Improving the Performance of CAR Systems Based on Mobile Phones

Victor Fernandez Bauset
PhD Student
University of Valencia. Spain
Victor.Fernandez-Bauset@uv.es

Juan M. Orduña, Pedro Morillo
PhD advisors. Departamento de Informática
University of Valencia. Spain
{*Juan.Orduna,Pedro.Morillo*}*@uv.es*

Doctoral Dissertation Colloquium Abstract
***Keywords*-Real-time and embedded P & D Applications; Distributed Multimedia Systems; Mobile and Wireless Applications; Cooperative Information Systems and Applications**

I. Introduction

From the beginning of Augmented Reality (AR) systems, wearable devices were used to provide Collaborative Augmented Reality (CAR) systems, where a local user with a wearable device could collaborate with a remote user at a desktop computer [1]. On other hand, Mobile phones have become the most extended example of wereable devices [2]. They are considered as an ideal platform for CAR systems, due to the multimedia hardware they include, like full color displays, integrated cameras, fast processors and even dedicated 3D graphics chips [3]. CAR applications being executed on CAR systems are usually split into four stages: first, the image acquisition, through the onboard camera. Second, the position and orientation tracker stage. At the end of this stage, the application being executed on the mobile phone knows the position and orientation of the desired zone. Third, the rendering stage, where the application uses the position and orientation obtained in the previous stage to draw the information that will augment the real world, like 3D object or a text with some kind of knowledge. Finally, the last stage is the sending stage, where each device shares information with other client devices, normally through a server. Nevertheless, the wide variety of current mobile phones, with different graphic and processing capabilites, and different operating systems, can have significant effects on the performance of the four stages (in terms of system latency, frames per second or number of supported clients with certain latency levels), particularly if the CAR application is executed on a large-scale CAR system. These effects should be taken into account when implementing CAR applications based on mobile phones, in order to avoid a performance degradation in terms of both system latency and throughput. Thus, CAR applications like virtual reality guides for visitors through painting collections within a museum (or any other application that should support tens or even hundreds of clients) require the characterization and improvement of CAR systems based on mobile phones. Concretely, the objective of this thesis dissertation is the performance improvement of Collaborative Augmented Reality (CAR) Systems based on mobile phones.

The proposed research allows the selection of the most appropriate mobile phones for developing CAR applications, determining the maximum number of clients that a CAR application can support while maintaining interactive response times for different types of mobile phones. Also, it helps to increase the scalability of CAR systems based on mobile phones, since many potential system bottlenecks have been detected so far and different implementations of CAR servers have been proposed.

CAR applications can be typically divided into two groups, one of them using markers and the other one using feature detection, which correspond to the second stage of CAR. There are few solutions based on fiducial marker tracking over mobile phones. In 2003 ArToolKit, one of the most well-known software libraries for building Augmented Reality (AR), was developed for Windows CE, and the first self-contained AR application was developed for mobile phones [4]. This software evolved later as the ArToolKitPlus tracking library [5]. A tracking solution for mobile phones that works with 3D color-coded markers was developed [6], and later a version of ArToolKit for Symbian OS was developed, partially based on the ArToolKitPlus source code [3]. The research teams behind these works have worked on fiducial marker tracking, but not from the collaborative point of view. Also, there are many other works that focus on natural feature tracking as can be seen on [5]. On the other hand, as the computational power of mobile phones increases, the feature descriptors are becoming more common. Currently, it is common that high-end devices are built with CPUs working at frequencies up to GHz, GPUs and at least two cores. Since this new hardware is capable of supporting more computational workload, a new library based on natural feature tracking, developed by Vuforia and called Qualcomm [7] has widely extended as a common tool for developing AR applications. Regarding the characterization of mobile phones, there are some previous works [8], but taking into account how fast new devices appear in the market, these works are old enough to be considered as obsolete.

II. Method and Proposed Solution

We have focused on the objective of improving the performance of CAR Systems based on mobile phones by means of different phases. First, we carried out the experimental characterization of mobile phones when used

Table I
EXECUTION TIME (MS) PER STAGE IN DIFFERENT MOBILE PHONE

Stages(ms)	Acq.	Detect	Render	Send	Total
Milestone	248,64	288,53	30,42	14,14	698,34
Nexus One	40,25	78,08	13,23	5,54	167,11
iPhone 3G	33,29	58,07	28,26	15,42	398,00
iPhone 4	17,66	182,17	23,34	7,06	523,26

in CAR applications. There are different kinds of mobile phones, with different operative systems (OS) and capabilities. There are a number of mobile operating systems: Nokia Symbian, Google Android OS (commonly referred as Android), RIM/Blackberry, Apple iOS, Microsoft Windows Mobile/Phone 7 and Samsung Bada [9]. In this thesis, we are focusing on two of them, Android and iOS, because they share the vast majority of the current market [10]. We performed a characterization from the point of view of latency and throughput by decomposing the action cycle in the four stages of CAR applications described above. Next, we carried out a performance characterization of CAR systems from the server side. Starting from a typical current computer, we implemented a CAR server and we measured its behavior, also in terms of latency and throughput, for an increasing amount of clients in the simulation, . The third and final stage is the removal of any potential system bottleneck that may arise in these systems. We have followed the same methodology for both approaches in the position and orientation marker stage (marker-based and markerless CAR systems), in order to sweep the main trends in CAR systems and/or applications.

III. CURRENT STATUS AND RESULTS

We have implemented a CAR application on a real system and we have measured the performance achieved with different mobile phones, considering two operating systems: Android OS and iOS. As an example, table I shows the time required for executing each stage in different mobile phones. Concretely, the first four columns show the average duration of each stage per cycle for each device, and the rightmost column shows the total aggregated cycle time.

The results show that the most time consuming stage in a CAR application is the marker detection stage, followed by the image acquisition stage, the rendering stage and finally, the transmission stage. The results also show that the rendering stage is decoupled in devices executing Android OS, in such a way that it is concurrently executed with the rest of stages. Finally, the results show that some recent mobile phones like iPhone 4 only works with high resolution images. As a result, these mobile devices achieve the most visual quality at the expense of requiring a lot of time for detecting the markers in the camera image plane [11], [12].

We also carried out a performance characterization from the server side, measuring the system response time

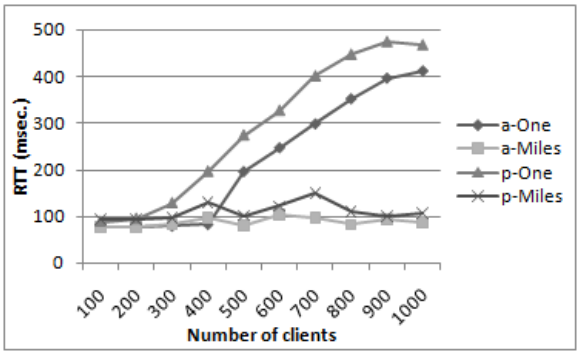

Figure 1. Average system response times for a working group size of 20 neighbors

and system throughput when varying different systems parameters like the number of clients in the system, the number of neighbor clients to which the updating messages should be sent, and the cycle time of clients. As an example, figure 1 shows the average system response times obtained for a CAR application of working group sizes of 20 neighbors. Each plot in this figure correspond to a configuration of either a passive or active server implementation (different implementations based on TCP [13]), and using the HTC Nexus One or the Motorola Milestone as client devices. The X-axis shows the number of clients in the system, while the Y-axis shows the average system response times in the simulation.

Figure 1 shows that a CAR application using a standard server and with working groups of 20 neighbors (each client device should send each updating message to 19 neighbor clients) can support up to four hundred client devices while providing interactive response times [13], [14]. Also, these works show that, as it could be expected, the system saturation point depends on the overall percentage of CPU utilization in the computer platform acting as the system server. Although the CPU threshold is not a fixed value, it is inversely related to the number of processor cores. Additionally, the results showed that the CAR systems throughput heavily depends on the kind of client devices, although for certain kind of devices the system bottleneck is the server I/O.

Next, we have developed different server implementations, with the purpose of removing the system bottleneck due to the server I/O. In order to achieve this goal, we have changed the sockets type from TCP to UDP, and we have also used the sockets' SELECT operation in order to reduce the amount of application threads and avoid deadlocks. We have made a comparison study including all the implementations. As an example, table II shows the results for the server implementation based on UDP sockets when using working group sizes of 25 clients devices.

Table II (and the rest of results, not shown here due to space limitations) show that the server implementation based on UDP sockets significantly improves the performance, both in terms of latency as well as in the system

Table II
RESULTS FOR A WORKING GROUP SIZE OF 25 NEIGHBORS

	UDP implementation				
Size	RTT	Dev	CPU	RT_S	% lost
100	9.86	6.78	72.50	4.06	0.83
200	21.70	14.73	82.00	9.84	1.18
300	26.01	21.91	79.60	11.61	0.69
400	39.41	30.66	81.90	18.26	0.83
500	48.68	39.68	83.80	22.84	0.74
700	79.70	97.87	85.10	37.26	0.76
1000	122.37	85.35	85.00	44.98	0.90

throughput, with respect to the rest of the implementations, at the cost of dropping a small percentage of messages. Thus, a CAR system with this server could support up to one thousand clients [15].

Along the making of this research work, a new platform that covers the detection phase of CAR, called Vuforia [7], has emerged. Since it provides the user with an easy and fast method of markerless detection, its use as rapidly extended and it has become a de-facto standard in CAR applications. In order to make our research as complete as possible, we are characterizing both mobile phones and different server implementations with this new platform. We plan to repeat the whole process of characterization and enhancement of the CAR system, with the purpose of publishing the results for this platform in the short term.

IV. FUTURE RESEARCH

As a future work, we plan to make source code optimizations in order to reduce the duration of the most time consuming CAR stages, that is, the first stage (acquisition) and the second stage (position and orientation tracker). In the latter one, we want to study the use of mobile's Graphics Processor Units (GPU), since it is a common feature in nowadays mobiles. There are a few works at the moment [16], but they are not focused on mobile phones. The huge number of cores existing in current GPU provides these devices with computing capabilities that can be exploited by the marker tracking or the characteristic extraction that represent the second stage on CAR applications.

ACKNOWLEDGEMENTS

This work has been jointly supported by the European Commission FEDER Funds and the Spanish MINECO under grant TIN2009-14475-C04-04.

REFERENCES

[1] T. Hallerer, S. Feiner, T. Terauchi, G. Rashid, and D. Hallaway, "Exploring mars: Developing indoor and outdoor user interfaces to a mobile augmented reality system," *Computers and Graphics*, vol. 23, pp. 779–785, 1999.

[2] A. Henrysson and M. Ollila, "Umar: Ubiquitous mobile augmented reality," in *Proc. of 3rd Int. Conf. on Mobile and ubiquitous multimedia*, ser. MUM '04. New York, USA: ACM, 2004, pp. 41–45.

[3] A. Henrysson, M. Billinghurst, and M. Ollila, "Face to face collaborative ar on mobile phones," in *Proc. of Int. Symp. on Mixed and Augmented Reality (ISMAR)*, October 2005, pp. 80 – 89.

[4] D. Wagner and D. Schmalstieg, "First steps towards handheld augmented reality," in *Proc. of 7th IEEE Int. Symp. on Wearable Computers*, ser. ISWC '03, 2003, pp. 127–135.

[5] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg, "Pose tracking from natural features on mobile phones," in *Proc. of the 7th Int. Symp. on Mixed and Augmented Reality*, ser. ISMAR '08, 2008, pp. 125–134.

[6] M. Mohring, C. Lessig, and O. Bimber, "Video see-through ar on consumer cell-phones." in *ISMAR'04*, 2004, pp. 252–253.

[7] D. Wagner, I. Barakonyi, I. Siklossy, J. Wright, R. Ashok, S. Diaz, B. MacIntyre, and D. Schmalstieg, "Building your vision with qualcomm's mobile augmented reality," in *Int. Symp. on Mixed and Augmented Reality (ISMAR)*, oct. 2011, p. 1.

[8] D. Wagner and D. Schmalstieg, "Making augmented reality practical on mobile phones, part 2," *Computer Graphics and Applications, IEEE*, vol. 29, no. 4, pp. 6 –9, july-aug. 2009.

[9] T. Ahonen, *TomiAhonen Phone Book 2010*. TomiAhonen Consulting, 2010.

[10] S. P. Hall and E. Anderson, "Operating systems for mobile computing," *J. Comput. Small Coll.*, vol. 25, pp. 64–71, December 2009.

[11] V. F. Bauset, J. M. Orduña, and P. Morillo, "Performance characterization on mobile phones for collaborative augmented reality (car) applications," in *Proc. of Int. Symp. on Distributed Simulation and Real Time Applications*, ser. DS-RT '11, 2011, pp. 52–53.

[12] ——, "Performance characterization of mobile phones in augmented reality marker tracking," in *Proc. of Int. Conf. on Computational and Mathematical Methods in Science and Engineering*, ser. CMMSE '12, vol. 2, July 2012, pp. 537–549.

[13] V. Fernández, J. M. Orduña, and P. Morillo, "On the characterization of car systems based on mobile computing," in *Int. Conf. on High Performance Computing and Communication (HPCC)*, june 2012, pp. 1205 –1210.

[14] V. F. Bauset, J. M. Orduña, and P. Morillo, "Characterization of car servers for augmented reality marker tracking," in *Actas de las XXIII Jornadas de Paralelismo*, ser. JJPP '12, Elche, Spain, Sept. 2012, pp. 598–603.

[15] ——, "On the implementation of servers for large scale car systems based on mobile phones," in *Int. Conf. on Computer Vision, Imaging and Computer Graphics Theory and Applications*, ser. GRAPP 2013, pp. 381–384.

[16] Y. Allusse, R. Grasset, and M. Billinghurst, "Accelerating template-based matching on the gpu for ar applications," in *Int. Symp. on Mixed and Augmented Reality (ISMAR)*, nov. 2007, pp. 271 –272.