A 3D collaborative editor using WebGL and WebRTC

Caroline Desprat*
University of Toulouse

Jean-Pierre Jessel†
University of Toulouse

Hervé Luga‡
University of Toulouse

Abstract

In 3D collaborative environments, users needs interactivity and real-time updates. With web-based applications, such requirement implies that conventional client-server -alone- is no longer enough. To overcome this unmet need, we propose a hybrid client server peer-to-peer (P2P) communication model based on pluginless web standards enabling users to design collaboratively 3D scenes. The client part includes a WebGL editor to visualize and edit 3D scenes while the server side provides data and ensure persistence. Using the WebRTC protocol, a P2P mesh is generated to transmit directly the updates through a scenes working group. The feasibility of our approach is demonstrated with a web-based prototype submitted to a qualitative evaluation highlighting the usage of WebRTC for direct 3D data transmission with low latency and high throughput, and WebGL for 3D rendering.

CR Categories: I.3.2 [Computer Graphics]: Distributed/network graphics— [I.3.4]: Computer Graphics—Graphics editors H.5.3 [Group and Organization Interfaces]: Web-based interaction—;

Keywords: collaboration application, WebGL, WebRTC, P2P

1 Introduction

This research is led by the need to work collaboratively and to share 3D scenes across the network. 3D scenes or models are very likely to be constructed and reviewed by more than one person, particularly in the context of 3D scientific visualization [Marion and Jomier 2012], CAD (Computer Aided Design) [Houston et al. 2013] or BIM (Building Information Modeling) [Chen and Hou 2014] solutions. The new usages and the increasing mobility of workers are pointing to web-based collaborative solutions [Mouton et al. 2011]. In small designing teams, most of the design process is conducted through direct communication channels. Even if we need data persistence of what is done in this type of processes, why to pass through a proxy when the team members are so close? As a mimic in computing, there is peer-to-peer communication. Since the network speed can be a limiting factor in collaborative design, one of the main criteria for our system is to spread and display only relevant information between the users without overloading the server. The contributions of this work are:

- consider the solutions for pluginless visualization of 3D scenes on the web,
- use efficiently the local client resources for visualization and communication,
- allow small and asynchronous message system,
- overcome the difficulties related to real-time collaboration across the network with shared access to 3D models.

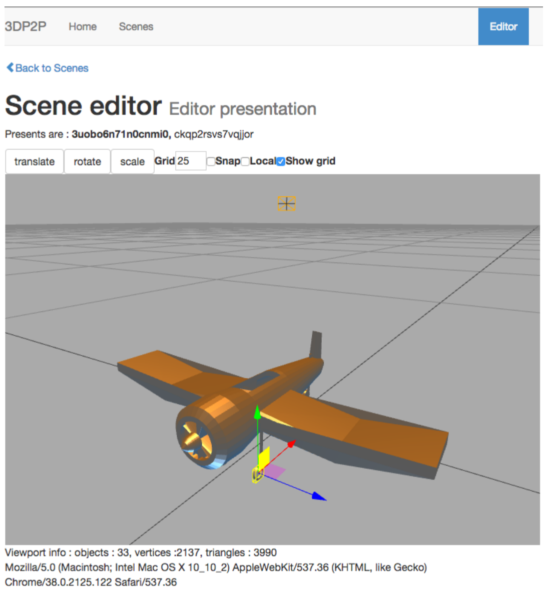
*e-mail:desprat@irit.fr
†e-mail:jessel@irit.fr
‡e-mail:luga@irit.fr

2 Hybrid model

To set up our collaborative design environment, we developed a full web based communication architecture with a 3D modeling platform. The users are working together on the same scene where they can add, remove and update 3D object models. In such virtual workspace, the contributions of each users are directly transmitted to others, observing the progression in real-time. The network model is mixing conventional client-server architecture, mostly used for persistence, and a full mesh peer to peer network for the real-time data transmission between clients. The system overview in Figure 2 illustrates the communication architecture topology between the peer clients and server (plus *signaling*).

2.1 Web-based 3D Editor

WebGL is a pluginless solution for powerful accelerated 3D rendering in a web browser client. It is used in our editor to process imported, server retrieved, updated 3D models for client visualization, interaction, and edition. The 3D editor displayed in Figure 1 integrates the 3D editing interface (tools, viewport, viewport info) and the list of the collaborators present on the scene.



Scene editor Editor presentation

Presents are : **3uobo6n71n0cnmi0,** ckqp2rsvs7vqjjor

Grid 25 ☐ Snap ☐ Local ✓ Show grid

Viewport info : objects : 33, vertices :2137, triangles : 3990
Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/38.0.2125.122 Safari/537.36

Figure 1: *Editor presentation*

2.2 RESTful server architecture

The client-server part is based on a REST (Representational State Transfer) architecture that benefits distributed hypermedia systems such as ours. The client queries are sent to the database (DB)

Web3D '15, June 18 - 21, 2015, HERAKLION, Greece
ACM 978-1-4503-3647-5/15/06.
http://dx.doi.org/10.1145/2775292.2778297

through the NodeJS server. MongoDB is a NoSQL DB providing dynamic schema (for large 3D data sets) and a rich query language API for data manipulation facilitating the enrichment of the (3D) objects on-the-fly. We mainly use it to maintain a world state persistence and provide robustness to the system.

2.3 WebRTC P2P communication

WebRTC (Web Real-Time Protocol) allows DataChannel protocol for P2P channels to exchange any raw data between web browsers in real-time [Grigorik 2013]. WebRTC uses *signaling* mechanism (via WebSocket server) to send control messages, get the network configuration and the media capabilities of the clients. The flow of users (arrivals and departures) and their relationships are managed by the server to build the full mesh topology network (every client is connected to others). The P2P message layer uses the star topology: a broadcaster send a message operation on WebRTC connection to its direct neighbors.

Workflow When arriving on the scene, each P2P client is assigned a ID sent to the server DB to get into collaborative network. Then, the client retrieves the entire scene document from the DB to build its own scene graph (but identical to others' peers). Each client is responsible for broadcasting its own messages (to peers and to DB) according to the granularity of the data transmission defined as follows (actions/message content): on import/meshes and materials; on delete/object ID; on transformation/object ID and transformation matrix. At message reception, the peer client updates its viewport rendering.

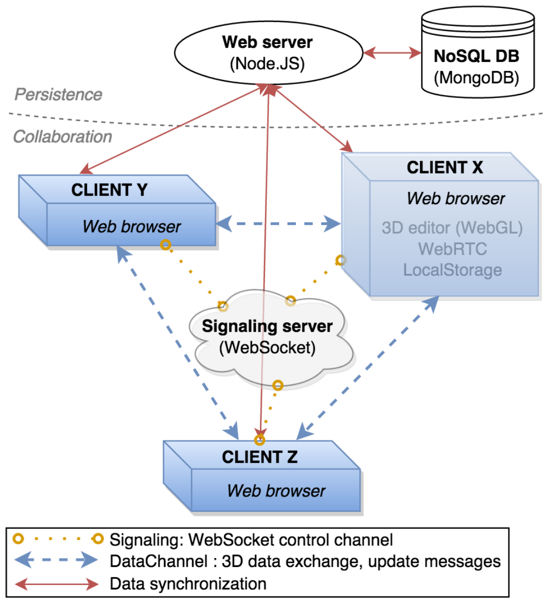


Figure 2: *System overview*

3 Evaluation

We proposed three experiments that last about 5/10 minutes each. The goal was to assemble collaboratively multiple parts of a scene (like a car with wheels, body, seats...) to match a given picture.

Users were satisfied of the collaborative and visual results of the experiments because they were able to reach the experiments' goal without frustration. They pointed out a lack of visual feedback on ongoing collaboration operations done by other users such as object prehension ownership. A few issues with the viewport and the object manipulation were met on reception of new imported models related to the size of the message to process. The user had to reload the scene to refresh the viewport. At its return, the user appreciated the application's robustness with the retrieval of the scene data and connections. Variating the number of users from one experiment to another has not altered the user experience (both rendering and networking). Latencies due to transformation operations were low enough for them to qualify the quality of the collaboration as real-time more than interactive.

4 Conclusion

The web-based 3D collaborative editor technology and design has the potential to offer 3D content creation to everyone with a web access. Our web application is based on a hybrid client-server (WebGL and NodeJS) and P2P (WebRTC) architecture. The client is in charge of 3D visualization, user interactions handling and updates management on the scene. For data persistence, clients are connected to a NoSQL DB to store modifications through the web server. It manages user presence on the scene and automatically creates the P2P full mesh topology network between them. The P2P connection (established by *signaling*) hosted by the client relies on the WebRTC DataChannel protocol that broadcasts messages directly between browsers according to the P2P star topology. Messages follows a granularity policy to limit the amount of sent data.

The qualitative evaluations of the experiments were conclusive overall, even if some points are to be improved (model import, richer interface, collaborative feedbacks). To handle larger scenes, we consider using adaptive rendering and enriching our P2P data streaming relying on seed peers with a better abstraction [Koskela et al. 2014]. Therefore, future works will be supplemented with a quantitative evaluation, to compare our hybrid architecture to others, using server logs, FPS in client and WebRTC tools (Chrome: `chrome://webrt-internal`; Firefox: `about:webrtc`) which provide statistics on the exchanged data.

References

CHEN, H.-M., AND HOU, C.-C. 2014. Asynchronous online collaboration in BIM generation using hybrid client-server and P2P network. *Automation in Construction 45*, 72–85.

GRIGORIK, I. 2013. *High Performance Browser Networking: What every web developer should know about networking and web performance.* " O'Reilly Media, Inc.".

HOUSTON, B., LARSEN, W., LARSEN, B., CARON, J., NIKFETRAT, N., LEUNG, C., SILVER, J., KAMAL-AL-DEEN, H., CALLAGHAN, P., CHEN, R., ET AL. 2013. Clara. io: full-featured 3d content creation for the web and cloud era. In *ACM SIGGRAPH 2013 Studio Talks*, ACM, 8.

KOSKELA, T., VATJUS-ANTTILA, J., AND DAHL, T. 2014. Communication architecture for a p2p-enhanced virtual environment client in a web browser. 1–5.

MARION, C., AND JOMIER, J. 2012. Real-time collaborative scientific webgl visualization with websocket. In *Proceedings of the 17th international conference on 3D web technology*, ACM, 47–50.

MOUTON, C., SONS, K., AND GRIMSTEAD, I. 2011. Collaborative visualization: current systems and future trends. In *Proceedings of the 16th International Conference on 3D Web Technology*, ACM, 101–110.