. Conclusion: Applying this model to the CSpace system can effectively expand support capabilities for 3D models and provide users with more diverse methods for 3D model retrieval and application.

Full Text

Preamble

ChinaXiv Collaborative Journal, Issue 1, 2017
Research on 3D Model Retrieval and Display Technology for Institutional Repositories
Wu Zhiqiang, Zhu Zhongming, Liu Wei, Zhang Wangqiang, Yao Xiaona
(Lanzhou Library, Chinese Academy of Sciences, Lanzhou 730030)

Abstract

[**Objective**] To address the functional expansion needs of institutional repositories, this study investigates 3D model retrieval and display technologies, designing and implementing content-based 3D model retrieval along with Web3D display and interactive capabilities. [**Methods**] Building upon the open-source 3D model retrieval algorithm from National Taiwan University, we modified the approach to obtain orthogonal projections of models through off-screen rendering for feature extraction. Java3D was employed to generate 3D model thumbnails, while Three.js was utilized to achieve web-based online display and interaction. [**Results**] Users can retrieve 3D models by submitting URLs or uploading files, and can rotate and zoom models via mouse interaction for detailed examination. [**Limitations**] While current retrieval results and efficiency meet basic requirements, the recall and precision rates need further improvement. Continuous attention to 3D model retrieval technologies is needed to optimize and supplement retrieval functionality. **Conclusion** When applied to the CSpace system, this approach effectively expands 3D model support capabilities and provides users with more diverse methods for 3D model retrieval and application.

Keywords: Institutional Repository; CSpace; 3D Model Retrieval; Web3D; Three.js
Classification Number: G250

Introduction

Institutional Repositories (IR) serve as critical infrastructure for research and educational institutions to store, organize, and manage their intellectual output. With the development of open access initiatives and web technologies, an increasing number of institutions have established their own IR systems to organize and manage research outputs while promoting dissemination and utilization through open sharing agreements, thereby expanding academic influence. Currently, most IRs primarily support traditional text-based content such as journals, conference proceedings, and reports, while offering inadequate support for non-textual resources like images, audio, video, and 3D models [1]. As the fourth generation of multimedia formats, 3D models offer more natural and intuitive representation of objects compared to sound, images, and video, while also providing interactive capabilities that other media lack. With advances in computer hardware/software, computer graphics, and computer vision, 3D models have become widely adopted in industrial manufacturing, film production, virtual reality, education, and scientific research, emerging as a common form of intellectual asset [2]. This necessitates that IRs adapt to the diversification of knowledge output formats by supporting 3D model storage, organization, and management.

In terms of management and support capabilities, IR support for 3D models can be divided into two approaches. The first adopts methods similar to text content management, supporting 3D model description, upload, and preservation while

providing metadata-based retrieval and discovery. The second approach resembles image/audio/video management, offering online browsing capabilities and employing content analysis and indexing technologies to provide content-based retrieval and visual navigation for information discovery. The second approach is more user-friendly and enables more intuitive information retrieval. Previously constrained by technological limitations, 3D model organization could only adopt the first approach. However, recent advances in computer graphics and pattern recognition have made the second approach feasible.

This paper surveys the current state of 3D model retrieval and display technologies, investigates these technologies in depth, and integrates them with the functional requirements of the Chinese Academy of Sciences Institutional Repository (CSpace). Using open-source visual similarity-based 3D model retrieval algorithms and the Web3D display tool Three.js, we have constructed a 3D model retrieval and display system.

Compared to images, audio, and video, 3D models exhibit richer content and unique characteristics: numerous file formats due to diverse authoring tools (the 3D Object Converter supports at least 720 formats [3], far more complex than image/audio/video formats); separate storage of geometry and texture maps; text-editable file formats; higher graphics processing requirements; and invariance to translation, scaling, and rotation. These features make 3D model retrieval and display methods varied and technically complex.

2.1 3D Model Retrieval

3D model retrieval technology emerged in the early 1980s. After nearly 40 years of development, the field has accumulated substantial theoretical research and practical achievements, with several operational retrieval systems released. Representative foreign systems include the Princeton 3D Model Search Engine from Princeton University's Shape Retrieval and Analysis Lab [4], the University of Konstanz' s universal 3D model retrieval system [5], and Carnegie Mellon University' s content-based 3D model retrieval system [6]. Domestic research started later but has produced notable results, such as the visual similarity-based online 3D model retrieval system from National Taiwan University' s Communication and Multimedia Laboratory [7] and Jilin University' s Colored 3D Model Database (CDB) supporting color and texture-based retrieval [8].

Retrieval methods primarily fall into two categories: text-based and content-based retrieval. Text-based retrieval requires pre-labeling of 3D models and often fails to meet users' precise search needs. With enhanced graphics computing capabilities, content-based retrieval has become a research focus in recent years. Content-based retrieval includes three approaches: file-based retrieval, 2D sketch retrieval, and hand-drawn 3D sketch retrieval. Most current 3D model retrieval systems employ file-based retrieval, where users select a 3D model to find similar models in the database.

2.2 3D Model Display

3D model display comprises model rendering and user interaction, with interaction being the key aspect—enabling functions like zooming and rotation to allow users to examine model details from multiple perspectives. Display methods are divided into web-based browsing (Web3D) and client-side browsing. Client-side browsing requires dedicated software installation. For Web3D, major technologies include cloud rendering, Java3D [9], Away3D [10], Unity Web Player 3D [11], Cult3D [12], and Three.js [13].

Cloud rendering performs 3D model rendering on the server side, requiring no client configuration and enabling display across platforms including mobile devices, tablets, and PCs. During display, the client loads only rendered images rather than model files, eliminating loading delays. Theoretically, any model size can be displayed depending on server configuration, with enhanced security.

Java3D primarily implements Web3D through Java Applets, requiring JRE installation and browser security configuration, which complicates user operation. Away3D, Unity Web Player 3D, and Cult3D all require plugin installation. Three.js is an open-source WebGL framework that functions as a standard JavaScript file. While requiring WebGL-enabled browsers (supported by Chrome, Firefox, 360 Secure Browser 6.0, IE11, etc.), Three.js represents the future direction of browser development.

Through investigation and experimental research, we found that although numerous 3D model retrieval systems and platforms have been released, most lack open APIs or detailed algorithm descriptions, preventing effective integration into IR systems. National Taiwan University' s visual similarity-based 3D model retrieval algorithm, with its publicly available source code and demonstrated high recall and precision rates, can be modified and customized for IR system integration. Therefore, we selected this algorithm for our research.

From a user experience perspective, plugin-free Web3D display requiring no user configuration is most desirable. Both Three.js and cloud rendering meet this requirement. However, cloud rendering demands significant hardware and software investment, while Three.js encapsulates many low-level Web3D functions. By referencing its API documentation and importing relevant JS files, developers can implement algorithms supporting multi-terminal display (PC, tablet, mobile), aligning with CSpace' s adaptive interface requirements. Consequently, we adopted Three.js for our 3D model display research and implementation.

3. Institutional Repository 3D Model Retrieval and Display System Functions

The system provides six core functions: (1) **Model Format Conversion**: Given the diversity of 3D model formats, we convert submitted models to a unified format to avoid compatibility issues in subsequent feature extraction, thumbnail generation, and display. (2) **Feature Extraction**: 3D models offer

rich features including shape, contour, and color. We select appropriate parameters for feature extraction to prepare data for similarity calculation and retrieval. (3) **Indexing**: We design and maintain a 3D model index using selected storage methods, automatically updating the index when models are added or deleted to ensure consistency. (4) **Retrieval**: The system extracts features from user-submitted models, calculates similarity against all models in the repository using the index, applies a similarity threshold, and returns a ranked list of similar models. (5) **Thumbnail Generation**: To facilitate user selection, the retrieval results page displays 2D thumbnails of candidate models sorted by similarity. Users can click thumbnails to access the detailed model display page. (6) **Model Display and Interaction**: In the Web-based 3D display area, users can rotate and zoom models using mouse interactions for detailed examination.

The system functional framework is illustrated in [Figure 1: see original paper], comprising two main components: the retrieval module (format conversion, feature extraction, indexing, retrieval) and the Web3D display module (thumbnails, model rendering, interaction).

4. Design and Implementation of Key Functions

4.1 3D Model Processing

The implementation requires processing tools with specific environmental dependencies. Embedding these directly into existing IR systems would compromise OS compatibility and complicate installation and upgrades. Therefore, we employed Java RMI (Remote Method Invocation) [14] to deploy 3D model processing on a Windows Server 2003 machine, enabling remote calls from the IR system via IP address and port, as shown in [Figure 2: see original paper].

The client first connects to the processing server, uploads the 3D model, and upon successful transfer, the server performs format conversion, feature extraction, and thumbnail generation based on client parameters, returning the resulting files and status. Core implementation code:

```
// Connect to 3D model processing server
if (!connect()) return "Can not connect the 3D server!";
// Upload 3D model
if (!objFeatureGenerate.upload(fileName, fileToByte(srcFile))) return "Upload 3D file failed
// Model conversion
if (isconvert2obj) {
    if (!objFeatureGenerate.convert2obj(fileName)) return "convert2obj 3D file failed!";
    if (objFeatureGenerate.isFindObj(fileNameWithoutExt+".obj")) {
        // Generate thumbnail
        if (isGetImage) {
            if (!objFeatureGenerate.generateImage(fileNameWithoutExt)) return "Generate 3D i
        }
        // Generate feature parameter file
```

```
        if (isGenerateFeature) {
            if (!objFeatureGenerate.generateFeature(fileNameWithoutExt)) return "Generate 3D
        }
    }
}
```

The client-side method class and server-side interface are shown in [Figure 3: see original paper] and [Figure 4: see original paper] respectively. The package names must remain consistent between client and server to avoid method invocation errors.

The processing module implements three functions: (1) **Format Conversion**: We adopt the widely-used obj format as the unified standard. The 3D Object Converter software supports conversion from common formats (3ds, dxf, cad, geo, stl, c4d, rwx) to obj, installed on the processing server and invoked via the `convert2obj` method. (2) **Thumbnail Generation**: Since 3D model files are text-based, we use Java3D to read files, perform off-screen rendering into a buffer, and extract image data streams saved as JPG files. Due to complexity in texture loading, thumbnails exclude texture information and are stored as child items of the 3D model in the IR system. (3) **Feature Extraction**: We employ National Taiwan University' s visual similarity-based algorithm [15], which uses LightField Descriptors representing multiple viewpoints. We modified the original algorithm' s rendering method from Glut Windows (prone to occlusion artifacts) to off-screen rendering and adapted parameter invocation for Java compatibility.

4.2 3D Model Indexing and Retrieval

We use Solr-4.10.2 [16] for index maintenance. The indexing scheme aligns with the retrieval algorithm, which stores model lists in text files with each line representing a feature storage path (e.g., `Example/m0`). The Solr index primarily stores these path references. The schema.xml configuration defines:

```
<fieldtype name="string" class="solr.StrField" sortMissingLast="true" omitNorms="true"/>
<field name="id" type="string" indexed="true" stored="true" multiValued="false" required="tr
<field name="_{version}" type="long" indexed="true" stored="true"/>
```

3D model files are stored as Bitstream objects in the IR system, with feature files named using the Bitstream ID. Upon submission, the system identifies 3D models by file extension and triggers automatic format conversion, feature extraction, thumbnail generation, and Solr indexing. The Bitstream creation/modification/deletion indexing mechanism is detailed in literature [17].

The retrieval process is illustrated in [Figure 5: see original paper]. When a model is submitted, the system checks its format—obj files proceed directly to feature extraction while others are first converted. Features are compared against all models in the repository, similarities calculated, and results sorted. Similarity scores are positive integers where zero indicates identical models and

larger values indicate lower similarity. Based on experimentation, we set the similarity threshold at 1000, filtering out results above this value. The retrieval results page displays thumbnails sorted by similarity, with pagination implemented via Ajax for performance when many similar models exist.

4.3 Model Display and Interaction

Three.js is an open-source WebGL framework encapsulating many low-level Web3D functions. While its basic obj loader example enables simple display, it lacks adaptive sizing and centering—small models may be invisible while large models exceed canvas boundaries. We addressed this using Three.js' s `Box3` function to center models and calculate their dimensions for proper camera positioning:

```
var box3 = new THREE.Box3();
box3.setFromObject(object);
box3.center(object.position);
object.position.multiplyScalar(-1);
var a = new THREE.Vector3();
a = box3.size();
var objectSize = Math.max(a.x, a.y);
var fov = camera.fov * (Math.PI / 180);
var distance = Math.abs(objectSize / Math.sin(fov / 2));
camera.position.z = distance;
```

Rotation and zoom interactions are implemented using `OrbitControls.js`. The display function is embedded in `display-item.jsp`, rendering models without texture files against a white background with cyan lighting. The `displayModel(container, objname)` function enables multiple model display, where `container` specifies the display area and `objname` provides the model file path.

4.4 Implementation Results

To validate our implementation, we downloaded 20+ obj format models from National Taiwan University' s system and converted subsets to 3ds, dxf, cad, geo, stl, c4d, and rwx formats using 3D Object Converter. Upon submission to our IR system, non-obj formats triggered automatic conversion, followed by feature extraction, thumbnail generation, and index building.

Users initiate retrieval by clicking a camera icon and uploading a model file. The system identifies 3D models by extension and performs retrieval. Results demonstrate that various format uploads successfully complete automatic processing, with results sorted by similarity and presented as thumbnails. Thresholds distinguish between exact matches and similar models. The retrieval results page is shown in [Figure 6: see original paper].

Clicking thumbnails navigates to the detailed model view page, where mouse

interactions enable rotation and zooming. The model display interface is shown in [Figure 7: see original paper].

Conclusion

Institutional repositories are vital infrastructure for managing institutional knowledge output. As information technology evolves, knowledge formats are diversifying. To expand IR management capabilities, this paper integrated open-source 3D model retrieval algorithms with CSpace requirements to build a retrieval system and implemented Web-based display and interaction using Three.js. Testing confirms that retrieval results and efficiency meet current basic needs, though recall and precision require improvement. Future work will monitor advances in 3D model retrieval technology, optimize existing systems, and research semantic retrieval methods to provide better 3D model support services.

References

[1] Zhang Xiaolin. Trends and Challenges for Institutional Repositories[J]. New Technology of Library and Information Service, 2014(2): 1-7.

[2] Lu Tong. Retrieval of 3D CAD Model: Survey[J]. Computer Science, 2012, 39(4): 14-22, 27.

[3] 3D Object Converter[EB/OL]. [2016-06-10]. http://3doc.i3dconverter.com/features.html.

[4] Princeton 3D Model Search Engine[EB/OL]. [2016-06-10]. http://shape.cs.princeton.edu/search.html.

[5] Zarpalas D, Kordelas G, Daras P. Recognizing 3D Objects in Cluttered Scenes Using Projection Images[C]//Proceedings of the 18th IEEE International Conference on Image Processing. 2011: 673-676.

[6] AMP 3D Model Retrieval[EB/OL]. [2016-06-10]. http://chenlab.ece.cornell.edu/projects/3DModelRetrieval

[7] 3D Model Retrieval System Based on LightField Descriptors[EB/OL]. [2016-06-10]. http://3d.csie.ntu.edu.tw/.

[8] Cheng Yanzhi, Lü Tianyang, Wang Sen, et al. 3D Model Retrieval Based on Surface Color Properties[C]//Proceedings of NDBC2009. 2009: 366-372.

[9] Java3D[EB/OL]. [2016-03-20]. https://java3d.java.net/binary-builds.html.

[10] Away3D[EB/OL]. [2016-04-10]. http://away3d.com/.

[11] Unity3D[EB/OL]. [2016-04-10]. http://unity3d.com/cn/webplayer/.

[12] Cult3D[EB/OL]. [2016-04-10]. http://www.web3d.com.cn/new/teach/cult3d/.

[13] Three.js[EB/OL]. [2016-04-10]. http://threejs.org/.

[14] RMI[EB/OL]. [2016-04-10]. https://docs.oracle.com/javase/tutorial/rmi/.

[15] Chen D, Tian X, Shen Y, et al. On Visual Similarity Based 3D Model Retrieval[C]//Proceedings of Computer Graphics Forum. Blackwell Publishing, Inc, 2003: 223-232.

[16] Solr[EB/OL]. [2016-04-10]. http://lucene.apache.org/solr/.

[17] Wu Zhiqiang, Zhu Zhongming, Liu Wei, et al. The Image Retrieval of LireSolr-Based for Institutional Repository[J]. Research on Library Science, 2016(14): 58-63, 39.

Author Contributions

Wu Zhiqiang: Drafted the manuscript, designed and implemented system functions.
Zhu Zhongming: Proposed research ideas and designed the study framework.
Liu Wei: Deployed and tested system functions.
Zhang Wangqiang: Implemented 3D model indexing interfaces and revised the manuscript.
Yao Xiaona: Conducted research on 3D model retrieval technologies.

Conflict of Interest Statement

All authors declare no conflict of interest.

Support Data

Support data is self-archived by the authors. Contact: wuzq@llas.ac.cn.
[1] Wu Zhiqiang. GenerateFeature.zip. Remote 3D Model Processing Program.

Received: August 25, 2016
Revised: September 26, 2016

Note: Figure translations are in progress. See original paper for figures.