FRI-1.414-1-MIP-02

AN APPROACH FOR RENDERING 3D CONTENT FOR ONLINE SHOPPING

Stanislav Kostadinov, PhD
Developer at ForkPoint Ltd.,
Ruse, Bulgaria
Phone: +359898237753
E-mail: stanislav@forkpoint.com

Plamen Mihaylov, MSc
Co-founder of ForkPoint Ltd.,
Sofia, Bulgaria
E-mail: plamen@forkpoint.com

Kiril Kirov, MSc
CEO at ForkPoint Ltd.,
Sofia, Bulgaria
E-mail: kiril@forkpoint.com

Abstract: *This paper describes WebGl-Based viewer for rendering 3D content for the purposes of online shopping. The paper addresses techniques for transforming, visualizing and interacting with 3D objects. Special attention was paid to efficiency, cross-platform and cross-browser implementation of the viewer. The main purpose was to provide realistic 3D view of online shop items which will enrich the user experience and increase the sales of the e-shops. The viewer is designed to be extended with mass-spring simulations on different browsers and platforms. Viewer provides full set of light model characteristics, 3D object transformations and animations, camera animation, multi-format import and export, desktop, iOS and Android support, object space and image space collision detection techniques and 3D human body customization.*

Keywords: *E-Shop, 3D, GPU, Simulation, Mobile, Web-Based Viewer.*

INTRODUCTION

Web technologies developed very quickly in the last two decades and provide full sets of tools used in many different fields in human life. Nowadays, a lot of applications require visualization of three dimensional (3D) objects. Different graphics libraries and packages have been developed like OpenGL, WebGL, Direct3D, etc. However, some of the software approaches are not very realistic or they require specific hardware to be executed. One of these solutions is visualization of online shop items.

This paper describes software solution for visualization of 3D content on the Web, addresses the features and advantages from the standard solutions. The customer does not have to install any additional web browser plug-in for the visualization, apart from the WebGL graphics library that comes out of the box with each browser. The paper presents the advantages of such solution that can increases the realism of the presented item and help resellers present in a more user-friendly way their items. The rest of the paper is organized as follows. Second section describes the problem and the disadvantages of current solutions. Next section provides description of the proposed solution. It also presents some of the techniques used to implement the 3D viewer. Results from the implemented viewer and images produced by our software are placed in the fourth section. The conclusion section describes the advantages of the application and suggests additional features which can be implemented as part of the viewer.

RELATED WORK

Most of the solutions used for online shopping provide set of images of the shop item from shot different perspectives or with different variational attributes – colour, accessories, size, etc. (Zhao W., Yi O., 2008). Sometimes this information is not enough and user requires additional multimedia information. But even multimedia is not enough to fully present some shop items. This is the place where 3D rendering viewer helps enriching the user experience and increase the sales of the e-shops. Most of the online shops which trade with cloths provide tables with size characteristics for each item which sometimes makes user choices though. One user friendly way to solve this problem is by viewing 3D content that can be deformed, animated, painter to simulate the real-world items (Poon G., Yeung Y., 2014) (Wodohause A., Abba M., 2016).

Most of the e-shops provide standard page where user can see and choose different variations of the shop items (Fig. 1). They also provide a return policy where it is said that if the item is not matching to the needs of the user (if the item is cloth for example – whether it fits to the human's body) it can be returned. But this policy decreases the profit, so the e-shops starve for more realistic view of their items.

Fig. 1. Standard product page with image of the image and menu for selection of different variations of the item

DESCRIPTION OF PROPOSED SOLUTION

This is where 3D visualization comes in mind. The application presented in this paper fills in the gap between the needs of customers and the way online shops present their items. Nowadays, most of the software applications are designed with mobile first approach, so this 3D view is designed to work properly on both desktop computers and mobile devices. This requirement limits all possible approaches to implementation of HTML canvas and drawing content with WebGL graphics library. Furthermore, the version of the WebGL shaders (programs running on the GPU) is implemented with the initial version of the graphics language. Most of the client-side viewer is implemented in Typescript - superset of JavaScript that brings users optional static types and solid tooling and that can help avoiding painful bugs. Additional GLSL (OpenGL Shading Language) programs are implemented and ran on the GPU and speeds up the 3D visualization in a way the viewer can run real-time simulations.

A realistic view, that can help selling more shop items, requires implementation of some basic concepts of rendering 3D on the Web. Currently, this viewer provides some useful features like:

1. light model which can be configured, edited real-time and extended with some advanced characteristics:
   - physically based rendering pipeline – ensures realistic view as closer as possible to the real-world light model
   - local reflections – used mostly for shiny items
   - refraction – transforms light passed through semi-transparent items
   - per pixel lighting – ensures smooth surfaces of the shop items (Wang J., Sun J., 2004)
   - background (custom solid color or background image) - bring more realistic environment to the user
   - omni light and global ambient light – ensures simulation of real-world lighting
2. configuration of the color of three-dimensional objects:
   - common shader model (Lambertian model) – one of the most implemented shader model for rendering 3D objects (Wu Z., Sun Y., 2013)
   - integrated with external material editor – provides techniques for updating color model of the objects
   - common characteristics (ambient, diffuse, specular and emission color/texture) – all part of the shader model
   - advanced shader characteristics (shininess, global transparency, transparency mapping, reflection coefficient, spherical reflection mapping and parallax mapping) – these settings finish the global light model and create approach that guarantees that viewer can render any king of real-world objects
   - MTL file format support – imports all described characteristics from standard file formats
   - dynamic update of each material characteristics – panel that is used from the user to manipulate color characteristics
3. animation and transformation:
   - skeletal animation and deformation – this functionality deforms or animates any rigid object imported within the viewer (Tsai J., Chang S., 2015)
   - human skeleton animation and transformation (skin and bones) – functionality that is used for rendering human avatars with different cloths from the shopping items
   - camera animation – it is useful for animating the perspective and focusing on a specific part of the item that should be sold
4. navigation:
   - type of navigation "moving observer, fixed target navigation" - viewport rotation and zooming with mouse or on touchscreen devices
   - full-screen mode – provides maximum size canvas where all the details of the shop items can be seen easier without interruption of any other HTML information from the page
   - predefined viewpoints – allows navigation to specific views where user can focus on important detail of the shop item
5. file formats:
   - viewer imports objects from OBJ and STL files and export to OBJ file
   - it imports materials from MTL file format
   - JPEG, PNG, TGA and TIFF image support
   - it also imports and exports animations and transformations to JSON file format
6. other useful features:
   - desktop, iOS, Android support – allows users with any device to view the 3D content
   - asynchronous resource loading – enables user manipulation of the page while the contents are loading

- flexible approach for conversion of unknown image formats by plugging external APIs – JavaScript event API is used to convert unknown formats to others that are well known
- manipulation of materials by any other JavaScript APIs with event notification
- animation of human body model by any other JavaScript APIs with event notification
- animation and transformation of OBJ meshes by any other JavaScript APIs with event notification
- converter from rigid OBJ meshes to deformable cloth model
- advanced physics simulator for mass-spring simulations – used to simulate basic cloth deformation
- object space (Vassilev T., Dochev V., 2010) and image space (Dochev V., Vassilev T., 2004) collision detection – one of the newest GPU algorithms for collision detection

This WebGL viewer implements a set of techniques used for some of the major functionalities. One of the most useful functionalities is the animation and transformation of objects. The main structure is the skeleton used to transform the object. It is a tree of connected bones and each of the bones create a new local coordinate system. Transformation of a point connected to a specific bone depends on transformation components of this bone and all parent transformations (Bleisch S., Burkhard J., 2009). Transformations can be presented with equation with transformation matrixes:

$$P' = T * C * R * SR * S * -SR * -C * P \quad (1)$$

P' is the result point and P is the initial one. T is the translation matrix, C is the distance from the center of the coordinate system to the start of the coordinate system, R is the rotation matrix, SR is the scale orientation used to define the orientation of scaling matrix (which is S). The equation (1) presents the deformation of a point from the main coordinate system and all child systems use the parent transformation matrixes to generate their own transformation matrix. Each object point can be connected to one or more bones with bone weights. The sum of all weight is equal to 1 and the resulting point is also computed by calculation of the resulting point from each coordinate system and multiplication with the corresponding weight (2)

$$P' = \sum_{k=1}^{n} \mathrm{W_k} * \mathrm{P_k} . \quad (2)$$

Another useful technique is per pixel lighting which can be implemented by passing of all light information from vertex shaders (where they normally are calculated) to the fragment shader where colors of each point are calculated. Per-pixel lighting is commonly used with techniques like normal mapping, bump mapping, specularity, and shadow volumes. Each of these techniques provides some additional data about the surface being lit or the scene and light sources that contributes to the final look and feel of the surface.

Lambertian model is another approach implemented within the shaders. It also provides additional characteristics that increases the realism of the scene. The viewer contains multiple GLSL shaders and one of them executes mass-spring simulations over a set of connected points that forms cloth. It does not contain collision response functionality but the approach allows plugging such code. The core of this functionality is the collision detection shader program that finds collisions between any kind of objects (deformable and nondeformable). Two approaches for collision detection are implemented: object space and image space. Object space detection is implemented by separation of the space by an oct-Tree. The concept is to present coordinate system as cube and each cube component is divided into two equal parts in a way that new eight cubes are generated. The separation stops when cubes are small enough (depends on the initial settings of the algorithm) and each cube keeps information if there is a part of the scene objects within them. This information is later used in the simulation to find if any moving part is situated in a cube filled with another object part. The other approach for collision detection (image space) uses ray-tracing algorithm for intersection of moving object parts with front and back surfaces of colliding objects. These surfaces are normal maps generated by custom GLSL shader and their

alpha component keeps the z-index of the scene objects. During simulation, each point and its velocity is used as ray that intersects with one of the surfaces and should reflect in a way that should look as realistic as possible. The reflection after collisions can be added by extending one of the algorithms with collision response algorithm.

RESULTS

Example of online shop item rendered through the 3D view can be seen on Fig. 2. It allows updating of the size, color and of the mannequin (human avatar) and cloth's during rendering.

Fig. 2. Sample page with 3D viewer that renders men's cloths

Example of Lambertian model implemented within the viewer can be seen on the windows and doors of the car on Fig. 3.

Fig. 3. Lambertian model with additional characteristics (reflection, transparency, etc.)

CONCLUSIONS

This paper presented an approach for rendering 3D object for the need of online shopping sites. The following conclusions can be drawn:

- the current approach is more realistic than the currently used solutions
- it is fast enough and can be run on any device and browser
- it can be used to render realistic complex scenes (human avatar with suits, etc.) which helps users to choose the right item and this is how it can increase profit for the resellers

This approach can be extended and it can run physics simulations (mass-spring simulations) with additional environment settings like wind, gravity, etc. This ensures realistic simulation which is the perfect sign of flexible tool that satisfies all the needs of the clients. Additional features can be implemented:

- real-time shadows of the items
- panels user for control over animations and deformations of the items
- three dimensional selection of precise part of an item
- additional camera control to simulate human navigation through buildings or landscapes. Fig. 4 is an example of such building view

Fig. 4. Interior of a house

REFERENCES

Bleisch, S., Burkhard, J. & Nebiker, S. (2009). *Efficient Integration of data graphics into virtual 3D Environments*, International Cartographic Conference, 15–21 November 2009, Santiago de Chile.

Dochev, V., Vassilev, T. & Spanlang, B. (2004), *Image-space based collision detection in cloth simulation on walking humans*, International Conference on Computer Systems and Technologies (CompSysTech), 17–18 June 2004, Ruse.

Poon, G., Yeung, Y. & Pang, W. (2014). *Enabling 3D online shopping with affordable depth scanned models*, International Conference on Smart Computing (SMARTCOMP), 3–5 November 2014, Hong Kong.

Tsai, J., Chang, S., Yen, S. & Li, K. (2015). *3D skeleton construction by multi-view 2D images and 3D model segmentation*, International Journal of Computational Science and Engineering, 10(4), 368–374.

Vassilev, T. & Dochev, V. (2010). *Object Space Based Collision Detection for Cloth Simulation on the GPU*, Ruse University Conference, 30–31 October 2010, Ruse.

Wang, J. & Sun, J. (2004). *Real-time bump mapped texture shading based-on hardware acceleration*, International Conference on Virtual Reality Continuum and its Applications in Industry, 16–18 June 2004, Singapore.

Wodohause, A. & Abba, M. (2016*). 3D visualization for online retail: factors in human behaviour*, International Journal of Market Research, 58(3), 451–472.

Wu, Z., Sun, Y. & Jian M. (2013). *A 3D reconstruction algorithm based on PMS and Lambertian model under clear water*, Journal of Information and Computational Science, 10(17), 5805–5810.

Zhao, W. & Yi, O. (2008). *Ecommerce image retrieval system based on TSS*, Journal of Information and Computational Science, 5(4), 1689–1696.

