Multimed Tools Appl
DOI 10.1007/s11042-017-5115-x

Interactive mosaic building and its application to marketing strategies using NFC

Irene Luque Ruiz[1] **· Gonzalo Cerruela García**[1] **·**
Miguel Ángel Gómez-Nieto[1]

Received: 5 February 2017 / Revised: 10 August 2017 / Accepted: 13 August 2017

Abstract Although the building of photo mosaics has been widely studied, the existing solutions are not appropriate for solving problems where the photos are not already known. In this paper, we describe an efficient solution to interactive photo mosaics building in real time through the incorporation of new images provided by users. Efficient matching algorithms and data structures make possible the incorporation, in real time, of new images whose visualisation can be created from mobile devices, producing high quality and realistic mosaics. The application and validation of our solution using Near Field Communication (NFC) as a trigger for user interaction has demonstrated its usefulness for its application in marketing strategies.

Keywords Photo mosaic · Matching algorithm · Marketing · NFC

1 Introduction

The building of digital mosaics is an area of research of image treatment in which different techniques of NPR (non-photorealistic rendering) have been studied in order to improve the visualisation of large or complex images, or simply for artistic purposes [23].

From a general perspective, independent of its application, a digital mosaic is a collection of small images called tiles that, grouped together, reproduce an illustration or base image

✉ Irene Luque Ruiz
iluque@uco.es

Gonzalo Cerruela García
gcerruela@uco.es

Miguel Ángel Gómez-Nieto
mangel@uco.es

1 Department of Computing and Numerical Analysis, University of Córdoba, Campus de Rabanales, Albert Einstein Building, E-14071 Córdoba, Spain

Published online: 21 August 2017

(background image) simulating a mosaic. Depending on the set of tiles and the rules or constraints placed on their assembly, it is possible to build a wide variety of types of mosaics that have captured the attention of researchers in recent years [3, 7, 8, 34, 47, 65, 72].

According to Battiato et al., [9] the building of a digital mosaic entails resolving a problem of optimisation that may be defined in the following way: "*Given a rectangular region* I^2 *in the plane* R^2, *a tile dataset and a set of constraints, find N sites* $P_i(x_i, y_i)$ *in* I^2 *and place N tiles, one at each* P_i, *such that all tiles are disjoint, the area they cover is maximized and the constraints are verified as much as possible*".

Thus, depending on the type of constraints placed on the building process of the mosaic, many different types can be created for specific purposes of problem-solving in computer graphics (examples of different mosaic types can be observed in Fig. 1):

– Ancient or Classic Mosaic: in which the constraints require tiles to be placed according to a direction vector $\Psi(x, y)$, there is no overlapping of tiles, all of which follow the same direction and are the same colour as the space of the base image.
– Puzzle Mosaic: in which the tiles are disjoint, of irregular shape, as is the space that they occupy in the base image, and their position must be that which is "most like" the tile.
– Photo Mosaic: in which the tiles have a regular shape, generally square or rectangular, are disjoint and their position must be that which is "most like" the tile.

Different techniques have been proposed for each of these types of mosaic that try to reproduce artistic mosaics using computational geometry algorithms. Thus, solutions have been developed based on Voronoi diagrams [20, 31, 42, 45], rendering techniques based on lines of direction and border detection using segmentation techniques, such as SRM (Statistical Region Mergin) [10, 57], or gradient vector and tile cutting [11] in order to generate Ancient Mosaics.

Regarding puzzle mosaics, among the most relevant proposals are those of Kim and Pellacini [37, 38] who used functions for minimising the final energy of the mosaic by means of the difference in colour between the tiles and the background, gaps between tiles, overlapping and the shape or misshape of the tiles, building what are known as Jigsaw mosaics, and those of Di Blasi et al., [18, 19] which generate a "Magic Mosaic" using matching techniques based on Antipole data structure and directional guidelines working on a channel of luminescence and image histograms.

Fig. 1 Examples of ancient (*left*), puzzle (*center*) and photo mosaic (*right*)

Finally, the building of a photo mosaic is what has attracted the most attention, wherein many commercial products have been developed, such as Runaway Technology [64], ArcSoft PhotoMontage 2000 [60] and others which we will describe later in this manuscript. The building entails substituting portions of parallelograms of equal or different size of a background by tiles of the same size, whose colour characteristics are as similar as possible. In this process, generally, there is no analysis of image characteristics, such as edges or shapes, but rather an application of efficient matching algorithms, data structures and searching techniques in databases. Thus, Chi et al., [15] propose the use of QuadTree method and self-organising map clustering in order to cut the computing costs of the process and generate different visualizations of the mosaic without the need to rebuild.

Despite the popularity of mosaics and their use by companies and artists for different purposes, including encrypting information in images [36, 46], their application has not been sufficiently widespread in marketing campaign strategies, where they can have an even wider variety of applications. However, in recent years their potential in the world of marketing has taken on some importance, being used by companies as a means of promoting products [56], events [67] including charity events [49] among other promotional activities. Applications that use several mobile sensors have also applied to study the tourist travel behaviour [39, 59].

Independently of the algorithms used in this process, existing solutions are based on generating the mosaic in one global step from a database of images that does not allow participation from different users in its building, nor does the generated mosaic support any other functionality apart from its own visualisation.

While the building of a mosaic from a database of images is a process that, despite its computing cost, is a good solution for the large number of existing applications, it is not so when the aim is for mosaics to be built step by step, in real time and interactively.

In the step by step process there is no pre-existing image database. The characteristics of the new image to be inserted are unknown and the assignment of the new image to a cell must be done in real time and with the greatest possible similarity.

The proposal described in this paper is geared to creating interactive mosaics that can be used as a marketing tool for strategies of diffusion of information, client acquisition or loyalty, among other factors. To do so, mosaics must satisfy the following technical characteristics:

- The mosaics can be built interactively in which new, previously unknown images are incorporated with the greatest possible similarity.
- They must allow different cell sizes and shapes and image-treatment functions that enable the most faithful visualisation possible of the background image.
- They must allow you to determine which areas of the background will be used, thus enabling the creation of more realistic mosaics.
- The insertion time of a new image must be minimal (less than 2 s), thus enabling real time visualisation.
- The insertion of a new image and the visualisation of the mosaic must be able to be carried out from any mobile device in real time.

In order to satisfy these requirements, it is necessary for there to be adequate data structures and matching algorithms that allow the assignment of images to cells with the greatest similarity in the least possible time, as well as visualisation of mosaics in real time in mobile devices in which there are limitations in the speed of data communication, screen characteristics, processing, etc.

In this paper we describe a useful data structure allowing the building of a photo mosaic with different cell size and format at time of high realism thanks to the organization of these cells into slots and the pre-processing of the image's background, the definition of usable, not usable slots. New images uploaded to the mosaic by the user are processed with color attributes, border and blending algorithms allowing to generate different photo mosaics and realism types. Using different similarity metrics, k-means clustering and the powerful data structure the mosaic is updated in real time (in less than 1 s) allowing to build the mosaic step by step and avoiding the needing of whole reconstruction of the mosaic carried out in other proposals. These characteristics of our proposal allow it to be used in a large type of applications, mainly marketing oriented. We describe in the manuscript that the use of NFC technology in these types of campaigns along with our photo mosaic solution could be useful.

The manuscript has been organized as follows: in Section 2 related works on the mosaic building researches are introduced. In Section 3 can be found the terminology used, the main functionalities of building and visualising mosaics are described along with the steps necessary for generating interactive mosaics. In Section 4 a study of different similarity measures for assigning images to cells are presented, as the mosaic characteristics (mosaic size, cell size, background image, etc) are sensitive to the similarity measure used for the assignment of images to cells. In Section 5 we describe our own solution for the creation of more realistic mosaics, including the image-treatment techniques and management of areas that are not assignable to the base image. Section 6 deals with the fine-tuning technique used for the building and rebuilding of mosaics. Finally, in Section 7, we describe the developed software solution, presenting its architecture, database and functionality, all of which has been validated in experiments described in Section 8. The conclusions of the proposal of this paper are presented at the end.

2 Related works

On the market there is a wide range of products [5, 13, 16, 26, 28, 32, 33, 51, 53, 61, 63] destined for mosaic building, all of which have many characteristics in common. Although some are Web applications (Go4mosaic [28], Picture Mosaics [61]), in general, they are applications of desktop ownership or shareware that start from a collection of images supplied by the user to build a mosaic, in which the user can often personalise a series of parameters of the mosaic, such as the size, cell size, shape and borders, etc.

Generally, these products operate in the following way:

1. The user selects the base (background) image of the mosaic and the set of photos with which to build it.
2. The user selects the values for the parameters allowed for the application.
3. A colour measure is obtained for each of the background cells.
4. The photos are loaded onto a database, and during this process a colour measure for the photo is obtained.
5. A measure of the similarity between the background cells and the photos is obtained by the colour measure of both obtained in the previous step.
6. The mosaic is built allocating each photo (repeated or not) to a background cell.
7. An efficient data structure is built for visualising the mosaic at different zoom levels.

There already exist a large number of proposals, libraries and software for the building of mosaics [26], AndreaMosaic [5], JImage Mosaic [33], YAPMG [63], C++ − Photo Mosaic library [13], Image Mosaic using C# [32], Mosaizer [16], Python Packages [51], etc.

Some products, such as Mosaic Creator [52], Mosaizer [6], Photo-Mosaik-Edda [25], allow various cell shapes, include blending and masking techniques, rotation and superimposing of the photos (Mosaikify [68], Mosaic Wonder [54]). However, most solutions require the base image to be square, the same size as the cells, or restricted to a list of predefined sizes.

Furthermore, existing solutions do not incorporate manipulation procedures of the base image (background), therefore do not allow the user to determine areas that are not used for the formation of the mosaic. Not all solutions include options for manipulating photos enabling the application of filters or blending, although some do allow generating border colours to the cells of the generated mosaic.

These functionalities of image treatment have been described in several papers, enabling mosaics to be built with cells of different size and disposition, as well as the use of blending techniques for improving visualisation of the mosaic at different zoom levels [4, 62, 69]. Thus, the proposals for generating existing mosaics are geared to finding a personal solution for the end user, building mosaics for his/her own printing or publishing on Web sites.

It is also necessary to consider that although not all products admit generation of large mosaics (images larger than 128 Mb), the most costly steps of the process are the loading of images from databases (Step 4) and the allocating of photos to the background cells (Step 5 and 6), due to the need to calculate the C (cells) x F (photos) measures of similarity for this process and, subsequently, calculate the best allocation of all the photos to all the cells.

This last process requires the solution to an optimisation problem that enables generating a mosaic in which the total difference in similarity between all the photos and cells is minimal. For this purpose, existing solutions are based on building complete mosaics in a process using photos that are previously stored in the database. Therefore, the use of these applications is poor and limited to the generation of mosaics for end users and their personal use, due to the fact that the modification of a previously-generated mosaic entails the repetition of all or most of the required steps (depending on the application) and a high cost and computing-time factor. Thus, if the user wishes to incorporate one or a set of new photos to the mosaic, the process must be repeated from the start.

As can be appreciated, the higher computing cost and the determining step of the process consist in resolving a problem of allocation with a minimum cost. These types of problems have been described and resolved for decades [40, 43] and their high computational complexity is known.

Most of these algorithms are only capable of dealing with matrices of squared costs (equal number of unknowns and variables). Thus, when the cost or similarity matrix is not squared (different number of cells and photos) the existing algorithms require the creation of fictitious rows or columns.

One of the most used algorithms is the Simplex method [24] which resolves the problem of minimum cost allocation by means of lineal programming. This method has an exponential computing cost in the worst case and is efficient when the number of elements CxF is not very high, which means it is not appropriate for large mosaics.

Another of the most used methods is the Hungarian or Kuhn-Munkres algorithm. This algorithm was proposed by Kuhn [41], and later improved by Munkres [55]. It has a complexity of $O(n^3)$ and, although very efficient in the first steps of allocation, again when the cost matrix CxF is large, in the final steps it requires many repetitions in order to find the final optimum solution.

With the aim of reducing the computing cost of this allocation process some data structures have been proposed to represent the similarity between background cells and photos. Thus, Kärner [35] proposes the building of a kd-tree of photo candidates for each cell and the use of lineal searching methods based on the nearest neighbours, which entails a computing cost of $O(n \log n)$ and $O(n^2)$ respectively.

Cantone et al. [14], also propose an indexed structure for storing similarities between photos and cells and use allocation methods based on the search for the nearest neighbour [27]. Brady et al., [12] present a wide range of implementations of the problem of allocation in parallel architectures. Sang-Un [44] forwards a proposal that reduces the Hungarian algorithm from 4 steps to 2 in the first place, the algorithm selects the minimum matrix cost and eliminates the remaining rows and columns and, in the second place, improves the solution by means of a process of reassignment. Hong Cui et al. [17] improve the original Kuhn-Munkres algorithm by using the sparsity structure of the cost matrix, proposing two algorithms: a) sparsity based and b) the parallel implementation.

Finally, the use of matching on bipartite graphs has been presented as a very robust and efficient solution for problems of assignment in which the graphs are large. Thus, Golin [29], proposes representing the cost matrix as a bipartite graph, thereby reducing the optimisation problem to a consistent combinatorial process for finding the maximum perfect weight matching, a proposal that will be used in our own solution.

3 Proposal of tailored mosaics

Throughout the manuscript we will use the following terminology:

– Background: is an image on which one wishes to build a mosaic.
– Photo/tile: is an image that substitutes cells or sets of bits of the background.
– Mosaic: is a background in which there are zero or many photos that have substituted cells or sets of bits of the background.

Thus, a mosaic is made up of two basic elements: a) a background image and b) a set of images that replace areas or bit cells of background in order to obtain a final picture whose visual appearance is the most similar to the background.

As we have already described, the existing solutions are based on using hierarchical structures to represent large images at different zoom levels and deep zoom visualisers that manipulate this structure.

Our aim is based on the reconstruction of a background image substituting cells or sets of its original pixels for pixels belonging to other images or photos. The mosaic formed must be the most similar to the background, so that from different zoom levels the visual appearance of the mosaic is as similar as possible to that of the background.

As already indicated, the building of mosaics using any type of background image is complex, as several problems arise to be resolved:

– The background image can have objects of any shape and size and, depending on which objects, one can want the pixels of the image they form to be substituted or not by pixels of the photo.

– The size of the background images and the distribution of colour may be very different, as in the photographs, so that it is necessary for the insertion of the photos in the background image to maintain the appearance of the background image, at least at low zoom levels; the visualisation of the photos must be clear, at least at high zoom levels.

3.1 Material and tools

The building of a mosaic entails the resolving of three problems: a) an appropriate structure to represent it so that its manipulation and visualisation on mobile devices is efficient, b) some adequate algorithms that carry out the assigning of photos to background cells so that the background is minimally affected visually and, c) a system of visualizing the mosaic which is efficient and also enables its visualization at different zoom levels as well as integrating extra functionalities for the improvement of the usability of the system.

Traditionally, the structures used for mosaic building are hierarchical ones in which each hierarchical level represents a zoom level of the mosaic.

This building process of the structure can be carried out in two ways:

– From top to bottom: in this case, for the upper level of the mosaic a cell size is selected that will store the background image with the lowest zoom. Recommended sizes are 128, 256 or 512, smaller sizes produce a very low resolution of viewing of the mosaic and larger sizes produce big distortions between the background image and the target mosaic image. The upper levels are built by generating a number of cells equal to the second level, making as many levels as necessary until all the cell sizes are equal or greater than the size of the original background image.
– From bottom to top: for this, again, we must decide on the cell size that will represent the background image in the first level of the structure, and that will determine the number of cells in which we will divide the image in the last level. The following levels are built by grouping lower level cells in the proportion 2 × 2, until reaching the upper level formed of one single cell.

Table 1 shows the building process of top-down and bottom-up for a background image of 8192 × 8192 pixel size and a cell size of 256 y 512 respectively.

Different tools have been evaluated to carry out this process. Deep Zoom Composer [22] is a Microsoft-owned application whose purpose is the preparation of images to be visualised by

Table 1 Example of construction of the hierarchical representation of the background

Top-Down Construction				Bottom-Up Construction			
Level	Image Size		Cells	Level	Image Size		Cells
	Size X	Size Y			Size X	Size Y	
0	256	256	1	4	8192	8192	256
1	512	512	4	3	4092	4092	64
2	1024	1024	16	2	2048	2048	16
3	2048	2048	64	1	1024	1024	4
4	4096	4096	256	0	512	512	1
5	8192	8192	1024				

using Silverlight 3 [21]. Due to it being an executable programme it is not possible to manipulate the generation parameters of the hierarchy, resulting in a large number of levels in which the cells are of different sizes and taking up a large amount of disk space and processing time (for a square image of 8192 pixels the process lasts a minute and generates files that take up 80 Mb).

Tile Generator [1] is an executable programme built in C# and publicly-owned that generates a hierarchical structure able to be interpreted by Google Maps [30]. The processing cost of the same example image takes around 8 min, generating a file structure of 200 Mb. It generates cells of 256 pixels and, although it is an open-code tool, its performance is not adequate.

Other existing tools were also tried out with similar results and producing different types of problems:

- The background image must be square, otherwise the user must make it so by adding the necessary transparent rows/columns, or in some cases this is done by the tool itself.
- The cell size at the lowest level of the hierarchy must be square and in most tools it is predetermined.
- The hierarchy structure is not compatible with standard visualising tools, such as Google Maps.
- The size of the photos that will make up the mosaic by substituting the cells or background pixel sets must be equal to the size of the cells.

Therefore, a tool has been built in pHp language that, starting from an initial image, generates the hierarchy of representation of the background in a format that is recognisable by these standard tools. The developed tools work in the following way:

- A background image is selected onto which the hierarchical structure of levels will be generated. There are no limits to the image size, which may be square or rectangular.
- The cell size of the lowest level in the hierarchy is selected. This cell may be square or rectangular, in which case it can have either a portrait or a landscape disposition. If the cell size is not a whole divider of the background image, the image is corrected by either increasing or decreasing the number of necessary pixels.
- The different levels in the hierarchy are built in a top-down process from the root of the tree, keeping the same cell size for all levels, until reaching a level whose cell number by size is equal to the size of the background image.

For the same square background image of 8192 pixels that were used to test the other tools, our algorithm took 5 s to build a hierarchical structure of 6 levels (from 0 to 5) with square cells of 256 pixels and requiring a disk space of 45 Mb. For rectangular images of 8192 × 4096 pixels and a cell size of 256 × 128 pixels, the time needed was 3 s to generate a tree of 5 levels and required a disk space of 35 Mb.

As can be observed, the algorithm's performance improves existing solutions, all the more so considering that, as we will describe later, in this process a series of calculations for the arrangement of cells and their information extraction is performed.

Another important aspect in the building of a mosaic is the selection of the cell or set of base image pixels that will be substituted for the photo pixels. To do so, the afore-mentioned tools do not give a solution due to their functionality being based on treating a static image and building the hierarchical representation of the mosaic only at different zoom levels. Our solution for this functionality is described in the Experimental section.

Finally, for the visualisation of mosaics, there are several types of tools. The best known, standard and most widely-used is Google Maps. The limitation therein is that, despite being free, it is owned by Google and therefore includes private advertising.

OpenSeaDragon [58] is a publicly-owned API written in JavaScript that works with several types of hierarchical structures (Microsoft, Google Maps, etc.). This tool uses Canvas to show the hierarchical structure and incorporates a wide range of functionalities that allow the use of square and rectangular cell sizes, includes markers, etc. Because OpenSeaDragon is open code, it may be used in any type of search engine without needing a specific plug-in.

3.2 Slots based mosaic building

Existing tools for mosaic building are based on using a cell size for the lowest hierarchical structure level and using this unit of pixels as the photo size for the mosaic. This one-to-one relation between cell size and photo is, as afore-mentioned, due to the dependence existing between the performance in building and manipulation of the hierarchical structure of the mosaic and its visualisation at different zoom levels.

Thus, a large cell size generates hierarchical structures of little weight and easy to manage, but the mosaic is then inappropriate due to it not being very similar to the background image, even when the photos are visualised with a high definition.

By contrast, a small cell size generates heavy structures, of many levels, which results in its visualisation at low zoom levels being of low quality, even though the mosaics have a greater similarity to the background image.

Furthermore, if the cell size determines the photo size, the resulting mosaics will only be able to keep a uniform appearance (square or rectangular), so that the building of the mosaic requires a "reconstruction" of the photos before being inserted in order to stick to these sizes.

In order to solve these problems, our proposal is based on the inclusion of one other level in the conceptual representation of the mosaic. This level consists on dividing each cell at the different levels of the hierarchical structure, in a set of slots. This is similar to the video frame representation in the High Efficiency Video Coding (HEVC) standard, in which video frames are represented by a tree structure obtained dividing in non overlapping coding tree units (CTUs) and splitting each CTU in coding units (CU) [66, 71].

A slot is a set of pixels of a cell that is to be used as a unit of substitution for a photo. They may be any size (either smaller than or equal to the cell), with the only limitation that the number of slots in a cell is a whole number, either square or rectangular (portrait and landscape).

This new level of representation enables us to build mosaics with ratios of 1:1 and 1:N between cells and photos, thus avoiding the afore-mentioned inconveniences. Besides this, given that the insertion of a photo requires that the mosaic image is as similar as possible to the background, as we will show later, by using a slot as unit of insertion, this leads to greater accuracy and closer similarity to the photo to be inserted.

Therefore, as the flow diagram in Fig. 2 shows, the process of hierarchical structure building is carried out in the following way:

- *Step 1*: The background image is selected to build the hierarchical structure that will be used to build the mosaic.

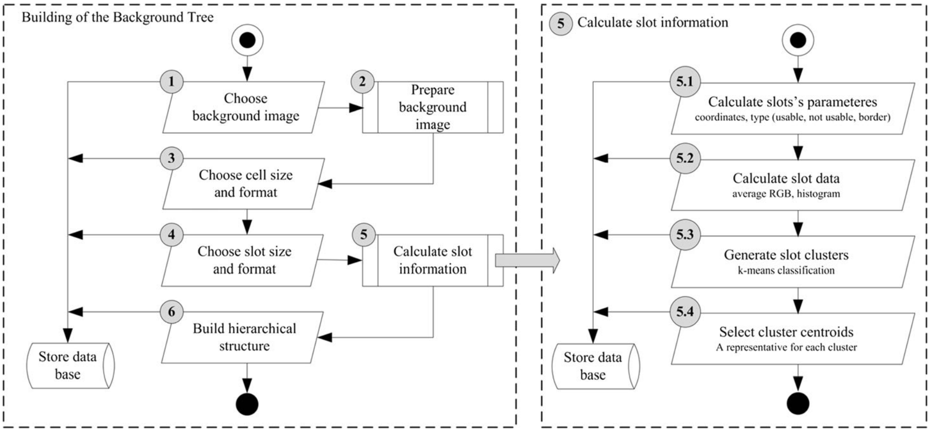

Fig. 2 Flow diagram corresponding to building the mosaic hierarchical structure

- *Step 2*: The image is treated using Adobe Photoshop [2], which consists on selecting those parts/pixels of the images that will make up the functional cells and which will then be filled with photos in the mosaic. The pixels that have not been selected will not be able to be substituted for photos in the final mosaic.
- *Step 3*: The size and shape of the cells is selected (square or rectangular), which will then determine the number of levels of the structure and the number of cells in each level.
- *Step 4*: Depending on the cell size, the slot size and shape is selected, which will determine the number of photos to be housed in each cell.
- *Step 5*: For each slot a set of calculation processes are performed to obtain the necessary information to be used in the slot selection for housing a photo (see Fig. 2 right). Due to the fact that the background image to be used in the mosaic building may have an irregular shape, there will be slots that can have parts of their pixels in unused cells, that is, those that do not make up the mosaic. These slots, called *border* slots, will have a special treatment that is described later in this manuscript.
- *Step 6*: Finally, the hierarchical structure is built storing all the information in the database.

4 A model for building mosaic similar to background

One of the main characteristics of mosaics is that their appearance should be as similar as possible to the background image, at least at low zoom levels, and to do this it is necessary for the photos to be inserted into slots with the most similar colour characteristics possible to those of the photos.

In order to achieve this aim, the study of the behaviour of two colour characteristics of slots has been undertaken: a) their average RGB and, b) their colour histogram. These two parameters are thus calculated for the last level of slots in the process of building the hierarchical structure.

The average colour is a parameter formed of three variables that correspond to the average of the RGB channels of all the pixels that make up the slot in the background.

Meanwhile, the histogram is built by considering four colour intervals (0–63, 64–127, 128–191, 192–255), and building a three-dimensional matrix which will store the frequencies of these intervals in the RGB channels.

To select the slot with the greatest similarity to the target image, a set of distance measures and similarity indices widely used in the treatment of images has been studied. They are the following:

$$Euclidean\ \ distance(P,S)=\sqrt{\sum_{i=1}^{n}(P_i-S_i)^2} \tag{1}$$

$$Bhatttacharyya\ index(P,S)=1-\sqrt{\sum_{i=1}^{n}\frac{\sqrt{P_i\times S_i}}{\sqrt{\sum_1^n P_i\times\sum_1^n S_i}}} \tag{2}$$

$$Square\ \ Chi\ \ index(P,S)=\frac{1}{2}\sum_{i=1}^{n}\frac{(P_i-S_i)^2}{(P_i+S_i)} \tag{3}$$

$$Correlation(P,S)=\frac{\sum_{i=1}^{n}\left(P_i-\overline{P}\right)\times\left(S_i-\overline{S}\right)}{\sqrt{\sum_1^n\left(P_i-\overline{P}\right)^2\times\sum_{i=1}^{n}\left(S_i-\overline{S}\right)^2}} \tag{4}$$

$$KL(P,S)=\sum_{i=1}^{n}P_i\times log\frac{P_i}{S_i} \tag{5}$$

$$Cosine(P,S)=\frac{\sum_1^n P_i\times S_i}{\sqrt{\sum_1^n(P_i)^2\times\sum_1^n(S_i)^2}} \tag{6}$$

$$Jaccard(P,S)=\frac{\sum_1^n P_i\times S_i}{\sum_1^n(P_i)^2+\sum_1^n(S_i)^2-\sum_1^n P_i\times S_i} \tag{7}$$

where: P represents the value of the parameter used for the photo, S the value corresponding to the slot and $\overline{S}$ and $\overline{P}$ the average parameter studied for the pixels of the slot and photo, respectively.

In the step by step insertion process in a mosaic it is necessary to make a comparison with each of the mosaic's free slots for each new image in order to obtain the slot with the greatest similarity. This is a costly process for large mosaics so, in order to reduce such a computational cost, as shown in Fig. 2 right, in the steps 5.3 and 5.4 we apply k-means clustering analysis [48] to the known functional slots, obtaining a series of clusters that group the slots with the most similar colour together and selecting a representative slot for each one of the generated cluster.

This process might be done by considering the average RGB or histogram. The computational complexity of k-means algorithm without considering neither the number of interactions nor the calculation of the slot distance is *O(nk)*, being *n* the number of slots and the number of clusters. However, as the process is executed as a pre-processed stage that does not affect the performance of the insertion of an image into the mosaic.

Therefore, the insertion of a new image in the mosaic has a low computational costs, consisting in the selection of the most similar cluster centroid *O(k)* and, later, the searching of the more similar empty slot belonging to the selected cluster *O(s), where* $s = n/k$ for ideal cases, when the classification process generates uniform clusters with the same number of slots.

This solution has proved to be very effective for mosaics with a wide variety of colours in the background. In images (such as stone monuments) in which the different slots are very similar in colour, very few clusters are generated with a large number of elements, which increase the selection time of the appropriate cell.

For the evaluation of colour characteristics and similarity measures that show improved behaviour, a square background of 8192 pixels was used. The cell size used was 256 pixels and a slot size of 64 pixels, also square-shaped. This selection resulted in the last level of the hierarchical structure being able to generate1024 cells and 16,384 slots. However, once the image for selecting the background pixels not to be used in the mosaic was pre-processed, the number of functional slots was 6752.

The photos used for the building of the mosaic were taken from the database FaceScrub [70]. In total there were 6752 randomly-picked photographs (from a database of over 15,000) and composed of images of people. The process used an algorithm that selected photos at random from the Scrubface database until all the functional slots were filled.

Figure 3 shows the performance of the process using the Euclidean distance (the other indices studied showed a similar behaviour) and considering the average RGB and the histogram for the calculation of the similarity between the photo and the slot.

When the average RGB value is used for this calculation, the insertion time is kept constant, at less than 0.1 s. Given that the calculation of similarity based on the histogram has a higher computational load, the insertion time is long when there is a large number of free slots, decreasing as the mosaic is filled.

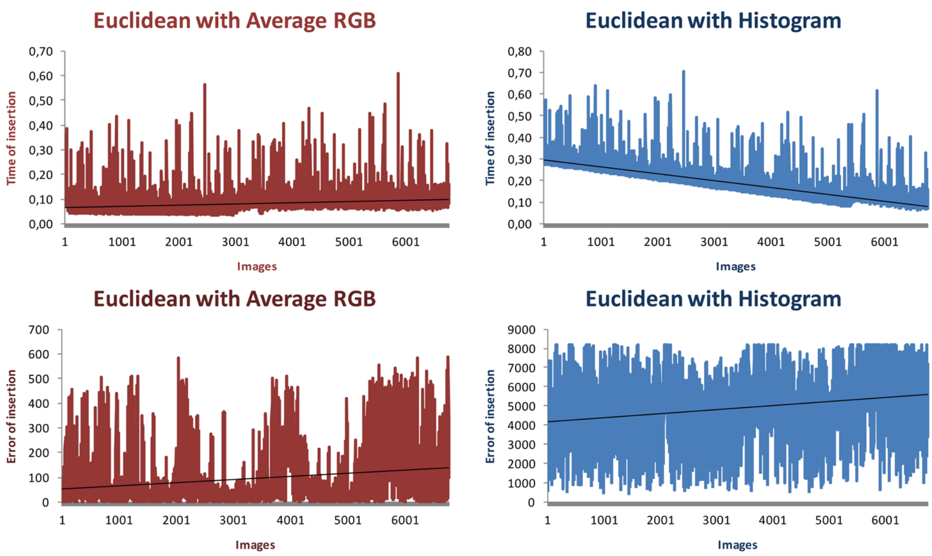

Fig. 3 Computing cost of mosaic building: time of insertion (*top*), error of insertion (*bottom*)

As can be observed, the average insertion time for a photo in a slot, including processing and storage time, is less than 0.1 s, without accounting for appreciable variables with changes in the colour characteristics of the slots and photos, nor for the density of filling the mosaic. The peaks are attributed to the computational load of the computer at that instant, owing to the fact that it shows in different photos/slots when the test is repeated several times.

Figure 3 also shows the error of assignment of the photo to the slot; these values are calculated as the difference between the variables used for the calculation of similarity, that is, the Manhattan distance between the slot and the photo, as can be seen in the following equation:

$$Error\ \ of\ \ Insertion(P,S) = abs\left(\sum_{1}^{n}(P_i - S_i)\right) \tag{8}$$

As can be observed, the error increases as the mosaic progresses, due to the fact that it is less likely for there to be empty slots with similar colour characteristics to the images left to be inserted.

The parameter used to measure the results was the total error of the mosaic's characteristics, which is calculated starting from eq. 8 and considering the RGB values of all the pixels of the slot and the associated photo.

Figure 4 shows each of the similarity measures studied, the average total error and the average insertion time for the tests carried out with the average RGB and the histogram. It can be observed that the error values are very similar for all the indices when the average RGB or the histogram are used. The worst behaviour is shown by the KL and the correlation indices.

The use of the histogram produces mosaics with a lower error of assignment, logically, given that instead of using 3 variables for the calculation of similarity with the average RGB, 64 variables are used with the histogram.

Likewise, the average insertion times are very similar when the average RGB is used and lower than those of the histogram, where it can be seen that the use of the correlation index requires a higher computing cost than for the other similarity measures.

Although the behaviour of the different similarity measures is very similar, the resulting mosaics are visually very different. This is clearly shown in Fig. 5 which represents the resulting mosaic after applying the Bhattacharyya and Jaccard indices to the same background using the average RGB and the histogram.

While the mosaic generated when using the histogram is visually very similar for the two similarity indices, although clearer in the case of the Bhattacharyya index, the clearest mosaic is that generated using the average RGB. However, if the average RGB is used with the Bhattacharyya index, the mosaic is distorted.

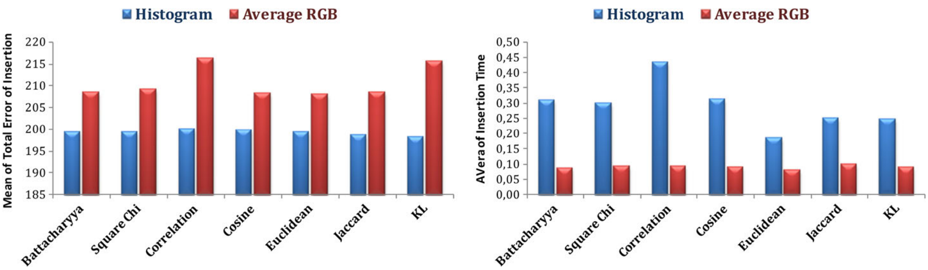

Fig. 4 Study of the insertion error and insertion time in the mosaic assignment for the similarity measures studied

Fig. 5 Mosaics generated using the Jaccard (*top*) and Battacharyya (*bottom*) indices with the average RGB (left) and the histogram (*right*)

In the study carried out it has been observed that the different characteristics of the similarity measures studied produce mosaics that are visually very different when using the average RGB or the histogram. This is due to the number of variables considered in each case and to the sensitivity of the index with small changes in their value.

Once the behaviour of all the similarity measures studied has been analysed, the indices of Jaccard, Chi Square and the Euclidean distance are the most satisfactory results obtained, and they will be the ones used in the next tests.

5 Tuning the mosaic visualisation

Once the colour and size similarity have been chosen to allocate a photo to a slot, the slot pixels are substituted for those of the photos. Whilst the resulting mosaic shows the closest similarity possible to the background image the visualisation should be able to be improved.

Due to the fact that the colour contents of the photos are essential for assignment to a slot, for a specific background image the resulting mosaics may contain major differences with the image background.

One example is shown in Fig. 6, where two very different background images have been used (Fig. 6a), and a mosaic has been created with the same set of photos.

As one can appreciate in the mosaic in Fig. 6b, although being very similar to the original background image, at high zoom levels distortion is evident at the edges which deteriorates the original image.

Fig. 6 Tuning the mosaics for the blending application: original image (*top*),0% blending (*centre*), 30% blending (*bottom*)

Moreover, Fig. 6-b having very different background image colour characteristics from the colour characteristics of the inserted photos, the mosaic appears very distorted compared to the background image even at medium zoom levels.

To refine the visualisation of the mosaics we have introduced a new parameter with the aim to lessen the differences in colour between the slots and the photos. This is the blending factor which ensures mixing the colour characteristics RGB/histogram of the slot with the photo using a function that determines the way that this is carried out. See Fig. 6-c, where we have applied 30% blending, notably improving the sharpness of the image obtained.

The blending factor and its function are selected in the building process of the hierarchical structure and stored in a database which is described in the next section.

The blending functions determine the geometry in which the blending factor is applied to the photo, which can be square/rectangular and circular/ellipsoidal in order to adapt the blending to the slot shapes.

In addition, for the photo/slot two zones or areas are designated for the blending application: central and peripheral, so that the results of applying blending can give the resulting mosaic different visual effects.

The blending factor is established by two values or limits for its application to the slots/photos: maximum and minimum, so that for each pixel of the photo, the blending value will vary according to the distance of the original pixel from its zone (inner or outer) in which it is applied.

There is also a percentage of improvement to the structure of the mosaic, which enables the visualization of the lower levels of the structure to be close to the original image and along with enlargement, clarifies the visualisation of the photos. This improvement percentage is used to diminish the blending values in proportion to the enlargement of the image.

Figure 7 shows different views without any enlargement and maximum zoom levels with blending factors varying from zero to 50 (using Jaccard index and average RGB for its calculation). As can be seen, blending factors at high levels ensure the mosaics retain the original image even under high magnification but the photo loses features of the objects that are included. Blending factors at low levels hardly improve the mosaic at the expense of greater computing cost.

When the colour characteristics of the background image are very different from those of the photos, high blending improves the appearance of the mosaic keeping the shape and characteristics of the background image.

Linear rectangular functions are more appropriate for background images in which the objects have straight lines (monument type), whereas circular ones are better adapted to backgrounds that are non-linear (flowers, etc.).

6 Tuning of mosaic building

With the aim to improve the visualisation of the mosaic, control of what we call slot expansion has been introduced, that is to say, those background slots that are not filled with a photo due to the fact that part of their pixels belong to the image background. These slots appear, due to their square or rectangular geometry, to sometimes have edges in the different areas of the background image that cannot house a complete photo.

To resolve this problem a partial fill-in process has been implemented for these slots. This process uses the same reconstruction algorithms and works in with a library of photos which are selected according to the background image.

Fig. 7 A study of the blending factor with rectangular lineal functions. From left to right; background image, factors of 0, 10, 20, 30 and 40

At the end of the process, as can be seen in Fig. 8, the mosaic does not show outlines in the background image which filled in to make up the mosaic. These outline slots (and their corresponding assigned photos) cannot be manipulated by the system and therefore cannot under any circumstances accommodate an incomplete photo.

Despite the techniques that have been developed to improve the visualisation of the mosaic, they can deteriorate due to the unpredictable process of assigning photos to slots when they are inserted. Although it is easy to find a slot with colour characteristics very close to a photo when there are a lot of free slots, as the mosaic gets filled the photos assigned to slots are less and less similar.

To resolve this problem, a mosaic reconstruction process has been designed. This process is executed in batch, so that it does not affect the insertion time of new photos.

Fig. 8 Mosaics with and without filled outline slots

The solution developed is based on an adopted algorithm Kuhn-Munkres [24], which is necessary, as mentioned above, because of the high computing cost of this algorithm and the large number of repetitions that are required in the final phase for high cost matrices (mosaics with a large number of slots).

In the case of mosaics, the matrix cost M, will be defined by a group of rows S, representing the number of available slots in the mosaic, and a series of columns P, representing the number of photos currently inserted in the mosaic, where $S \geq P$, and in which each element $M(i,j)$ of the matrix stores the similarity (or distance) of colour (average RGB or histogram) between the slot and the photo pixels.

Representing this matrix M, as a bipartite graph $G = (S,P,w)$, with w representing the distance or similarity between a slot $s \in S$ and a photo $p \in P$, as proposed by other authors [29], it is possible to find the optimum assignment if one obtains the perfect matching between the slots and photos, that is to say, the matching on the graph which includes all the vertices and satisfies the sum of weights as being maximum.

Using the Kuhn-Munkres algorithm on bipartite graphs reduces the problem of finding the maximum perfect weight matching to a combinatorial problem of finding the perfect matching, which reduces the computing cost to $O(nm)$ instead of $O(n^3)$ and eliminates the problem of assignment in the final stages of the original Kuhn-Munkres algorithm.

Using this technology, our algorithm resolves mosaics of 1000 slots/photos in less than 2 min and the results before and after the construction can be seen in Fig. 9.

As can be seen in Fig. 9 the reconstruction of the mosaics results in a better assignment of photos to the slots giving a more realistic appearance to the mosaics. As the time needed to reconstruct a mosaic is minimal, this process can be undertaken in batch processing at any moment, offering users very realistic mosaics.

7 Description of the proposed solution

Although there are many existing applications for mosaic technology, these are based on the complete construction of a mosaic using an existing database of photos. The aim of our paper is to create a mosaic step by step, without prior knowledge of the characteristics of the next photo to be inserted; the process is quickly undertaken from any device or mobile connection and the final mosaic allows its functional integration with other applications and social networks.

This proposed solution is based on using NFC (Near Field Communication) technology as a link between the user and the mosaic, allowing the creation of large scenes in which the objects that surround the users make it easier for the insertion of photos in the mosaic.

7.1 System architecture and interaction model

As shown in Fig. 10, the solution developed is based on taking into consideration the following factors: a) points of interaction (PoI), b) users and c) mosaics.

The PoI are assigned spaces with one or more elements of interaction which are used by the users to interact with the system. These elements of interaction are Smart posters with an associated NFC Tag.

The interactive elements are associated with one or various mosaics accessible to users by using PoI. Access to these mosaics allows the user to incorporate photographs. Management is

Fig. 9 Top (background images), Centre (mosaics before the reconstruction) bottom (mosaics after reconstruction)

undertaken by the Web services installed on the server side, which allows user interaction with Smart Posters without having to install in their Smartphone any mobile application, but only requiring the mobile device to have NFC incorporated.

As can be seen in Fig. 10, when users touch a Smart poster using a Smartphone supporting NFC, they receive the stored information in the Tag executing the Web browser of the mobile device and accessing to a Web service on the server side. This service asks for the users' identification, which can be done using their own system or social networks, having previously registered by any of the three following methods (system, Facebook or Twitter).

Once identified, the mosaic (or mosaics) associated with the Tag present in the Smart poster with which users have interacted, are then retrieved from the database. If there is more than one mosaic available, users select the target one and can select a photo stored in their device or instantly take a photo and upload it (such as a selfie).

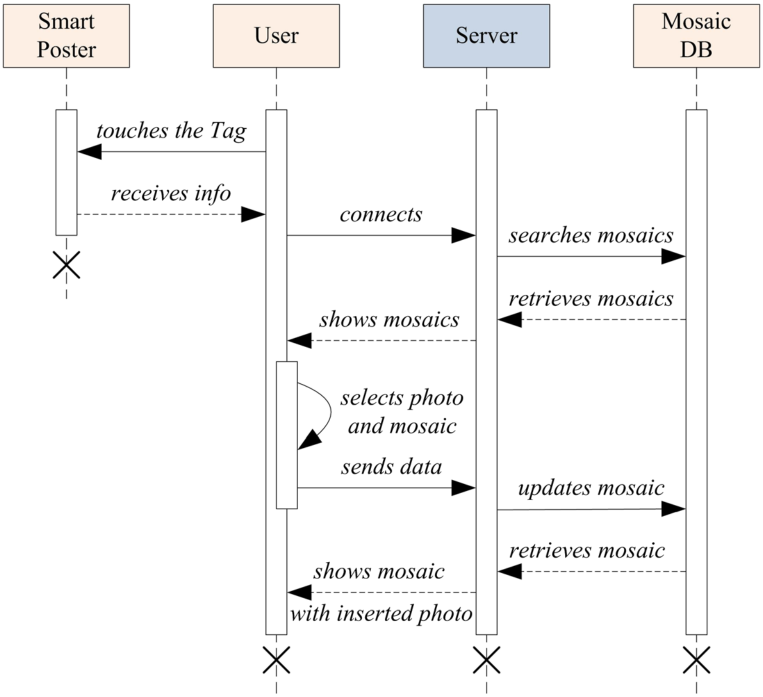

Fig. 10 Model of interaction of the use of mosaics in tourism

Once the server has received the users' photo, it is stored in the mosaic and the new mosaic is returned to the users showing that it is at their disposal.

The solution constructed allows users to be able to see the mosaic or mosaics in which their photos are included, as many times as they like and share this information on the social networks Facebook and Twitter.

Figure 11 shows some images captured by the user interacting with the system. The interface is made up of a viewer to present the mosaic which includes a search bar beneath (Fig. 11a), which enables the user to see the original mosaic enlarged (Fig. 11b), ask for help (?), enlarge (+), reduce (−) or centre the mosaic (home), as can be seen in Fig. 11c.

To participate in the mosaic users upload a photo, write a comment and press the corresponding icon in the control area on the screen. Once the photo has been uploaded, the mosaic shows a marker indicating to users where they have positioned their photo and on the interface this photo appears in miniature along with the comment (Fig. 11d).

By zooming over the mosaic users can enlarge their photo (Fig. 11e) and should they want to see it in full frame all they have to do is press the indicator on the mosaic (Fig. 11f). The application allows users to share their photo with their contacts. This is done by pressing the corresponding icon in the control area on the mosaic, and the window will unfold as in Fig. 11g) and users select the tool to share.

Through Whatsapp, Twitter and e-mail some messages will include a message by default (the user can edit this), as in an encrypted URL, which enables the receiver to see the mosaic and participate in it, giving the system the possibility of client acquisition without them physically needing to be present in the implementation area of the mosaic or touching the associated Tag. This URL is coded and controlled by the system to ensure unwanted usage.

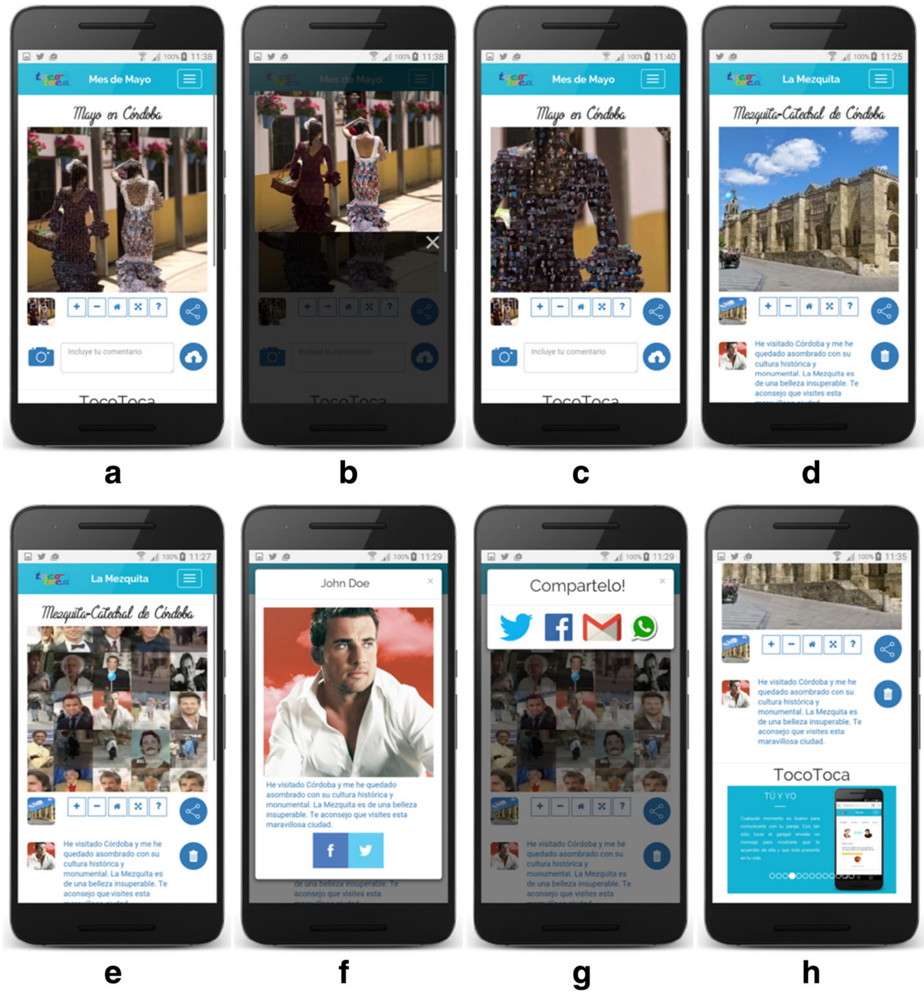

Fig. 11 Some snapshots of the application

Finally, as can be seen in Fig. 11h, the software solution developed allows for the possible inclusion of banners, which allow institutions or companies to use the system for publicity or client acquisition.

7.2 Database model

The domain of the information related to mosaics can be managed by a reduced number of tables as shown in the rectangular area of the related diagram Fig. 12.

The Mosaic table stores the related information to the background image, whose hierarchical structure of enlargement is stored in a structure of directories in which each directory is named with a zoom level. This table stores the necessary information for its management along with all the parameters required for the process of generating the mosaic: size of the background image, cell size, slot size, mosaic shape, blending factor and function, etc.

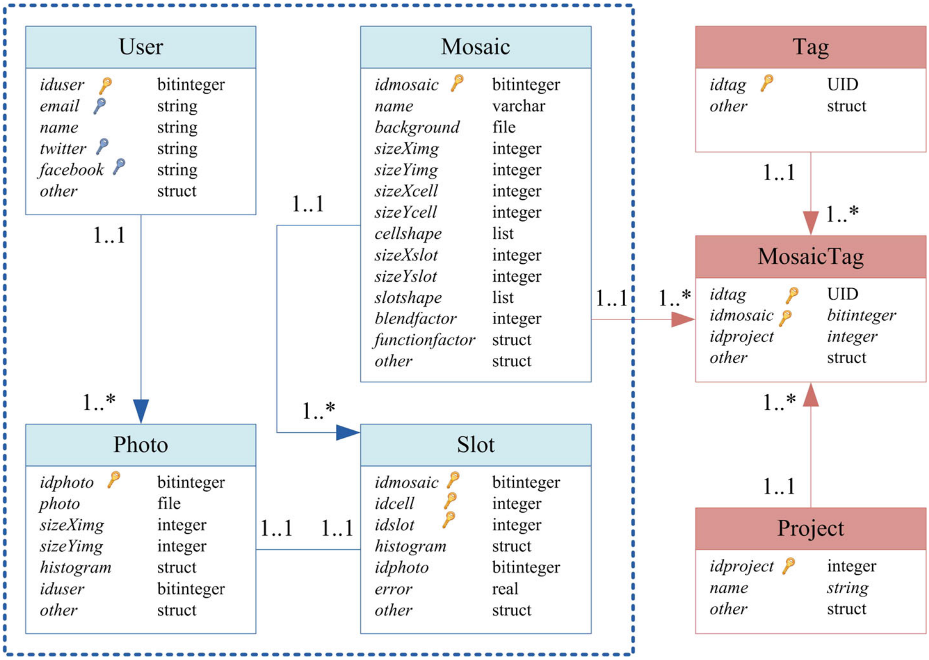

Fig. 12 Some of the main tables of the relational database model (only main attributes have been included)

The slot information is stored in the table *Slot*. Each slot is identified through the mosaic and mosaic cell where it is positioned, storing the pixel colour characteristics of the slot and, in its case, the users' photo it contains.

The table *Photo* manages all the photos that users associate with the mosaic. This table includes user information and the image and colour characteristics of the photo. Finally, the table *User* manages all the information about the users who have participated with their photos in the building of the mosaics.

For the deployment of a marketing project, a software solution has been developed which shall be explained in detail in the next article. This solution enables the defining of different marketing projects whose information is managed with a series of tables represented symbolically by the table *Project* in Fig. 12. These tables allow representation of any type of multimedia wished to be used in strategy or marketing campaigns, their scope and transient nature, associated functionality and permission to users, actions in social networks and electronic communication media (email, Whatsapp, social networks etc.).

The Tags associated with Smart Posters are clearly identified and managed in the table *Tag*. Each NFC Tag is uniquely factory-identified by a micro-chip that cannot be falsified, which in turn is used as an ID for this table.

Finally, the table *MosaicTag* allows a SmartPoster to associate one or more mosaics, which enables users to select the background image in which they can include their photo. One of the characteristics that makes our solution a viable potent marketing tool is that different mosaics associated with the same Tag can be associated with different projects or marketing campaigns, which transforms each mosaic into an individual means of promotion for each company participating in the project.

8 Method and results

In order to test the developed system two live experiences were carried out. For each experience a set of background images related to the experience subject were selected. Each background image was manipulated selecting the area corresponding to the mosaic for retaining the usable and not usable slots. Different tests were performed for each image background selecting the slots size and shape, colour characteristics, similarity metric and blending and border parameters.

Next, k-means clustering were performed for each image background and representatives or centroids were obtained. This information and the hierarchical structure of the image background were stored in the database.

Finally, the smart posters were designed and for each image background a NTag215 NFC chip was associated. Thus, when the users touch this Tag with their Smartphone the default browser installed in the device access to the URL associated with the functionality of corresponding mosaic.

The first of the live experience were carried out from the 10th–13th November 2016, the regional government of the province of Córdoba (Spain) organized the VII Municipal Fair, an event in which different villages and towns in the Córdoba province showed their products, culture, gastronomy and tourism offers in the area to promote and advertise its potential in tourism and business.

This event was in collaboration with the local authorities of the town of Santaella and which included two Smart Posters on its stand dedicated to the cultural promotion of agriculture, the environment and tourism respectively. Each Smart Poster was made up of 4 mosaics where users could interact and upload their photos.

Figure 13 shows some photos taken at this event in which, over three days, more than 831 people participated. The most attractive mosaics for the users were those related to the more significant monument of the town (Hermitage of Santaella) with 365 interactions and its agricultural activity with 200 interactions. However, due to the characteristics of the event, only relative data was obtained concerning the number of interactions and distribution in timetable slots.

For this reason, and with the aim to analyse the impact on the users of the solutions developed, a control test was undertaken on the university campus of Rabanales in Córdoba. On the university grounds they set up 4 Smart Posters and a totem as shown in Fig. 14.

The experience was named "Córdoba is a Mosaic", consisting of eight mosaics whose images represented eight characteristics or distinguishing elements about the city of Córdoba and with which users could identify themselves or wish to share about the city.

Fig. 13 Some pictures of the experience at the "VII Fair of the Towns"

Fig. 14 Some pictures of the experience at the Rabanales Campus

To do so, the Smart Posters and totem had representative images of the city for each of the following mottos: "Monumental Córdoba", Gastronomical Córdoba", "Córdoba in May", "Córdoba is Culture", "Córdoba is Passion", "Córdoba is a University City", "Córdoba is Different", "Córdoba is Much More".

Second year database students on their computer engineering course were invited to participate in the experience and complete a survey on their experience, for which they were given 48 h.

The survey was carried out anonymously via the Moodle system [50] available at Córdoba University. Of the 139 students enrolled on the course, only 89 took part in the survey (67%); they could not distinguish between men and women (although only 9 out of 132 students enrolled were women).

For the duration of the experience, 1874 mosaic interactions took place (an average of 21 interactions per person) that were distributed irregularly among the mosaics. As can be seen in Table 2, the associated image and motto influenced the users in their participation in the experience and being involved in uploading a photo.

Of the 89 users that were controlled in the experience, 67 shared some of the mosaics which they uploaded a photo of and shared with their contacts (75.3%).

Table 2 Results of the interactions with the Smart posters

Selected image	Corresponding lemma	% Interactions	% Uploaded Photo
Mosque-Cathedral of Cordoba	Monumental Córdoba	24,1	32,2
Cordovan Flamenquín	Gastronomical Córdoba	12,9	13,0
May fair	Córdoba in May	12,0	10,4
Alleyway of flowers	Córdoba is Culture	6,0	7,0
Córdoba Football Club	Córdoba is Passion	18,1	17,4
University of Córdoba	Córdoba is a University City	12,1	9,6
44 degree Thermometer	Córdoba is Different	8,0	4,3
Santaella town	Córdoba is Much More	6,8	6,1

The survey which was voluntarily and anonymously undertaken by students, was aimed at testing the application developed from the point of view of the end user, containing questions that were evaluated between [3, 7, 23, 47, 65], which had a corresponding value of Very Low to Very High.

Figure 15 shows the results of the survey. As can be appreciated, there are clear low value peaks in all the items corresponding to some students who tried to take advantage of the survey for uses other than the purpose of the survey.

In all cases, the surveys showed a positive response to the product and especially in its ease of use and the deployment in visualising processes and uploading of photos to the mosaic. The sensitively low value for the usability aspects was due to the fact that there were participants without NFC technology available on their mobile devices. Therefore, in each Smart Poster there was a QR code included which linked to a website in which the participant could select one of the mosaics, although once selected, this site could not be revisited. This fact had a negative influence on the survey results.

Until the end of the academic course (June 2017) the smart posters have been installed in the university location and the access to them has been opened to any visit. In this period, the mosaics received 6834 interactions. The most attractive mosaics for the users were those related to cultural subjects such as: Mosque of Córdoba ("Monumental Córdoba") with 2631 interactions, Fairs of Córdoba ("Córdoba in May") with 2370 interactions and, of course, the subject related with the football team of the city ("Córdoba is Passion") with 1027 interactions.

9 Conclusions

One of the multiple applications of photo mosaics is in its use in strategies for client acquisition and loyalty, as well as in marketing campaigns. Among the sectors in which it can be applied are retail and tourism, which in many countries makes up a large proportion of their GDP (Gross Domestic Product).

Tourists visit cities and their monuments and like to take photos and selfies appearing next to them. The inclusion of these photos in a mosaic of these places and monuments, the

Statistical	Usability	Simplicity	Easy use	Interface	Navigability	Help	Performance	Innovation
Average	3.38	4.29	4.11	3.63	3.88	3.67	4.10	3.85
STDEV	0.99	0.75	0.96	1.07	1.12	1.09	0.97	0.92

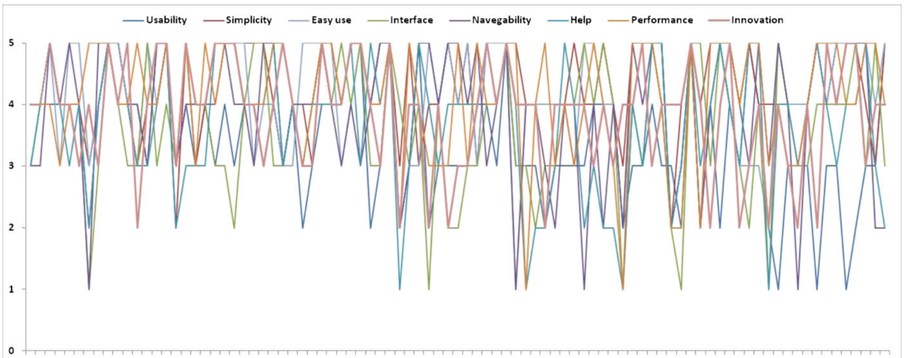

Fig. 15 Results of the survey

publishing of the mosaics in the Web of these cities or companies, and their link to social networks may result in an appropriate and effective marketing tool.

In this paper, a mobile solution has been presented that enables the easy application of marketing strategies based on the use of photo mosaics. The solution shown allows the building of very realistic mosaics in real time at a low computing cost and fast interaction for users. This means that it is possible to carry out through the use of mobile devices and low speed communication networks.

In order to do so, efficient algorithms, k-means clustering, usage of different similarity metrics and data structures have been used for the assignment of photos to slots in real time, which make up the mosaics. In addition, fine-tuning of the mosaics is performed in batch processing using bipartite graphs with Kuhn-Munkres algorithm allowing the reconstruction of large mosaics with a very reduced computational cost.

With the aim to improve the building and visualisation of mosaics, our solution allows the personalisation of a large number of parameters that are involved in the process. Thus, according to the characteristics of the photo/background, the objects it is composed of and the areas of the photo or background which should be filled to give it a more realistic appearance, one can select the similarity index, size and shape of the cell and slot, blending geometry and its characteristics, etc.

The use of NFC (near Field Communication) as a means of user interaction with the mosaics and a solution based on Web applications contribute to a value-added usability as well as enabling the acquisition of information related to such interactions. Finally, the solution enables users to share the experience with their contacts through social networks and other messenger services, all of which contribute to our proposal having added value for its use as a marketing and advertising tool, as has been demonstrated through experiments and experience.

References

1. A C# map tiles generator | estaun.net blog. Available online: http://www.estaun.net/blog/map-tiles-generator/. Accessed 2 Dec 2016
2. Adobe Photoshop CC. Available online: http://www.adobe.com/products/photoshop.html. Accessed 5 Dec 2016
3. Alamareen A, Al-Jarrah O, Aljarrah IA (2016) Image Mosaicing using binary edge detection algorithm in a cloud-computing environment. International Journal of Information Technology and Web Engineering (IJITWE) 11(3):1–14
4. Alkhathami M, Han F, Van Schyndel R (2014) A mosaic approach to touchless fingerprint image with multiple views. In: proceedings of the international conference on distributed smart cameras, Venezia Mestre, Italy. ACM, p 22
5. AndreaMosaic Home Page. Available online: http://www.andreaplanet.com/andreamosaic/. Accessed 3 Dec 2016
6. APP Helmond | The creative digital toolbox. Available online: view-source:https://www.mosaizer.com/. Accessed 4 Dec 2016
7. Azzari P, Di Stefano L, Mattoccia S (2008) An evaluation methodology for image Mosaicing algorithms. In: Blanc-Talon J, Bourennane S, Philips W, Popescu D, Scheunders P (eds) 10th international conference on advanced concepts for intelligent vision systems (ACIVS 2008), Juan-les-pins. France. Springer, Berlin Heidelberg, Berlin, Heidelberg, pp 89–100
8. Battiato S, Di Blasi G, Farinella GM, Gallo G (2006) A novel technique for opus vermiculatum mosaic rendering. In: the 14th international conference in Central Europe on computer graphics, visualization and computer Vision'2006. Plzen, Czech Republic, pp 1–6
9. Battiato S, Di Blasi G, Farinella GM, Gallo G (2006) A survey of digital mosaic techniques. In: Eurographics Italian chapter conference. Catania, Italy, pp 129–135

10. Battiato S, Di Blasi G, Farinella GM, Gallo G (2007) Digital mosaic frameworks-an overview. In: computer graphics forum. Vol 4. Wiley online library, pp 794-812
11. Battiato S, Milone A, Puglisi G (2012) Artificial mosaics with irregular tiles based on gradient vector flow. In: European conference on computer vision. Firence, Italy. Springer, pp 581–588
12. Brady M, Jung KK, Nguyen H, Raghavan R, Subramonian R (1993) The assignment problem on parallel architectures. Network Flows and Matching DIMACS 1:469–517
13. C++ − Photo Mosaic Algorithm. How to create a mosaic photo given the basic image and a list of tiles?. Available online: http://stackoverflow.com/questions/5478519/photo-mosaic-algorithm-how-to-create-a-mosaic-photo-given-the-basic-image-and-a. Accessed 3 Dec 2016
14. Cantone D, Ferro A, Pulvirenti A, Recupero DR, Shasha D (2005) Antipole tree indexing to support range search and k-nearest neighbor search in metric spaces. IEEE T Knowl Data En 17(4):535–550
15. Chi D, Li M, Zhao Y, Xu G, Liu W, Hu J (2012) Image-based mosaics: an variable construction method. In: 2012 fifth international symposium on computational intelligence and design. Zhejiang Sci-Tech University Hangzhou, China, pp 370–375
16. CMosaizer Lite | the fastest mosaics creator in the world. Available online: https://mosaizer.com/MosaizerLite/index.htm. Accessed 3 Dec 2016
17. Cui H, Zhang J, Cui C, Chen Q (2016) Solving large-scale assignment problems by Kuhn-Munkres algorithm. In: international conference on advances in mechanical engineering and industrial informatics. Hangzhou, Zhejiang, pp 822–827
18. Di Blasi G, Gallo G, Petralia MP (2005) Fast techniques for mosaic rendering. In: Computational aesthetics 2005: Eurographics workshop on computational aesthetics in graphics. Visualization and Imaging Girona, Spain. Citeseer, pp 29–39
19. Di Blasi G, Gallo G, Petralia M (2005) Puzzle image mosaic. Proc. IASTED/VIIP, In, pp 33–37
20. Dobashi Y, Haga T, Johan H, Nishita T (2002) A method for creating mosaic images using Voronoi diagrams. In: proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on computer animation. San Antonio, TX, USA, pp 341–348
21. Download SDK De Microsoft® Silverlight™ 3 from official Microsoft download center. Available online https://www.microsoft.com/es-es/download/details.aspx?id=16011. Accessed 2 Dec 2016
22. Download Deep Zoom Composer from Official Microsoft Download Center. Available online: https://www.microsoft.com/en-us/download/details.aspx?id=24819. Accessed 6 Dec 2016
23. Elber G, Wolberg G (2003) Rendering traditional mosaics. Vis Comput 19(1):67–78
24. Ficken FA (2015) The simplex method of linear programming. Courier Dover Publications, New York
25. FMEdda | Foto-Mosaik-Edda. Available online: http://www.fmedda.com/es/home. Accessed 4 Dec 2016
26. GitHub - codebox/mosaic: Python script for creating photomosaic images. Available online: https://github.com/codebox/mosaic. Accessed 3 Dec 2016
27. GitHub - jpgill86/photomosaic: An open-source photomosaic renderer. Available online: https://github.com/jpgill86/photomosaic. Accessed 6 Dec 2016
28. Go4Mosaic.com. Available online: http://www.go4mosaic.com/en/create.html. Accessed 10 Dec 2016
29. Golin MJ (2006) Bipartite matching and the hungarian method. Hong Kong University of Science and Technology. Available online, Course Notes http://www.cse.ust.hk/~golin/COMP572/Notes/Matching.pdf. Accessed 8 Dec 2016
30. Google Maps. Available online: https://developers.google.com/maps/. Accessed 3 Dec 2016
31. Hoff KE, Keyser J, Lin M, Manocha D, Culver T (1999) Fast computation of generalized Voronoi diagrams using graphics hardware. In: Proceedings of the 26th annual conference on computer graphics and interactive techniques. USA. ACM Press/Addison-Wesley Publishing Co., Los Angeles, CA, pp 277–286
32. ImageMosaic: Image Mosaic using C#. Available online: https://github.com/karthik20522/ImageMosaic. Accessed 6 Dec 2016
33. JImage Mosaic - The Free Java-Based Photomosaic Creation Utility. Available online: http://jimage-mosaic.sourceforge.net/about.php. Accessed 4 Dec 2016
34. Kang D, Seo S, Ryoo S, Yoon K (2013) A study on stackable mosaic generation for mobile devices. Multimed Tools Appl 63(1):145–159
35. Kärner M Process for creating photo mosaics. Available online http://kodu.ut.ee/~b04866/poster.pdf. Accessed 4 Dec 2016
36. Khizrai MSQ, Bodkhe S (2014) Image encryption using different techniques for high security transmission over a network. International Journal of Engineering Research and General Science 2(4):299–306
37. Kim J, Pellacini F (2002) Jigsaw image mosaics. ACM Trans Graph 21(3):657–664
38. Kim J, Pellacini F (2002) Jigsaw image mosaics. In: Proceedings of the 29th annual conference on Computer graphics and interactive techniques, San Antonio, TX, USA. ACM, 566633, pp 657–664
39. Kourouthanassis P, Boletsis C, Bardaki C, Chasanidou D (2015) Tourists responses to mobile augmented reality travel guides: the role of emotions on adoption behavior. Pervasive Mob Comput 18:71–87

40. Kovács P (2015) Minimum-cost flow algorithms: an experimental evaluation. Optim Method Softw 30(1): 94–127
41. Kuhn HW (1955) The Hungarian method for the assignment problem. Naval Research Logistics Quarterly 2(1–2):83–97
42. Laraqui A, Baataoui A, Saaidi A, Jarrar A, Masrar M, Satori K (2017) Image mosaicing using voronoi diagram. Multimed Tools Appl 76(6):8803–8829
43. Lawler EL (2001) Combinatorial optimization: networks and matroids. Courier Corporation, New York
44. Lee S-U (2013) Assignment problem algorithm based on the first selection method of the minimum cost. The Journal of The Institute of Internet, Broadcasting and Communication 13(5):163–171
45. Lee H-Y (2016) Automatic photomosaic algorithm through adaptive tiling and block matching. Multimed Tools Appl 75(24):1–17
46. Lee Y-L, Tsai W-H (2014) A new secure image transmission technique via secret-fragment-visible mosaic images by nearly reversible color transformations. IEEE Trans Circuits Syst Video Technol 24(4): 695–703
47. Li Z, Isler V (2016) Large scale image mosaic construction for agricultural applications. IEEE Robotics and Automation Letters 1(1):295–302
48. Maimon O, Rokach L (2005) Data mining and knowledge discovery handbook, vol 2. Springer, London
49. McGrath Foundation Mosaic. Available online: https://www.mcgrathfoundation.com.au/mosaic.aspx?close=y. Accessed 10 Dec 2016
50. Moodle - Open-source learning platform. Available online: https://moodle.org/. Accessed 23 Dec 2016
51. Mosaic 2.0.0 : Python Package Index. Available online: https://pypi.python.org/pypi/osaic/2.0.0. Accessed 3 Dec 2016
52. Mosaic Creator - professional mosaic software. Available online: http://www.aolej.com/mosaic. Accessed 3 Dec 2016
53. Mosaic Photo Creator-MosaicAce. Available online: https://play.google.com/store/apps/details?id=com.tecace.mosaicace&hl=en. Accessed 10 Dec 2016
54. Mosaic Wonder. Available online: http://download.cnet.com/Mosaic-Wonder/3000-2192_4-10915104.html. Accessed 4 Dec 2016
55. Munkres J (1957) Algorithms for the assignment and transportation problems. J Soc Ind Appl Math 5(1): 32–38
56. My Masterpiece. Available online: http://sharerefreshingideas.com/mymasterpiece/. Accessed 8 Dec 2016
57. Nock R, Nielsen F (2004) Statistical region merging. IEEE Trans Pattern Anal Mach 26(11): 1452–1458
58. OpenSeadragon. Available online: https://openseadragon.github.io/. Accessed 4 Dec 2016
59. Phithakkitnukoon S, Horanont T, Witayangkurn A, Siri R, Sekimoto Y, Shibasaki R (2015) Understanding tourist behavior using large-scale mobile sensing approach: a case study of mobile phone users in Japan. Pervasive Mob Comput 18:18–39
60. PhotoMontage AS (2000) Available online http://www.imaging-resource.com/SOFT/PMONTAGE/PM2K.HTM. Accessed 8 Dec 2016
61. Picture Mosaics - True Photo Mosaic Design. Simple. Free. Available online: https://picturemosaics.com/. Accessed 10 Dec 2016
62. Ranjan N, Soni BB, Shraman B (2015) An efficient technique for image Mosaicing using random sample consensus algorithm. Int J Comput Appl Technol 118(16):22–26
63. Reverland's Playground. Available online: http://reverland.org/python/2013/02/19/yet-another-photomosaic-generator/. Accessed 4 Dec 2016
64. Robert Silvers. Available online: http://www.photomosaic.com/. Accessed 8 Dec 2016
65. Singh R, Vatsa M, Ross A, Noore A (2007) A mosaicing scheme for pose-invariant face recognition. IEEE Trans Syst Man Cybern Part B-Cybern 37(5):1212–1225
66. Sullivan GJ, Ohm J, Han W-J, Wiegand T (2012) Overview of the high efficiency video coding (HEVC) standard. IEEE Trans Circuits Syst Video Technol 22(12):1649–1668
67. The Atlanta Falcons 'Rise Up' Mosaic. Available online: http://www.falconsriseup.com/demo.php. Accessed 12 Dec 2016
68. The best photo mosaics you will ever see. Available online: http://mosaikify.com/english/home. Accessed 4 Dec 2016
69. Uyttendaele M, Eden A (2001) Skeliski R eliminating ghosting and exposure artifacts in image mosaics. In: computer vision and pattern recognition, 2001. CVPR 2001. Proceedings of the 2001 I.E. computer society conference on. IEEE, pp II-509-II-516 vol. 502

70. Vintage - resources. Available online: http://vintage.winklerbros.net/facescrub.html. Accessed 23 Dec 2016
71. Yan C, Zhang Y, Xu J, Dai F, Zhang J, Dai Q, Wu F (2014) Efficient parallel framework for HEVC motion estimation on many-core processors. IEEE Trans Circuits Syst Video Technol 24(12):2077–2089
72. Zhou H, Zhou X, Gao H (2016) An image mosaic algorithm based on the combination of wavelet transformation and adaptive filtering. J Comput Theor Nanosci 13(1):588–592

Irene Luque Ruiz is full professor for Computer Sciences and Artificial Intelligent and she is member of the Department of Computing and Numeric Analysis at the University of Cordoba. Bachelor of Chemical Sciences in 1981 and Doctor Lecturer in the Computer Sciences in 1995, she is manager in the Software and Knowledge Engineering and Databases research group. Her teaching responsibilities are Software Engineering, Information Systems. Her research activities are oriented to new computational solutions in Chemistry, NFC Solutions and Mobile Computing.

Gonzalo Cerruela García is Associate Professor in the area of Computer Science and Artificial Intelligence and he is member of the Department of Computing and Numerical Analysis at the University of Córdoba. Bachelor of Electronic Engineering in 1989 and Ph.D. in Computer Science in 1999. His teaching responsibilities include the disciplines of Databases, Data Structures and Information, Web and Mobile Engeniering. His research interest focuses on Pervasive and Mobile Solutions and Information Systems.

Miguel Ángel Gómez-Nieto is full professor for Computer Sciences and Artificial Intelligent and he is member of the Department of Computing and Numeric Analysis at the University of Cordoba. Bachelor of Sciences in 1980 and Doctor Lecturer in the Chemical Sciences in 1982. His teaching responsibilities are Data Bases, Software Engineering and Computational Systems Analysis and Design. His research activities are oriented to new computational solutions in Chemistry, Computer Information Systems.