Mobile computation offloading architecture for mobile augmented reality, case study: Visualization of cetacean skeleton

Belen G. Rodríguez-Santana
Instituto Politécnico Nacional
SEPI-ESCOM
M´exico D.F.

Amilcar Meneses Viveros
Departamento de Computación
CINVESTAV-IPN
M´exico D.F.

Blanca Esther Carvajal-Gámez
Diana Carolina Trejo-Osorio
Instituto Politécnico Nacional
UPIITA
M´exico D.F.

***Abstract*—Augmented Reality applications can serve as teaching tools in different contexts of use. Augmented reality application on mobile devices can help to provide tourist information on cities or to give information on visits to museums. For example, during visits to museums of natural history, applications of augmented reality on mobile devices can be used by some visitors to interact with the skeleton of a whale. However, making rendering heavy models can be computationally infeasible on devices with limited resources such as smart phones or tablets. One solution to this problem is to use techniques to Mobile Computation Offloading. This work proposes a mobile computation offloading architecture for mobile augmented reality. This solution would allow users to interact with a whale skeleton through an augmented reality application on mobile devices. Finally testing to assess the optimization of the resources of the mobile device when performing heavy render tests were made.**

***Keywords*—Mobile augmented reality, mobile devices, render, mobile computation offloading.**

I. Introduction

Augmented reality (AR) complements the real world with computer-generated objects, its means AR is a window to the virtual world. The term mobile augmented reality (MAR) arises when integrating AR applications to mobile devices [1][2][3][4]. Initially, AR systems were very expensive and only used by small groups of researchers, such as the US Air Force to improve the efficiency of the tasks performed in flight simulation [5][6]. However, applications with AR have had great technological development and are already within reach of most people. This becomes more noticeable with the popularization of smart mobile devices, such as smartphone and tablets. The AR may serve as a teaching tool for different applications, as they can allow user interaction with 3D models or objects. Examples of these tools are medical applications for virtual surgery, or tourism information applications, among others.

The AR has been adopted in various fields and new forms of interaction between people with virtual environments are sought. The MAR is the result of the search for new forms of interaction. Visiting museums, such as the natural history may benefit from the use of mobile applications [7][8][9]. Where the use of a mobile device can be used by some museum visitors to interact with the skeleton of a whale and learn more about how the skeletal system is made of this mammal.

Interact with the skeleton of a whale using AR on a mobile device, it may be useful for visitors to natural history museums. The MAR would allows the visualization of a cetacean. This visualization can be the complete skeleton, region, or bone by bone. So, MAR technology will be highly advantage in this type of applications. Mobile devices such as smart phones, tablets, and digital cameras, among others, have had a major technological advances and they are a standard tool of communication for its portability. However, they are devices with limited resources. For this reason, the task of rendering within a mobile device may be impractical when the number of tiles to model graph is very large.

Rendering task inside a mobile device causes a high battery consumption, because the render takes a long time processing and also use the frame buffer actively. Some studies suggest that it is one of the main causes of battery consumption in smartphone [10]. Mobile Computation Offloading (MCO) it is technique to reduce the energy and time consumption for task in the mobile devices. The task of rendering within the mobile device depends on the complexity of the 3D object to be modeled (skeletal system of the whale or whale), this complexity is measured by the number of tiles that has the 3D object. In the image 1 we can see the 3D model of the whale. Image A shows the 3D object and the image B shows the wireframe or tiles of 3D model cetacean.

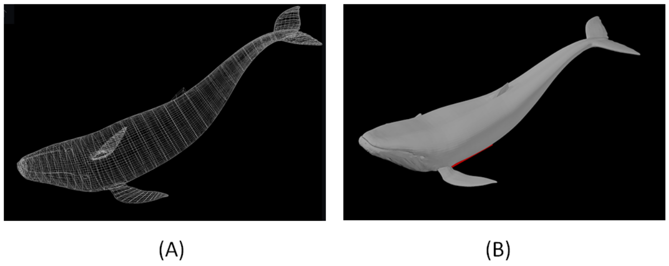

Fig. 1: 3D mesh model of the whale

The interactive visualization of the whale makes the MAR application is computationally complex. Also, the application must be efficient, in the sense that it must provide results in very short time for the user to maintain interaction with the application. The whale has about 172 to 176 bones [11] [12]. The amount of tiles for models of bones varies between 38 and 701440 tiles. This amount of tiles for the entire skeleton model generates several problems on the mobile device: power consumption, response times not suitable for an interactive system, and using a lot of memory.

This work proposes a mobile computation offloading architecture for mobile augmented reality. Also, we perform tests to find out where it is appropriate to local or remote rendering. These tests are: energy, time and memory by applying techniques MCO seeks to have an optimized use of resources of the mobile device to the tasks of the MAR application. This solution would allow users to interact with a whale skeleton through an augmented reality application on mobile devices. Also, this solution considers the time constraints required in HCI for proper interaction between the system and the user.

The paper is organized as follows. First, a brief discussion of related work of MCO applied to MAR applications is presented. Next, the system analysis is presented. Subsequently, the proposed architecture and its justification is shown. Later, we present a section of testing and result and finally the conclusions are presented.

II. Related work

There are many mobile applications that use AR, such as “Augmented Reality”, “Wikitud”, “Human Pacman”, among many others [13]. There are also MAR applications for museums, such as reported in [14][15][16][17]. In addition, various virtual reality applications have been developed for virtual tours in museums.

Some studies, such as [18][19][20], point to the need for MAR applications must adapt to the constraints of mobile platforms on which to run. This concept is not new, since previous work treated this adaptation as plasticity and is widely used in the context of development of type Rich-Client Applications [21].

The MCO is emerging as an alternative to optimize the use of limited resources of mobile devices. Some studies mentioned that it is important to study how to optimize the performance of mobile devices that are diverse in energy characterizing and modeling workloads to solve the weaknesses of MCO such as computational capacity, performance and energy consumption[10][22][23]. However the tests are performed for social networks, signal processing, games that require artificial intelligence module as chess or bingo, and has hardly been used for rendering.

Other studies mention that problems like energy consumption resulting from the use of graphics on the mobile device can be solved using techniques Mobile Cloud Computing (MCC), positive results in these tests are presented [24]. However quality connection Internet [25] is required, namely that it is stable, continuous, and good bandwidth and other features.

Make a combination of AR and render with MCO has several advantages as the optimization of workloads for mobile devices, and therefore, the appropriate use of device resources. For this reason, it should be an analysis of the requirements and resources they consume more battery as a mobile device [26], the requirements that a user waiting for an RA application workloads of application RA, the bandwidth required between the mobile device and a server to render the MCO, and the dimensions of time will render the server to perform the task.

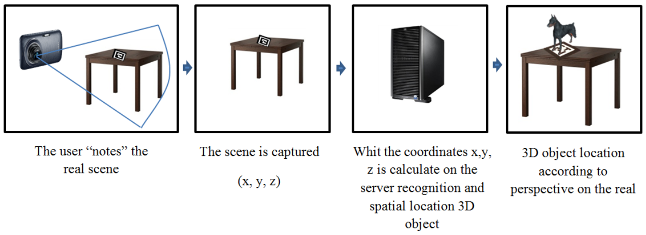
The user “notes” the real scene
The scene is captured
(x, y, z)
Whit the coordinates x,y, z is calculate on the server recognition and spatial location 3D object
3D object location according to perspective on the real

Fig. 2: Interaction system

III. System analisys

On the mobile device is going to implement an AR system based on markers, as shown in Figure 2. The use of markers is widely used in mobile devices for applications with AR technique. For this system the use of the marker will allow mobile camera calculate the distance and send a reference to the server.

To be able to optimize the resources of the mobile device such as memory and processing power will be applied MCO techniques, to determine whether the rendering of the 3D object to be performed locally, ie; within the mobile device, as shown in Figure 3, or remotely (outside the mobile device). The condition to be considered may depend on the management of resources on the mobile device, amount of computation, memory, bandwidth or time when the battery runs out. MCO techniques are to be employed in order to improve performance of the mobile device. Some conditions are:

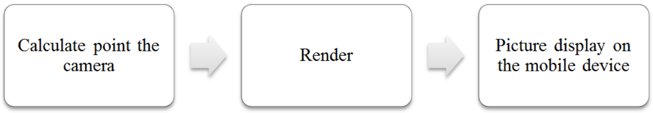
Calculate point the camera
Render
Picture display on the mobile device

Fig. 3: System without MCO

1) Compute the point of the camera: The camera of the mobile device help you to calculate the benchmark image. Considering three coordinates X, Y and Z in order to display the image in the mobile device.
2) Render: With the data obtained from the reference point is to run the rendering process to generate an image. If the image to be displayed contains few tiles rendering to be performed locally, ie within the mobile device, otherwise MCO applied.
3) Display the image on the mobile device: 3D object can be displayed on the mobile device.

The number of tiles that has the model is critical to whether to apply the techniques of offloading. For this case we can apply criteria of time, power consumption or memory usage.

In the case of time and energy consumption, the program can be divided in two parts. Part inherently running on the mobile device (as it is the GUI). And the other part that may be offloading. For the case where the processing time is important, offloading is applied when the following inequality is true:

$$t_m > t_T + t_s.$$

Where t_m is the time it takes to run in the region that can be potentially for Offloading on the mobile device, t_T is assigned to the transmission time of the data required for the calculation is run out of the mobile device, and t_s is the runtime the offloading of the server. That is, when the execution time is higher in the mobile device run time out of the device.

Usually, energy consumption is related to the execution time. The energy consumed by a process es the execution time multiplied by the power of the hardware components used to perform their computations. Thus, if the energy is the criterion for offloading, then the MCO applies if the following inequality holds:

$$t_m P_m > t_T P_c + t_s P_i.$$

Where P_m is the power of the mobile processor. The values P_i and P_c are the power required to send and receive data between the mobile device and server across the network. P_c is the power required by the wireless and P_i is the power that use the network interface to wait response from server. Thus, offloading can prolong the lifetime of battery consumption migrating computational heavy tasks to servers.

Where the criterion for memory usage is important, it is necessary to consider the processes running on mobile devices may have restrictions in their address space. For example, the processes running on Dalvick in Android, can use up to 54MB [27]. In addition, many mobile devices are restricted from RAM and not use virtual memory techniques.

The strategy of MCO that we use must be calculated experimentally. This is because the task of rendering is likely to halt, and there is no direct relationship between task execution times and energy consumption multithreading [28]. It totally depends on the hardware architecture of multicore processors. The complexity of this decision increases when considering the use of graphics acceleration units such as GPUs. For our case study, the complexity depends on calculating the image and this can be translated as the computational complexity.

The user may make use of your mobile device, whether smartphone or tablet to display the 3D object of the whale (blue whale) mainly bone and skin. It is proposed that the 3D model of the whale is in the server. This will allow much of the render process is within the server if the render process is very heavy, this decision will save the energy consumption of the mobile device. Then will have a client-server operation, where the client is the mobile device, as shown in Figure 4.

1) Compute the point of the camera: The camera of the mobile device help you to calculate the benchmark and it will send to the server. The computer that was used to send the point of the camera requires three coordinates X, Y and Z, we know that if these points are floating guy is talking about 12 bytes sent for him.
2) Send point the camera to the server: He sent will be made by WIFI connection type. The WIFI connection must have the following characteristics: at least 802.11a technology with a transmission speed of 54 Mbps and frequency band of 5 GHz .. However, this type of communication is not symmetric. That is, is not the same information as rising slope information.
3) Render: With the data obtained from the reference point of the chamber (12 bytes) is to run the rendering process to generate an image, which will be converted to PNG and then be sent to the mobile device.
4) Convert to PNG: The resulting image will be converted to PNG format so that the load is low and the obtained image data will not be lost at the time of conversion, the size of the image in PNG format it will be up to 3 megabytes.
5) He sent from server to mobile device: Once converted PNG image will be sent to the mobile device for display and interaction within the mobile device.
6) Mobile Device: Depending on what type of display you have the mobile device will allow determine which resolution you want the image to be received in this case is to consider the size of full definition that is 1080×1920 as main parameter. Table I shows the different types of display resolutions for tablets [29].

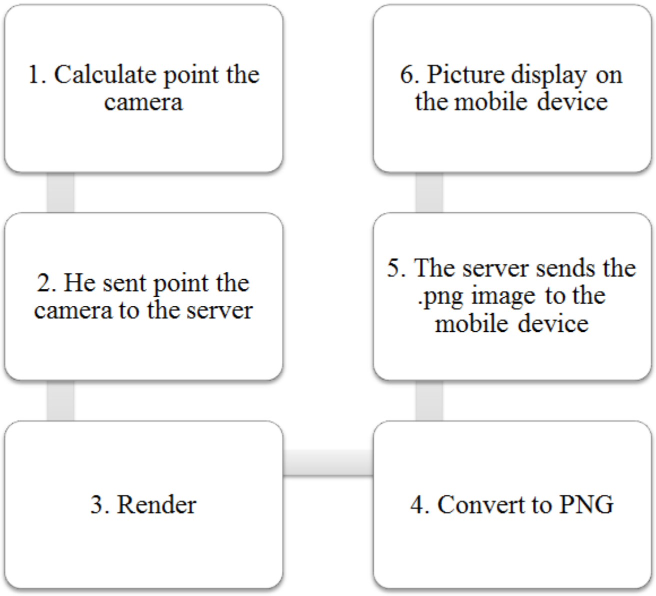
1. Calculate point the camera
2. He sent point the camera to the server
3. Render
4. Convert to PNG
5. The server sends the .png image to the mobile device
6. Picture display on the mobile device

Fig. 4: System with MCO

To describe of the sequence analysis (time constraints and data size to be transmitted. The user will send an action or event to a server using a mobile device, this action will be to send a reference point with a weight of 12 bytes. The file transmission will last approximately 0.0001 sec. Then the rendering process, which is performed; bytes are sent to an application that is within the server. The application makes

Tablet	Resolution	Size	PPI
Google Nexus 10 by Samsung	2560x1600	10.1in	300
iPad Air	2048x1536	9.7in	264
Asus Transformer	1200x1920	10.1in	224
Microsoft Surface Pro	1080x1920	10.1in	208
Samsung Galaxy Tab2 10.1	800x1280	10.1in	149
Google Nexus 7 by Asus	600x1400	7.0in	216
Samsung Galaxy Tab 2	600x1400	7.7in	170
BlackBerry PlayBack	600x1400	7.0in	169
Nook Tablet	600x800	7.0in	167

TABLE I: Tablet displays comparison

the rendering process where an image can be tif, bitmap or any other format is generated, subsequently calculated once obtained where the format is converted to PNG format. The PNG format is used because it is a lightweight format. It is noteworthy that the range obtained for the tempo of response should not exceed 0.2 seconds [30]. This time limit will be considered taking into account response time according to human perception, see Figure 5.

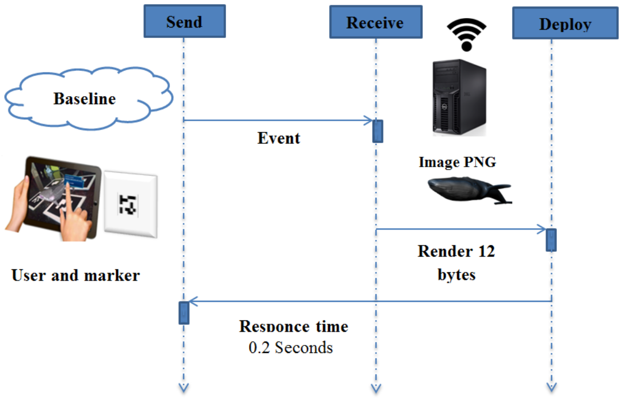

Fig. 5: Sequence diagram MCO

The problem to determine what kind of architecture for interaction is to be used is the level of detail that has the 3D object that can be rendered within or outside the mobile device, because the more higher the level of detail heavier the 3D object is made.

IV. Proposed Architecture

Once that is done the analysis of system requirements is proposed that the type of connection for communication a WIFI connection with 802.11a, a transmission rate of 54 Mbps and frequency band of 5 GHz, in order to meet the benchmark 3mb camera. With this kind of connection can be up to 15 connected users, it is noteworthy that the connection speed will depend largely on the process of loading images.

It is proposed to use as Unity 3D rendering engine, this render engine is used primarily for video games. However, Unity 3D has the following characteristics: cross-platform software, creating interactive 2D and 3D content, collaborative software, among others. As the above characteristics will be of great help to the rendering process.

The minimum characteristics of the mobile device are: 7” display, dual-core processor at 1GHz, android OS, resolution of 600 × 1024, 1G RAM and 3 megapixel camera.

A performance level for the display process of a whale (whale blue) may be performed as follows: When the user uses the camera of the mobile device is calculating the benchmark that will be sent to the server. This point is denoted by X, Y and Z. When the coordinate arrive at the server process may generate more two processes, which are also known as threads, the first thread stored in cache memory if the output images of the coordinate point and once was calculated, in order to be reused subsequently said images. However save images in cache may not be possible. The other thread is going to take to make the render and convert images to PNG. The thread that finishes first performing the process will be to send, see figure 6.

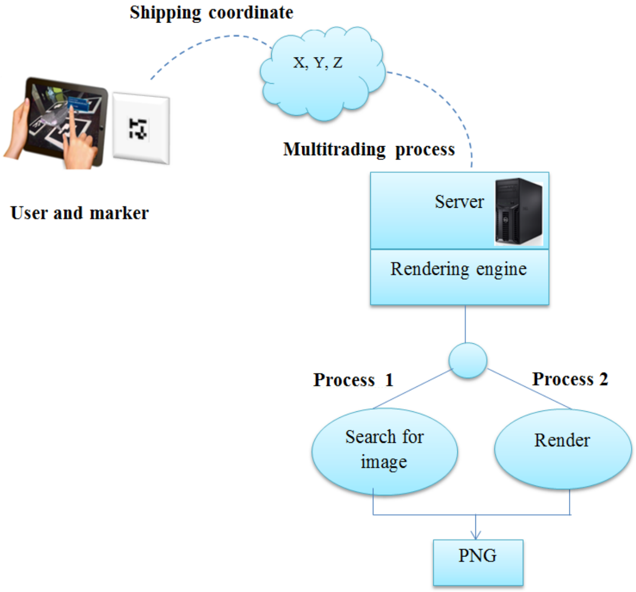

Fig. 6: Process Performance Level

The server has to be in a Red Hat Linux type either Fedora or CentOS, Filesystem minimum has to be a journalist type 3 Extension by the speed of access to the hard disk. The access must be direct route and there will be BD. Omission of a database is because it is very time consuming and would have to make a centered scheme documents for quick access, the same name of the database and distributing the directories. In one type 1.3 dual core arm cortex processor would take about 3.3 seconds while a k-5000 Nvidia 0.33 sec. A picture of 59104 tiles on a processor of these features can be 3 seconds. However if you switch to a server would be reduced to 0.33 seconds.

V. Tests and Results

Various 3D objects are tested and compared. Every 3D object has different features among which the number of tiles and if the 3D object has with / without texture. The tests were

Name	Total 3D models	Number of tiles	Object weight	Weight assembly
Cervical	8	198	150 KB	1250 KB
Thoracic	15	230	170 KB	3010 KB
Lumbar	11	205	160 KB	1800 KB
Tail vertebrae	23	48	70 KB	2760 KB
Ribs	30	59104	160 KB	11327 KB
Escapula	2	38	739 KB	28915 KB
Humerus	2	97	210 KB	2756 KB
Radio	4	75	168 KB	1807 KB
Ulna	40	56	77 KB	3900 KB
Head	1	701440	28254 KB	28254 KB
Total	136	-	-	-
Skin	1	14960	983 KB	983 KB

TABLE II: Table of characteristics of 3D models - Blue whale.

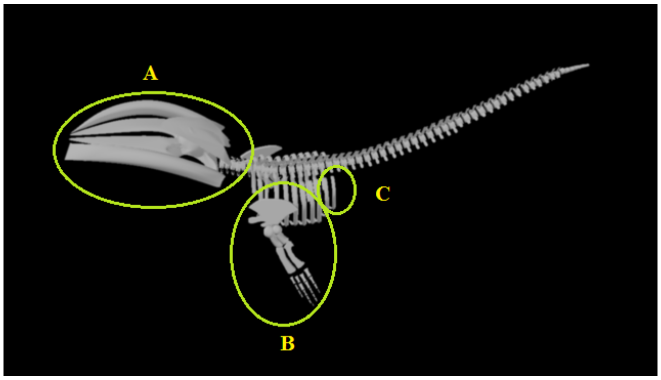

Fig. 7: 3D model of the skeletal system of the cetacean

carried out in two ways, the first is to make render within the mobile device and the second is the implementation of MCO in order to assess memory consumption, render time and energy consumption.

A. Features 3D model

3D models of the whale are developed using CAD tools and techniques of 3D modeling apply by mesh and 3D modeling by Solid. Once you have 3D models and objects are classified into two groups, the first consisting of 3D models with texture and the second non-textured 3D models.

Texture plays an important role. It is applied on the surface or 3D object model that is responsible for giving a better appearance and realism by allowing 3D models take the appearance of something real image. However, applying texture 3D models increases the complexity of the task of rendering.

As mentioned before, the skeletal system of the whale has 172 bones. However, the 3D model of the whale that was developed has 136 3D objects as some bones were simplified within the 3D object named “Skull”. We can see from the table II characteristics of the models and 3D objects of cetacean “Blue whale”. Render tests will be carried out taking into account the umber of tiles shown in the table.

B. Render tests

For testing render 3D models are to take into account the bones are mentioned below: Fin, skull, ribs and skin. Figure 7 present the 3D model of the skeleton of the whale. The part A is the 3D model of the skull which is considered to work as a heavy 3D object, B corresponds to the assembly of the 3D model of the fin which will be considered a small ensemble. And C is the 3D model of the rib which will be a lightweight 3D object.

C. Results

We are interested in analyzing the behavior of memory, performance and energy consumptions when the render is done. Figure 8 shows memory usage for cases that we selected in this test. Figure 9 shows the energy consumption involved in rendering. And finally, Figure 10 shows the runtime to perform the render.

Figure 8 present the behavior of memory usage when the render is local and remote. In this case the memory usage is largest than local because the PNG image size is bigger than the model 3D stored in the device. This behavior is very noticeable in the case of whale skin with and without texture. However, the PNG image size produce by the remote render does not exceed the memory available in the mobile device.

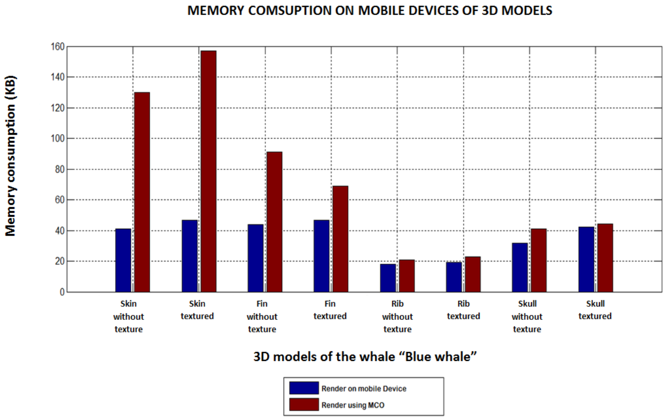

Fig. 8: Results of energy, time and memory without texture 3D models - Test your mobile device.

The energy consumption behavior is showns in Figure 9. We can notice that in all cases, remote render saving energy between 30 and 60 percent than local render.

The render time is shown in Figure 10. For this case, we are interested in the response time does not exceed 0.2 seconds, because is the HCI upper limit to respond to the user for a good experience. We can notice that local render have 6 tests that exceed this limit. For the remote render, only two cases exceed this limit. This cases are the skull with and without texture because this model is heavy to render.

In this work the case of the entire skeleton is not presented, because it takes too long to render local and presents problems of memory usage.

VI. Conclusion

MAR is an area that is being worked widely and consumes many computational resources. Most works that combine MCO

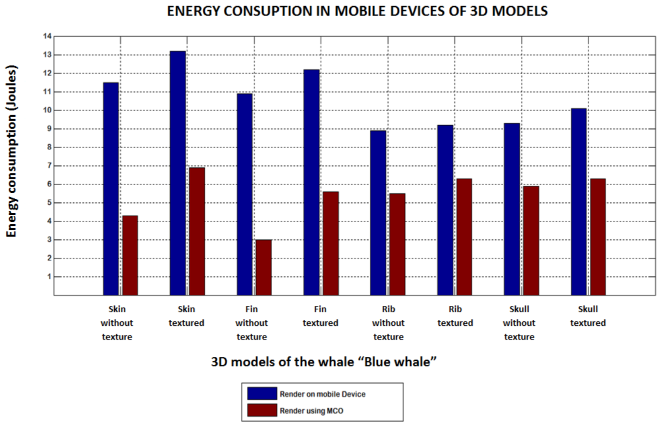

Fig. 9: Results of energy, time and memory without texture 3D models - Testing remote server.

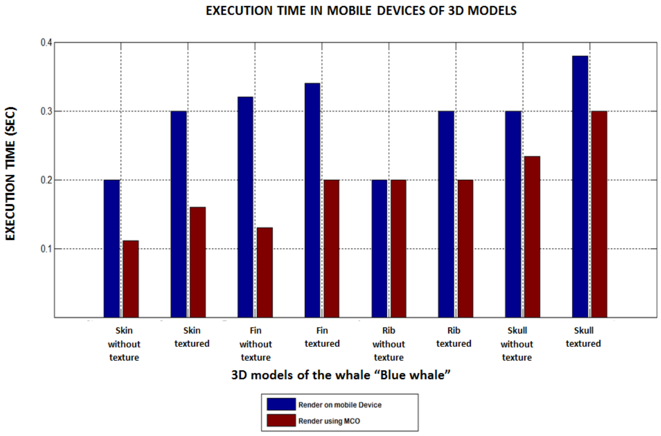

Fig. 10: Results of energy, time and memory textured 3D models - Tests on the mobile device.

and MAR have an approach to MCC. This approach generates a lack of control over resources as having: bandwidth, response times or server dependence, so if MCC is misconfigured there is no control of the internal parts of the server and you can get more expensive data transmission, in tasks where response time is critical. For our case study, it is more feasible to have the server in the museum and provide appropriate communication infrastructure. Mobile devices like tablets and phones have limited so heavy would render very time resources, battery and memory if done on the mobile device, so you should use MCO techniques. Developers have to consider mobile device restrictions such as cpu, battery and memory. Also, have to consider the constraints of HCI for interactive applications to be successful where the main constraint is time. A hardware level can consider using graphics accelerators on mobile devices. Using graphics accelerators low complexity of the task but does not know the rate of power consumption compared to a sequential program. Due to the scarcity of mathematical models to relate the behavior of time, energy, memory and updates that are in the hardware of mobile devices, the MCO decision factors would have to be done experimentally or perform correlation models. It can be considered to generate image retrieval techniques previously processed through shortcuts to the file system on the server. Another consideration is the use of streaming technology in skeletal sent, for viewing on the MAR application as streaming to employ more bandwidth is required to optimize unloading. So the model depends more bandwidth than the service capabilities to perform the render task. Communication is the main limitation to the number of users since the server can support at least 16 concurrent users if render is done in parallel but if sent concurrently is able to have up to 64 demands render. When working with mobile devices have to be connected antenna, it is essential to know what the right size of tiles to be considered under the proposed architecture, we mention that the complexity of the 3D model can increase if we add texture. MCO techniques can save energy and improve performance on mobile systems, this technique provides a solution to the demand for limited resources, ask users when interacting with games, videos or graphics. Which has been calculated to be part the infrastructure for communications based on the number of tiles and the resulting size in megabytes of the image, has been considered a standard format Full-HD yields on average 3 megabytes. We also study the different types of algorithms for partition and download information to improve performance or energy savings. MCO is a solution to increase the limited resources of mobile devices for a single user using immediate migration of information servers. Apart from the results obtained can be performed much future work as an important part observed in the test part is that the results render the 3D model change when it is zoomed in or out. Another factor that influences the quality of the .PNG image as the image is to be rendered.

From the test of local and remote render, we can observe that the implementation of MCO could be applied using the next algorithm:

```
Render(Model3D aModel, 3DPoint cameraPoint)
BEGIN
  IF (memorySize(aModel) < LIM_APP_MEMORY AND
    responseTimeOfLocalRender <  HCI_ThresholdTime AND
    energyForLocalRender < energyForMCORender)
    THEN
     localRender (Model3D aModel, 3DPoint cameraPoint)
  ELSE
      mcoRender(Model3D aModel, 3DPoint cameraPoint)
  ENDIF
END
```

Where `aModel` is a 3d model object, `cameraPoint` is the camera point for render, `LIM_APP_MEMORY` is the upper limit of memory for a application in the mobile device. If the memory size of the model is less than the limit of the memory of the application and the response time of the local render is less than the expected response time of HCI which is 0.2 s and the energy from the local render is less than the power remote render then it will make the 3D model rendering locally. In other case we chose the remote render.

Clearly, it is not difficult to apply the MC0 because the indicators used in this work vary depending on the device. However, it gives an idea of what could be the general behavior of these applications. Contrary to what you may think, we note that there is no direct provides between time and energy consumption, as shown in the case of skin render whale with and without texture. Perhaps this change in behavior is that the image of beauty skin framebuffer can use more than the other models. However, we note that in most cases, it's applying the MCO to do local rendering.

ACKNOWLEDGMENT

B.G.R.S., B.E.C.G. and D.C.T.O. would like to thank to IPN and Conacyt for their support to the project 221284. A.M.V. thanks Cinvestav-IPN, section of research and graduate studies (SEPI) of ESCOM-IPN, and Conacyt by the resources provided and the facilities for this work.

REFERENCES

[1] B. Ekengren, *Mobile augmented reality*. Skolan för datavetenskap och kommunikation, Kungliga Tekniska högskolan, 2013.

[2] B. Katier, "Mobile augmented reality," Ph.D. dissertation, Master Thesis, University of Amesterdam, 2011.

[3] T. Olsson and M. Salo, "Online user survey on current mobile augmented reality applications," in *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*. IEEE, 2011, pp. 75–84.

[4] S. E. Lee, Y. Zhang, Z. Fang, S. Srinivasan, R. Iyer, and D. Newell, "Accelerating mobile augmented reality on a handheld platform," in *Computer Design, 2009. ICCD 2009. IEEE International Conference on*. IEEE, 2009, pp. 419–426.

[5] D. Van Krevelen and R. Poelman, "A survey of augmented reality technologies, applications and limitations," *International Journal of Virtual Reality*, vol. 9, no. 2, p. 1, 2010.

[6] J. Carmigniani, B. Furht, M. Anisetti, P. Ceravolo, E. Damiani, and M. Ivkovic, "Augmented reality technologies, systems and applications," *Multimedia Tools and Applications*, vol. 51, no. 1, pp. 341–377, 2011.

[7] M. Kenteris, D. Gavalas, and D. Economou, "An innovative mobile electronic tourist guide application," *Personal and ubiquitous computing*, vol. 13, no. 2, pp. 103–118, 2009.

[8] C. Santoro, F. Paterno, G. Ricci, and B. Leporini, "A multimodal mobile museum guide for all," *Mobile Interaction with the Real World (MIRW 2007)*, pp. 21–25, 2007.

[9] E. Bruns, B. Brombach, T. Zeidler, and O. Bimber, "Enabling mobile phones to support large-scale museum guidance," *IEEE multimedia*, vol. 14, no. 2, pp. 16–25, 2007.

[10] A. Carroll and G. Heiser, "An analysis of power consumption in a smartphone." in *USENIX annual technical conference*, 2010, pp. 271–285.

[11] P. Flores, W. PERRIN, B. WÜRSIG, and J. THEWISSEN, "Encyclopedia of marine mammals," *Encyclopedia of marine mammals*, 2002.

[12] A. Berta, J. L. Sumich, and K. M. Kovacs, *Marine mammals: evolutionary biology*. Academic Press, 2005.

[13] A. S. Navia, "Realidad aumentada sobre web y vídeo en tiempo real: Plataforma de trabajo colaborativo para asistir al diseño arquitectónico," in *Congreso SIGraDI*, vol. 14, 2008.

[14] T. Miyashita, P. Meier, T. Tachikawa, S. Orlic, T. Eble, V. Scholz, A. Gapel, O. Gerl, S. Arnaudov, and S. Lieberknecht, "An augmented reality museum guide," in *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, 2008, pp. 103–106.

[15] D. Schmalstieg and D. Wagner, "Experiences with handheld augmented reality," in *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*. IEEE, 2007, pp. 3–18.

[16] A. Damala, P. Cubaud, A. Bationo, P. Houlier, and I. Marchal, "Bridging the gap between the digital and the physical: design and evaluation of a mobile augmented reality guide for the museum visit," in *Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts*. ACM, 2008, pp. 120–127.

[17] A. B. Tillon, I. Marchal, and P. Houlier, "Mobile augmented reality in the museum: Can a lace-like technology take you closer to works of art?" in *Mixed and Augmented Reality-Arts, Media, and Humanities (ISMAR-AMH), 2011 IEEE International Symposium On*. IEEE, 2011, pp. 41–47.

[18] T. Verbelen, T. Stevens, P. Simoens, F. De Turck, and B. Dhoedt, "Dynamic deployment and quality adaptation for mobile augmented reality applications," *Journal of Systems and Software*, vol. 84, no. 11, pp. 1871–1882, 2011.

[19] N. Ghouaiel, J.-M. Cieutat, J.-P. Jessel *et al.*, "Adaptive augmented reality: Plasticity of augmentations," in *VRIC: VIRTUAL REALITY INTERNATIONAL CONFERENCE*, 2014.

[20] S. Mendoza, D. Decouchant, G. Sánchez, J. Rodríguez, and A. P. M. Papis, "User interface plasticity for groupware," in *Digital Information and Communication Technology and Its Applications*. Springer, 2011, pp. 380–394.

[21] I. M. T. Hernandez, A. M. Viveros, E. H. Rubio, and B. Carvajal-Gamez, "Voice recognition framework for open rich-client mobile applications," in *Electrical Engineering, Computing Science and Automatic Control (CCE), 2013 10th International Conference on*. IEEE, 2013, pp. 302–306.

[22] A.-C. Olteanu, N. Tapus, and A. Iosup, "Extending the capabilities of mobile devices for online social applications through cloud offloading," in *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*. IEEE, 2013, pp. 160–163.

[23] K. Lai, H. Tang, H. Wang, S. Ding, and D. Wang, "Cloud offloading on customer-provided resources," in *Wireless Communications and Networking Conference (WCNC), 2013 IEEE*. IEEE, 2013, pp. 4695–4700.

[24] B.-R. Huang, C. H. Lin, and C.-H. Lee, "Mobile augmented reality based on cloud computing," in *Anti-Counterfeiting, Security and Identification (ASID), 2012 International Conference on*. IEEE, 2012, pp. 1–5.

[25] H. Luo and M.-L. Shyu, "Quality of service provision in mobile multimedia-a survey," *Human-centric computing and information sciences*, vol. 1, no. 1, pp. 1–15, 2011.

[26] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile Networks and Applications*, vol. 18, no. 1, pp. 129–140, 2013.

[27] D. Bornstein, "Dalvik vm internals," in *Google I/O developer conference*, vol. 23, 2008, pp. 17–30.

[28] R. I. Ramirez, E. H. Rubio, I. M. T. Hernandez, and A. M. Viveros, "Differences of energetic consumption between java and jni andoid apps," in *Integrated Circuits, ISIC'2014, Proceedings of the 2014 International Symposium on*. IEEE, 2015.

[29] T. D. Buskirk and C. Andres, "Smart surveys for smart phones: Exploring various approaches for conducting online mobile surveys via smartphones*," *Survey Practice*, vol. 5, no. 1, 2013.

[30] J. M. Carroll, "Human–computer interaction: Psychology as a science of design," *International Journal of Human-Computer Studies*, vol. 46, no. 4, pp. 501–522, 1997.