HTML5-based system for interoperable 3D digital home applications

Mikel Zorrilla, Angel Martin, Jairo R. Sanchez, Iñigo Tamayo, Igor G. Olaizola
Digital TV & Multimedia Services
Vicomtech-IK4 - GraphicsMedia.Net
San Sebastian, Spain
Email: {mzorrilla, amartin, jrsanchez, itamayo, iolaizola}@vicomtech.org

***Abstract*—Digital home application market shifts just about every month. This means risk for developers struggling to adapt their applications to several platforms and marketplaces while changing how people experience and use their TVs, smartphones and tablets. New ubiquitous and context-aware experiences through interactive 3D applications on these devices engage users to interact with complex 3D scenes in virtual applications. Interactive 3D applications are boosted by emerging standards such as HTML5 and WebGL removing limitations, and transforming the Web into a horizontal application framework to tackle interoperability over the heterogeneous digital home platforms. Developers can apply their knowledge of web-based solutions to design digital home applications, removing learning curve barriers related to platform-specific APIs. However, constraints to render complex 3D environments are still present especially in home media devices. This paper provides a state-of-the-art survey of current capabilities and limitations of the digital home devices and describes a latency-driven system design based on hybrid remote and local rendering architecture, enhancing the interactive experience of 3D graphics on these thin devices. It supports interactive navigation of sophisticated 3D scenes while provides an interoperable solution that can be deployed over the wide digital home device landscape.**

***Index Terms*—Home device interoperability; Digital home applications; Computer graphics; 3D virtual environments; Interactivity; Hybrid rendering system;**

I. INTRODUCTION

Users are becoming more accustomed to improved experiences that provide interactive 3D applications exploiting the technology in immersive environments. Thanks to the advent of low energy-consumption Graphic Processing Units, interactive 3D applications are currently running in most digital home devices. Connected TVs, smartphones and tablets are being fitted with graphic capabilities providing users an enhanced experience on top of interactive and 3D applications, and pushing the market to new advanced 3D applications with complex interactive virtual environments.

The landscape of digital home devices has changed last years completely with the introduction of smartphones and tablets in the home network bringing secondary displays to foster customized media, together with the evolution of the TV to Smart Connected TV. Moreover, these kind of devices are running over application-based Operating Systems. Most popular are Android and iOS[1] for smartphones and tablets and different proprietary platforms (Samsung, Philips, etc.) for Smart TVs. However big companies such as Google or Apple offer a Connected TV solution which could nearly provide a full digital home approach through the different devices and their Operative System. Each solution facilitates a framework and an SDK (Software Developer Kit) to exploit native assets providing the hardware features of the devices: connectivity, motion and voice control, camera, GPS, graphic capabilities, etc. However, the deployment of the applications from one OS to the others implies major changes and specific adaptation. This platform heterogeneity at the OS level generates an important interoperability problem.

The rapidly increasing use of the Web as a software platform with truly interactive applications is boosted by emerging standards such as HTML5[1] and WebGL[2] that are removing limitations, and transforming the Web into a real application platform middleware to tackle the interoperability problem. Following this trend, the new HbbTV[3] standard for broadcasting environment interactivity is also based on a specific HTML browser.

HTML5 provides devices the capability to run rich web applications accessing the entire device features on a web browser. It comes together with CSS[4] and JavaScript which provides an appropriate framework for the content interactivity and universal access to different APIs. WebGL is the API oriented to 3D graphics in the HTML5 canvas element. It is easier to craft innovative user experiences using powerful HTML5 layout and WebGL rendering engines than current native IDEs.

Digital home browsers are rapidly adopting HTML5 features on a tough race just after the desktop browsers. The standard has won a prominent place as a horizontal approach to reach interactive multimedia applications on home devices. HTML5 applications can be packed for the different execution environments providing an interoperable application with minor changes through different OSs. That is why HTML5 is being strongly promoted by the standardization bodies and a sector of the market to achieve a HTML5 marketplace instead of the different proprietary ones, such as Android Market, iPhone App Store, Samsung Apps Market, Net TV Apps, etc.

[1] Html5 standard specification (May 2011) http://www.w3.org/TR/html5/
[2] Webgl website (Mar. 2011) http://www.khronos.org/webgl/
[3] HbbTV 1.5 specification (April 2012) http://www.hbbtv.org
[4] Cascading style sheets (css) standard specification (May 2011) http://www.w3.org/TR/CSS/

Digital home applications are changing how people experience and use these devices. The incoming pioneer interactive 3D applications for mobiles are inciting users to discover new ubiquitous and context-aware experiences through smartphones and tablets and show the feasibility to access this rich media apps through the Smart TV. User requirements are involved in the mentioned tough race demanding power efficient techniques together with advanced interactive virtual applications with complex 3D scenes on digital home devices as they do on PCs.

This paper provides a complete state-of-the-art of the current browser capabilities of the digital home devices using HTML5. We present performance results concluded by experiments carried out in representative set-top boxes, smartphones and tablets. The current limitations to run advanced interactive 3D applications are also explained in the article. We also present a solution to overcome the detected limitations to be able to run advanced interactive 3D applications using HTML5, making thin devices usable platforms for a larger range of applications. A system architecture called *3DMaaS* is detailed to provide complementary rendering capabilities to these devices, adding to their own capabilities the chance to push to the cloud complex 3D rendering tasks. A technical validation of *3DMaaS* is done emphasizing on the overcoming of the limitations detailed on the state-of-the-art.

II. Digital Home Device Software Platforms

The TV is still the main device for watching media content in the digital home. Nevertheless in the same way that mobile phones have gone from thin to smartphones and tablets, providing access to all kind of services and contents, home television is evolving from a passive device for multimedia content consumption to the so called SmartTV. Worldwide shipments of Internet-connected televisions have reached 25% of total units in 2011 and it is expected that it will the 70% by 2016[5].

However, the Connected TV platforms are very heterogeneous and based on proprietary approaches, where the interoperability is a problem. TV manufacturers have developed their own frameworks, providing a SDK to develop specific applications and with a own marketplace. Samsung Smart TV provides a SDK to develop Flash-based or JavaScript engine-based applications. These applications are located by Samsung in their marketplace called Samsung Apps. Philips Net TV provides a CE-HTML browser with a index page to access to the Net TV apps. Connected set-top boxes are also very heterogeneous with different web browser such as Opera Mobile or specific OS such as Boxee[6]. Moreover, Google and Apple have their TV solutions, Google TV[7] and Apple TV[8] respectively, but they are not positioned yet as a market leader as they do on mobile systems.

[5]May 2012. IMS Research.
[6]http://www.boxee.tv/
[7]http://www.google.com/tv/
[8]http://www.apple.com/uk/appletv/

Smartphone and tablet market penetration is going faster than Connected TVs. In Q4 2011, global smartphone shipments jumped to 157 million units[9]. It was a rise of 56% from last year beating the annual growth rate. Growth continues but it is slowing down as most of the developed markets come close to 80-90% penetration. Meanwhile, global tablet shipments reached 17 million units in Q3 2011[10]. As tablets and smartphones get faster, allowing a quicker transfer of data, integrating new connectivity and interactivity paradigms along with fancy graphics, users have developed a habit for downloading applications. This pushes mobile application market to a rapid evolution shifting the business landscape and to a competitive environment. The research firm Gartner recently forecast that mobile application stores will deliver 17.7 billion downloads internationally in 2011[11]. A key difference of each platform is the application market, App Store, Android Market and Windows Phone Marketplace.

The Android Market is open [2], whereas others are gated. This means, Android foster developers to self-publish created applications into the Android Market, whereas Apple or Microsoft decide what gets published keeping the application approval right before they become available in the Marketplace.

The different marketplaces availability responds to the change on the mobile phone landscape, playing their correspondent OS, such as iOS, Android, Windows Mobile or Research In Motion (RIM) BlackBerry OS, a prominent role in the applications development [1]. The market penetration of Android and iOS is increasing strongly and both are becoming the two major OSs to take into account. While on January 2011 the market share of Android and iOS was 61% in Europe, on April 2012 it has raised up to 73% thanks to the big increase of the Android OS. This trend is more representative in North America, where on January 2011 the Android and iOS mobiles represented the 61% of the market, and one year later they are almost the 86% (Figure 1).

The Android operating system is built from a modified Linux kernel. Previous specific versions for tablets and smartphones, entirely designed for devices with large screens and thinner devices respectively, converge in the version 4 that brings together phones and tablets easing the multi-device development and interoperability. The software stack contains Java applications running on a virtual machine, and system components are written in Java, C, C++, and XML. In order to develop Android applications the SDK can be integrated in different environments such as Eclipse.

Apple developed the iOS for its products catalogue. The operating system is derived from Mac OS X and is built on top of the Darwin foundation and XNU kernel. XNU combines the Mach 3 microkernel, elements of Berkeley Software Distribution (BSD) Unix, and an object-oriented device driver API

[9]Kang, T.: Global smartphone vendor market share: Q4 2011. Strategy Analytics (Feb. 2012)
[10]Mawston, N.: Global tablet vendor market share: Q3 2011. Strategy Analytics (Jan. 2012)
[11]Getting into the mobile app market. IEEE Computer Society Website (2011)

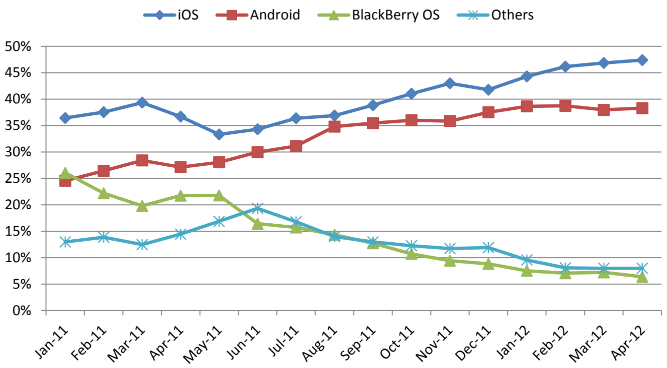

Fig. 1. Top Mobile OSs in North America from Jan 2011 to Apr 2012 (StatCounter Global Stats)

(I/O kit). iOS frameworks are written in Objective-C. In order to tackle application development for iOS, Apple provides a Xcode development environment and a iOS Simulator to test applications.

To sum up, the current digital home platform ecosystem is heterogeneous, with several operating systems, programming languages, and interfaces, resulting in more complex software cross-platform development and testing processes. Digital home devices increasingly depend on reliable software to offer a fresh user experience. Hence the current trend in developing interoperable applications lays on using Web technology instead of platform-specific APIs.

III. The Web as a Software Platform

According to the wide landscape of digital home application frameworks described in the previous section, developers need to carefully determine how and where to invest their time and effort before tackling an application development project. Writing native applications requires developers expertise and background in specialized IDEs.

However, Smart TV, smartphone and tablet trend is to be always connected to the Internet. Application developers should not ignore advantages of moving from desktop computing to web-based applications [3], [4], [5]. On the one hand, applications provided on the Web as services do not require installation or manual upgrades, easing the software life cycle management while inherit web security and privacy policies. On the other hand, in terms of monetizing an application, another relevant advantage lays on the deployment and sharing of Web applications that can be instantly worldwide, with no middlemen or distributors. This way the application monetizing strategy do not have to obey marketplace policies enabling a free design of the business model. Last but not least, the potential of the web-based applications can support user collaboration over the Internet, deploying virtual spaces where users interact and share application experience and data, fostering new paradigms of interactivity and social networking.

Developers can also benefit from web-based solutions saving time and effort. They can apply their knowledge of designing web applications to smartphone, tablet or TV application design, removing learning curve barriers. Web-centric approach for digital home applications enables not only rapid prototyping, but also unified integration with Web services. It requires access to the hardware resources of the digital home devices through JavaScript that always lags behind the new capabilities that manufacturers introduce. In order to mitigate this limitation new W3C's HTML standard provides device orientation, speech recognition and geolocation management bridging from native features to web-centric development.

From the viewpoint of the developers, the key for transition towards web-based software is the ongoing evolution of web development technologies, specifically HTML, CSS and JavaScript. This way, development turns more efficient to face interoperable and innovative mobile user experiences exploiting powerful HTML layout and rendering engines than native IDEs.

Emerging standards such as HTML5 and WebGL will play a crucial role removing the remaining limitations and transforming the Web into a horizontal software platform. They will significantly shift the perception of the web browser and web applications capabilities to a fully featured web-centric operating system and to a fully interoperable application respectively.

W3C HTML5 standard specification[12] defines the core language of the World Wide Web. New features and elements are introduced paying an special attention to improve interoperability.

HTML5 provides many capabilities enabling developers to combine video, audio, 3D, and 2D into one seamless application. HTML5 embraces multimedia by means of built-in audio and video support through *<audio>* and *<video>* tags that allow media files to be played without third party browser plug-in components.

HTML5 also brings relevant features fostering new paradigms of interactivity and user experience. The Canvas API provides salient 2D drawing capabilities for interactive graphics. Moreover, HTML5 specification provides numerous additions and enhancements such us realtime message based, speech recognitions, device orientation awareness or drag&drop action to be applied to the HTML objects.

CSS3[13] brings lots of possibilities that boost creativity such as transitions, opacity definition and native columns. It also provides much more flexibility enabling 3D effects such as zoom, pan, rotation, transformations and animations.

But the most relevant features that turn HTML5 into a interoperable software application platform are:

- Offline operation. The HTML5 contains several features that address the challenge of building web applications that allow to operate even when an active network connection is not available.
- Local storage. HTML5 brings a persistent cache based on local SQL database, allowing data to be stored locally in the device. It also provides a filesystem API in order to manage read and write actions.

[12]Html5 standard specification (May 2011) http://www.w3.org/TR/html5/

[13]Cascading style sheets (css) standard specification (May 2011) http://www.w3.org/TR/CSS/

Moreover, HTML5 development can be easily transformed in an application package ready to be provided in the Android marketplace or in other App stores. PhoneGap[14] is an open source framework for creating mobile web applications in HTML5, JavaScript and CSS3 while still taking advantage of the core features of native applications in some platforms such as iOS and Android devices.

Applications often engage users through 3D visual interfaces. They are more effective, attractive, and are considered as a key factor to add value to applications improving the overall user experience. For the Web, WebGL takes the role of enabling technology as a solid foundation for 3D graphics applications [6].

WebGL[15] is a cross-platform web standard for hardware accelerated 3D graphics API developed by the Khronos Group that includes among others Mozilla, Apple, Google and Opera. WebGL brings to the Web the support to display and manipulate 3D graphics natively in the web browser without any plug-in components. WebGL performs 3D graphics on top of the HTML5 canvas element and is accessed using Document Object Model (DOM) interface. WebGL allows communication between JavaScript applications and the OpenGL software libraries, which accesses the graphics processor of the device. This makes possible to exploit hardware capabilities to render 3D content. WebGL is based on OpenGL ES 2.0, and it uses the OpenGL shading language GLSL.

IV. HTML5 Interactive 3D Applications

A. Advanced 3D Application requirements

Recently the use of 3D graphics in many industrial fields and applications such as games, advertisement products interaction, serious gaming/simulation for more effective training, financial and medical data analysis, and CAD design are increasing more and more. Often applied data 3D applications interfaces exploit 3D graphics to support professional user productivity or represent data that could not be done otherwise such as Google Earth. But 3D graphics is also used for making more visually attractive interfaces.

Due to the mobility of users and professionals involved in these applications, it is mandatory to provide access through mobile devices tracking the variety of contexts of the user. Facing the emerging trend in consumer technology for delivering 3D content to the mainstream user via digital home devices[7], new solutions must promote the communication between the TV and mobile devices of the digital home. This provides users access to 3D graphics applications through Connected TVs, smartphones and tablets having a great experience interacting with 3D virtual environments. Demand for 3D visualization is increasing in these devices as users expect more realistic immersive experiences. So 3D graphics combines immersion and interactivity fostering creativity for new envisaged applications and information navigation interfaces.

[14] Phonegap website (Jan. 2012) http://www.phonegap.com

[15] Webgl website (Mar. 2011) http://www.khronos.org/webgl/

Mobile games are one of the fastest growing segments of the application industry. Bringing together the social gaming paradigm and the internet connection capability of most of the digital home devices, users will embrace the online interaction trend from PC.

User interfaces based on 3D graphics let users interact with virtual objects, environments, or information but the experience can be improved with the inclusion of real media sources around the user. According to [8] definition, virtual worlds technologies completely immerse a user inside a synthetic environment. In contrast, augmented reality allows the user to enjoy the real environment, with virtual objects superimposed upon or composited with the real world providing extra information or interactivity about what is around. This requires the fusion of very heterogeneous media sources in a concept called 3D Media [9], [10]. 3D Media is composed of different audio and video sources, static images and 3D objects enabling enhanced experiences.

3D rendering pushes the visual boundaries and interactive experience of rich environments, but 3D interfaces, virtual worlds and augmented reality applications require high 3D graphical features. The more complex the 3D scenes are, the higher the hardware requirements are. Although connected TVs, set-top boxes, smartphones and tablets are rapidly improving their graphic capabilities thanks to the integration of low energy-consumption Graphic Processing Units, the capabilities are below the user expectation. Users are demanding experiences they are used to in powerful devices such us PCs, mixed with the new characteristics that the digital home devices provide, such us using the camera of the smartphone for an ubiquitous augmented reality experience.

B. Limitations of the Browsers in Digital Home Devices

WebGL API coupled with JavaScript engines are boosting increasing capabilities of the web browsers making possible to develop complex computational environments including 3D graphics. Therefore platform-independent applications are directly performed through the web browser on different devices without the need to install additional software or plug-ins bringing the accessibility and interoperability of the web. However, constraints to render complex 3D environments are still present in digital home devices. It is necessary to define the hurdles, in terms of performance, that a developer will face when creating a web browser-based software for 3D interactive applications on top of HTML5 and WebGL. Here, we introduce not only the limitations around complex applications that require 3D graphics technology according to a set of evaluations, but also the clues about the bottlenecks origin that would enable the work to be done to remove the detected barriers.

We have chosen different devices that provides a wide representative landscape of the current digital home platforms, in order to detect the browser capabilities and limitations. On one hand, we have selected two high performance set-top boxes with browsers that support HTML5 and WebGL:

- **Innout Media Center 4Gs HD Set-top Box**: Opera Mobile 12.0 browser supporting HTML5 / WebGL profile and HbbTV profile.
- **Gigabyte GN-SB100 series**: Android 2.2 OS, Opera Mobile 12.0 browser.

These set-top boxes support WebGL but do not have specific hardware to run it. However, Opera Mobile has announced[16] that its TV browser with WebGL runs on the recently launched Intel Atom Media Processor CE5300. Mitsubishi Electric is also working on a set-top box with a high performance TV browser called Espial[17] with WebGL applications support.

On the other hand, according to the mobile devices in the digital home, two of the selected devices are Android and the other two are iOS. In order to track the market trend, where the tablets have an increasing presence, the evaluations consider two smartphones and two tablets:

- **Samsung Galaxy S**: GT-I900 smartphone with Android 2.2.1 firmware.
- **Samsung Galaxy TAB**: GP-P1000 tablet with Android 2.2 firmware.
- **iPhone 4**: iOS5 smartphone.
- **iPad**: iOS5 tablet.

Android and iOS Safari default browsers do not support WebGL yet. Neither Opera Mini nor Google Chrome Beta version for Android 4 do, but all of them have included it in their roadmaps. Here, for the Android devices, the Mozilla Firefox 4.0 browser have been employed for the tests. Firefox has WebGL support and it can be installed from the Android Market. Other browsers such as Opera Mobile 12.0 also support WebGL for Android devices. But for the iOS devices, a specific application which runs a webkit based browser called GoWebGL[18] provides WebGL capabilities.

In order to measure the frame rate achieved for each device, a simple 3D scene is composed using rotating cubes. In each test the total number of 3D objects is increased as well as their polygonal complexity, ranging from 1 to 80 objects and from 12 to 200k polygons per object. The geometry of one single cube is loaded into a vertex buffer which is drawn multiple times using a different transform matrix for each cube. The performance is measured as the average time per frame sampled over 50 frames for each object and polygon configuration. Figure 2 shows the results of these tests as the maximum number of polygons that can be rendered in interactive time (15 fps) in function of the object quantity in the iPad.

As seen in the plot, the maximum number of polygons drops exponentially with the number of 3D objects. Given that the geometry is loaded as a vertex buffer and provided that the total number of polygons is maintained constant, these results can be explained by two reasons. On the one hand, floating point operations are very CPU demanding in JavaScript. Since an additional cube means an additional matrix rotation, the overall performance is significantly affected. On the other hand, the new transform matrix must be transferred to the GPU overloading the CPU-GPU communication bus. Although the amount of data is quite small, the bus latencies of these small devices can have a negative impact which means a notable bottleneck.

[16]March 2012. IP&TV World Forum in London.

[17]March 2012. http://www.espial.com/company/press_item/id745

[18]https://github.com/gauthiier/GoWebGL

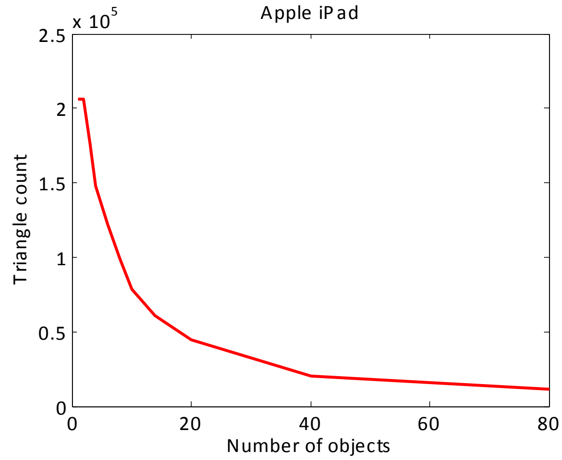

Fig. 2. Maximum number of polygons that can be rendered in interactive time (15 fps) in function of the object quantity in the iPad

From these results it can be derived that current WebGL subsystems can support a good performance for simple scenes composed by small amounts of objects, regardless of its polygonal complexity. This limitation brings an important drawback hindering scene-graph based rendering engines, since each object in the graph must be transformed recursively with respect to its parent.

Figure 3 compares the performance of the different devices in terms of the maximum number of polygons that can be rendered against the number of objects while keeping 15 fps target frame rate. The trend is quite similar through the four devices. The iPad and iPhone have the same 3D processing behavior while increasing the number of objects, being the performance of the iPad slightly better than others. The Android devices achieve almost the same throughput while increasing the number of objects and the responsiveness is very close to the iPad. However, the capabilities drop from 40 objects, specially in the Samsung Galaxy S.

These results become evident the need to improve the performance of 3D applications over the digital home browsers. However, these measures where unachievable some months ago, and the rapid adoption of HTML5 features on the mobile browsers let us think that these results are going to be improved very fast removing barriers in terms of WebGL compliance. Android and iOS browser will be able to run WebGL soon in the same way that other mobile browsers will do it (Google Chrome, Opera Mini, etc.) and will be accessible from these platforms. The WebGL performance itself needs to be improved by a better integration of the JavaScript capabilities of the browser and the architecture of the device. However, remaining throughput limits closely

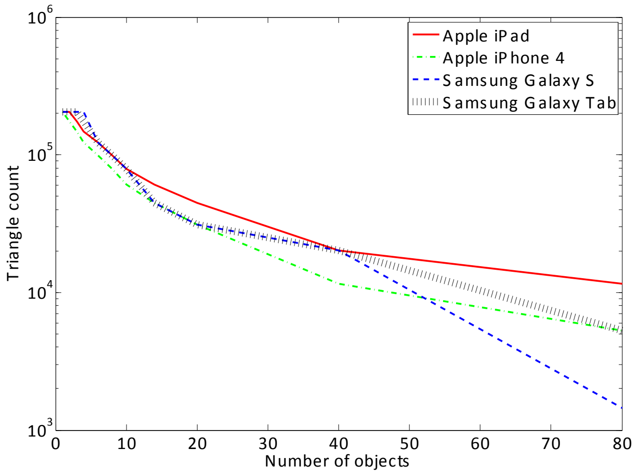

Fig. 3. A comparative of the maximum number of polygons that can be rendered in interactive time (15 fps) in function of the object quantity in the different devices

related to GPU potential would not disappear quickly due to life battery technological constraints.

In terms of HTML5 and WebGL support for Connected TVs and set-top boxes, different initiatives such as Espial or the Opera Mobile for TV highlight the relevance of these technologies on the roadmaps of the TV browser developers.

Anyway, according to the obtained results, and even if the capabilities of the devices are going to increase rapidly, users are already demanding advanced 3D applications on digital home devices. Section V shows the related work on different approaches to increase the capabilities of the devices to render 3D content and introduces the *3DMaaS System* proposed in section VI, which allows to extend the capabilities of the devices pushing to the cloud complex 3D rendering tasks and combining it with its hardware possibilities on a hybrid system.

V. Related Work

A solution based on remote rendering performed by a high processing cloud server with enough network bandwidth resources can keep the target performance while achieve interoperability widen the audience. The server would manage all the 3D Media involved in order to render the 2D result according to the user actions. Last but not least, standard mechanisms to adapt the video stream to the network capacity can solve bandwidth problems. However, this solution delegates the final performance to interaction latency. Different approaches driven by the described solution, face digital home device's applications to overcome the current limitations in terms of 3D processing and rendering.

The gaming sector is the main driver for graphics computation. There are many advances when combining the computational load of the local machine with remote rendering by sending complex calculations to a remote server using proprietary approaches. [11] proposes the Games@Large System oriented to set-top boxes on home networks and for enterprises such as hotels. [12] extends the Games@Large System with the main idea to calculate motion vectors directly from the 3D scene information used during rendering of the scene. The concept of Gaming as a Service (GaaS) is presented on [13] where the quality of experience and the latency is a key factor of success.

Similar hybrid computation approaches also tackle visualizing 3D objects on other sectors. These solutions consist on sending graphical commands such us roto-translation parameters from the end client to the server. This way the server can calculate the strictly necessary data that the end client needs and stream it offering a progressive reconstruction of the polygons. These solutions are valid for a mere combination of 3D objects, but not extensible for 3D Media based applications.

[14] proposes a remote rendering scenario for mobile devices like PDAs running a dedicated application called Mobile 3D Viewer. This approach is based on the Chromium software [15]. [16] presents an approach sending 3D graphical commands in a stream from the server to the client and it is based on WireGL [17].

SHARC System [18] is an approach for enabling scalable support of realtime 3D applications in a cloud computing environment. It is based on service virtualization with tools like VNC. This solution extends VNC as a video streaming platform. VNC and similar virtualization tools are also used on [19] and [20].

[21] presents a MobiX3D mobile player for access 3D content through mobile devices using OpenGL ES. In section VI we present the *3DMaaS System* which does not require a specific player o application on the client side, running on a HTML5 browser. The approach exploits the potential of WebGL, based on OpenGL ES 2.0, leveraging 3D processing on mobile devices by delegating 3D WebGL rendering to a remote server. *3DMaaS System* enables the 3D Media content based applications by means of adaptative video streaming from the server side to the end device.

VI. 3DMaaS System Design and Experiments

A. Design

There are three main actors on the *3DMaaS System*[22] (Figure 4): MaaS Manager (MM), Rendering Server (RS) and their communication with the end devices. The features that *3DMaaS System* requires are really affordable for any kind of end device. Moreover, the cloud rendered stream is adapted to the different codecs and parameters to represent the media content at the different end devices (set-top boxes, smartphones, tablets, etc.). A block diagram of the general architecture is shown in Figure 5 and all the modules are more deeply explained below.

1) MaaS MANAGER (MM): This module enables the supply of 3D Media content as a service to the final client. MM separates the final client from its context and makes the right decisions in order to provide the content in the proper way for each situation to combine it with the local 3D scene of the end device. It has 3 different blocks:

- **Negotiation block**: It analyzes the 3D Media content description required by the end device on the initialization and its rendering capabilities provided with the scripting

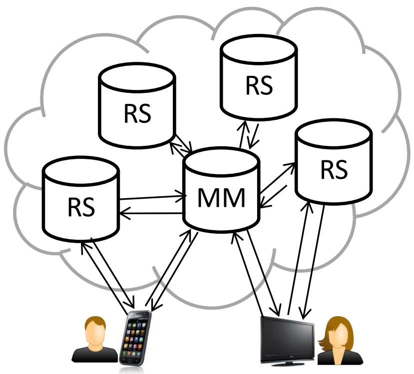

Fig. 4. General infrastructure of the *3DMaaS System*

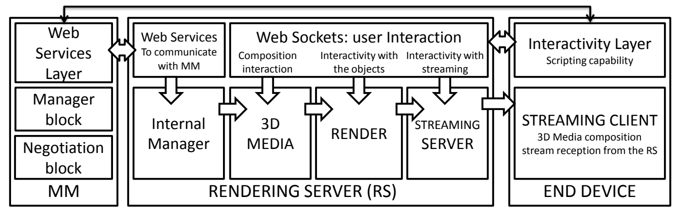

Fig. 5. The block diagram of MM and RS and their communication with the end device

language in the HTML5 application. According to this information it negotiates which part of the content is rendered at the device itself, which part at the server and how to adapt the content to the context.

- **Manager block**: During the negotiation the computational load required at the server side is also estimated. Depending on this, MM decides what RS may offer the service. MM reports the entire context and the decisions taken to the chosen RS. A direct communication between the end-device and the RS is created regarding this information, and the 3D Media stream starts.
- **Web Services layer**: It is used for communication with the final client and the RS. The end client performs the initial connection request to MM through Web Services. The same communication protocol is followed between MM and the RS in order to exchange context information and the needs of the service for a specific client.

2) RENDERING SERVER (RS): This module is in charge of performing the 3D Media composition, encoding it in real-time and sending it as a video stream to the end device. It allows the interaction of the final client both with the content and the parameters of the stream in order to adapt the needs of the context. Each RS is able to manage several specific clients depending on the computational capability of the RS.

3DMaaS offers a system that manages a resource pool to achieve a target QoS for any request volume. Each RS forgets about the rest of them and focuses on answering the requests made by MM, who is responsible of continuously monitoring the load state of each RS and of load balancing strategies.

Figure 5 shows the different block of the RS and its communication with MM and the end device:

- **Web services with MM**: MM reports the context to the RS to establish a new connection between the RS and the end device.
- **Internal manager**: This block creates and manages the streams in the RS to answer different users. This block also informs the MM about the changes of the different sessions.
- **3D Media & Render**: According to the context information this block generates the initial objects needs for the composition and it can be modified in any moment by the user. It is also possible to interact with each of the objects in the composition.
- **Streaming server**: It encodes the content in real-time and establishes the streaming session with the final client. All the initial parameters, both 3D Media and the streaming session are configured depending on the context information reported by the MM: Supported streaming protocols[23], codecs, screen size, etc. of the device, communication bandwidth, etc.
- **Web sockets for user real-time interaction**: Once the streaming communication is established between RS and end device, TCP web sockets are used to assure low-latency. The user can interact with the content in three different ways: changes on the composition (add new elements, delete them, move their position, resize them, etc.); modifications over each object (3D movements, texture changes, stop or rewind a video or audio, etc.); and adjustments of the streaming parameters (video resolution, bitrate, codec, etc.).

3) END DEVICE: The capabilities required by *3DMaaS System* for the HTML5 application of the client are really affordable for most of the common digital home devices. It only has to include *video* tag with the video streaming address provided by the RS and scripting capabilities to send HTTP interaction parameters. Their target is twofold: establish a new connection with *3DMaaS System* on a initial negotiation through MM; and for delivery of TCP web socket requests for low-latency interaction once the streaming communication is running with the RS.

B. Experiments

The critical performance metric of remote rendering solutions is the experienced latency for delivering a frame after graphics rendering update driven by user interaction. Our approach based on a video streaming server for 3D interactive application overcomes latency challenge. It is tackled by the hybrid solution proposal combining remote rendering of background 3D objects, where latency does not have a high impact on the user experience, with local browser WebGL rendering of foreground 3D objects which require low latency. The application server processes the user input and renders new screen frames and transmits them to the device in real time. Moreover, [23] presents *3DMaaS System* results for low latency streaming applications achieving 27.84 ms latency score. The hybrid strategy minimizes the number of objects that the browser have to render optimizing performance. For this, various experiments were carried out in order to assess the efficiency of the proposed architecture for visualization of 3D scenarios through digital home browsers. Users interact with the 3D rendering applications running on an accelerated graphics back-end for remote rendering and web browser for local rendering, allowing highly interactive experiences regardless of the complexity of the scene being considered.

Since visualization frame rate experienced at the mobile client constitute the main limitation of 3D web based applications, especially when considering complex 3D scenes, frame rate driven analysis tests have been designed in order to accurately quantify critical parameters of our hybrid visualization system, thus providing an effective measure of the performance of the proposed architecture.

Unfortunately, none of current TV sets are not able to deal with 3D rendering tasks. These devices cannot perform 3D WebGL applications due to lack of specific hardware but can also benefit from the *3DMaaS System* pushing to the cloud the whole rendering scene instead of building an hybrid rendering approach. Therefore, the experiments performed to define the performance thresholds on hybrid scenarios have been focused on mobile devices.

The tests have been done over the same devices described in section IV-B in order to measure the frame rate achieved for each device with the *3DMaaS System*. But in this case two superimposed HTML5 canvas have been involved. The one on the front is the simple 3D scene described on section IV-B, composed by rotating cubes. The canvas in the back is a <*video*> tag receiving a live video stream from the remote rendering server with the 3D background.

In order to set up the tests, the same range that defined in section IV-B has been employed for the number of 3D objects as well as their polygonal complexity in the front canvas. This way the performance combining the 3D local rendering capabilities and video stream reception on the different mobile devices is compared with the obtained measures on section IV-B with a mere local 3D rendering.

Figure 6 compares the frame rendering time for a number of polygons performed in a Samsung Galaxy TAB including the 3D object canvas and the live video stream visualization, with the measures obtained for the same 3D scene without the background video stream. The aggregation of the remote rendered live video stream does not have a considerable impact on the performance adding just an extra constant CPU demand. This way, rendering time for advanced applications with demanding 3D capabilities are not penalized by the added video stream. The GPU turns into a bottleneck from 10^6 triangles for this simple 3D scene, so this barrier settle the complexity that can be afforded by the device GPU without performance drawbacks. From this point remote rendering would make possible complex scenarios with no GPU overhead keeping the interactivity performance of the application. So this approach provides the application enriched 3D rendering capabilities, extending the device's hardware through remote rendering.

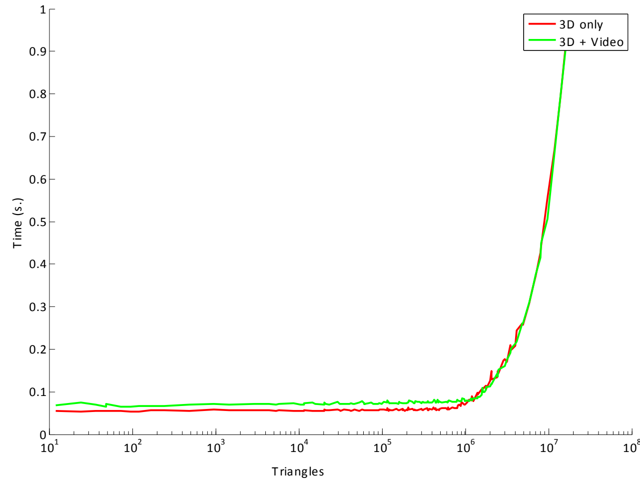

Fig. 6. A comparative of the frame rendering time and the number of polygons rendered in the Samsung Galaxy TAB, for 3D contents and added remote rendered live video stream to the 3D contents

The results obtained by the proposed architecture for hybrid remote and local rendering enhance the interactive experience of 3D graphics on digital home devices, proving the feasibility of interactive navigation of high complexity 3D scenes while provides an interoperable solution that can be deployed over the wide device landscape. However, this approach transfers responsibilities related to synchronization and OpenGL state consistency of local and remote 3D scenes to the application.

VII. Conclusions

Digital home application is a very disruptive market overcoming the potential that the Internet has and the incorporation of mobile devices together with the evolution of the TV to Smart TV in the digital home. These devices increasingly depend on reliable software to offer a good user experience. However, the current digital home platform landscape is highly heterogeneous, with different operating systems resulting in barriers to achieve cross-platform development and testing processes for digital home applications. New envisaged applications could engage with information and services exploiting the context. However, context awareness for pervasive applications introduces new challenges for ensuring that the desired user experience is achieved. The hardware and software of the devices vary so many that it is difficult to achieve portability feature across platforms. Hence the current trend in developing interoperable applications is to use web technology instead of platform-specific APIs.

HTML5 and WebGL are fully aligned with this trend by providing the Web as a software platform for interoperable applications. They offer device orientation, geolocation management and 3D rendering, bringing from native features to web-centric development. However, constraints to render interoperable complex 3D environments are still present especially in digital home devices such as TVs, set-top boxes, smartphones and tablets. Results described around the browser limitations to render 3D scenes of these devices, become evident the need to improve the performance of 3D applications over the digital home browsers to satisfy the prospects of the users, even if the these devices are being fitted with improved low energy-consumption GPUs.

In order to overcome this problem, the *3DMaaS* approach introduced in this paper, deploys remote servers performing the remote rendering of complex 3D scenes and then sending the frame results to a digital home device. This video streaming server approach pushes part of the graphics generation logic to the cloud and, in essence, turns the end device into a thin terminal. Driven by latency constraints, our approach proposes a hybrid solution combining remote rendering of background 3D objects, where the latency does not have a high impact on the user experience, with local browser WebGL rendering of foreground 3D objects which require low latency. Synchronization and 3D scene consistence challenges must be managed by the HTML5 application and the related complexity depends on its domain. Experiments show the results obtained by the proposed system for hybrid remote and local rendering enhance the interactive experience of 3D graphics on digital home devices proving the feasibility of interactive navigation of high complexity 3D scenes while provides a interoperable solution that can be deployed over the wide device landscape.

ACKNOWLEDGMENT

The authors would like to thank the Spanish Ministry for Industry, Tourism and Commerce, and the European FEDER funds for supporting the research activities involved in this paper.

REFERENCES

[1] E. L. Sasu Tarkoma, "Arching over the mobile computing chasm: Platforms and runtimes," *IEEE Computer*, pp. 22–28, 2011.
[2] M. Butler, "Android: Changing the mobile landscape," *IEEE Pervasive Computing*, pp. 4–7, 2011.
[3] T. M. Antero Taivalsaari, "The death of binary software: End user software moves to the web," *Ninth International Conference on Creating, Connecting and Collaborating through Computing*, pp. 17–23, 2011.
[4] ——, "The web as an application platform: The saga continues," *37th EUROMICRO Conference on Software Engineering and Advanced Applications*, pp. 170–174, 2011.
[5] T. M. A. T. Matti Anttonen, Arto Salminen, "Transforming the web into a real application platform: New technologies, emerging trends and missing pieces," *Proceedings 2011 ACM Symposium on Applied Computing*, pp. 800–807, 2011.
[6] S. Ortiz, "Is 3d finally ready for the web?" *Computer*, no. 1, pp. 14 –16, jan. 2010.
[7] ——, "Bringing 3d to the small screen," *Computer*, no. 10, pp. 11 –13, oct. 2011.
[8] R. Azuma, "A survey of augmented reality," 1997.
[9] P. Daras and F. Alvarez, "A future perspective on the 3d media internet," *Media*, pp. 303–312, 2009.
[10] T. Zahariadis, P. Daras, and I. Laso-ballesteros, "Towards future 3d media internet," *NEM Summit*, p. 1315, 2008.
[11] A. J. P. F. Arto Laikari, Jukka-Pekka Laulajainen and F. Bellotti, "Gaming platform for running games on low-end devices," *In Proceedings of ICST Personalization in Media Delivery Platforms*, p. 4, 2009.
[12] P. Fechteler and P. Eisert, "Accelerated video encoding using render context information," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*, sept. 2010, pp. 2033 –2036.
[13] C. Moreno, N. Tizon, and M. Preda, "Mobile cloud convergence in gaas: A business model proposition," *Hawaii International Conference on System Sciences*, pp. 1344–1352, 2012.
[14] A. S. Fabrizio Lamberti, "A streaming-based solution for remote visualization of 3d graphics on mobile devices," *IEEE Transactions on Visualization and Computer Graphics*, p. 14, 2007.
[15] R. N. R. F. S. A. P. D. K. J. T. K. Greg Humphreys, Mike Houston, "Chromium: A stream-processing framework for interactive rendering on clusters," *ACM Transactions on Graphics (TOG)*, p. 10, 2002.
[16] G. Marino, P. Gasparello, D. Vercelli, F. Tecchia, and M. Bergamasco, "Network streaming of dynamic 3d content with on-line compression of frame data," in *Virtual Reality Conference (VR), 2010 IEEE*, march 2010, pp. 285 –286.
[17] G. Humphreys, M. Eldridge, I. Buck, G. Stoll, M. Everett, and P. Hanrahan, "Wiregl: a scalable graphics system for clusters," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '01. New York, NY, USA: ACM, 2001, pp. 129–140.
[18] J. E. Weidong Shi Yang Lu, Zhu li, "Scalable support for 3d graphics applications in cloud," *2010 IEEE 3rd International Conference on Cloud Computing*, p. 8, 2010.
[19] P. Simoens, F. De Turck, B. Dhoedt, and P. Demeester, "Remote display solutions for mobile cloud computing," *Computer*, no. 8, pp. 46 –53, aug. 2011.
[20] Y. Lu, S. Li, and H. Shen, "Virtualized screen: A third element for cloud mobile convergence," *Multimedia, IEEE*, no. 2, pp. 4 –11, feb 2011.
[21] D. Nadalutti, L. Chittaro, and F. Buttussi, "Rendering of x3d content on mobile devices with opengl es," in *Proceedings of the eleventh international conference on 3D web technology*, ser. Web3D '06. New York, NY, USA: ACM, 2006, pp. 19–26.
[22] C. M. U. A. M. J. F. O. D. . O. I. G. Zorrilla, M., *"Next Generation Multimedia on Mobile Devices." Mobile Technology Consumption: Opportunities and Challenges.*, B. Ciaramitaro, Ed. IGI Global, 2012.
[23] M. A. M. J. F. G. J. O. I. G. Zorrilla, M., "End to end solution for interactive on demand 3d media on home network devices," *To be published on BMSB 2012*.