2012 International Conference on Indoor Positioning and Indoor Navigation, 13-15th November 2012

A multisensor LBS using SIFT-based 3D models

Antonio J. Ruiz-Ruiz , Pedro E. Lopez-de-Teruel , Oscar Canovas
Department of Computer Engineering
University of Murcia, Spain
Email: {antonioruiz,pedroe,ocanovas}@um.es

***Abstract*—This paper introduces an LBS multisensor system that acquires data from different sensors available in commodity smartphones to provide accurate location estimations. Our approach is based on the use of visual structure from motion techniques to run off-line 3D reconstructions of the environment from the correspondences among the SIFT descriptors of the training images. We present several solutions to reduce the deployment cost, in terms of time, and to minimize the interference degree within the environment, but also pursuing a good balance between accuracy and performance. To determine the position of the smartphones, we first obtain a coarse-grained estimation based on WiFi signals, digital compasses, and built-in accelerometers, making use of fingerprinting methods, probabilistic techniques, and motion estimators. Then, using images captured by the camera, we perform a matching process to determine correspondences between 2D pixels and model 3D points, but only analyzing a subset of the 3D model delimited by the coarse-grained estimation. We implement a resection process providing high localization accuracy when the camera has been previously calibrated, that is, we know intrinsic parameters like focal length, but it is also accurate if an auto-calibration process is required. Furthermore, our experimental tests show promising results, since we are able to provide high accuracy with an average error down to 15 cm in less than 0.5 seconds of response time, making this proposal suitable for applications combining location-services and augmented reality.**

***Index Terms*—Image processing, multisensor, training, SIFT, structure from motion, smartphones**

I. Introduction

For a diverse set of areas including tracking, geographic routing or entertainment, location-sensing systems are currently a very active research field. The wide adoption of smartphones enables new scenarios to offer context-tailored information and services to an increasing number of users. Smartphones are equipped with several sensors, which has simplified the process of obtaining the required context information. In this work we will focus on indoor environments, but our approach is also valid for outdoors.

Nowadays, there are several widespread applications that make use of the images that users obtain from the camera in order to display augmented reality. Indeed, those images can be used to provide a better estimation of the users' position, offering a seamless integration of the sensor and the display. Though accuracy obtained using RSSI-based techniques is good enough for some applications scenarios, our main contribution in this work is the development of an LBS suitable for precise augmented reality services, which require not only very fine-grained position estimations (up to a few cm of maximum error), but also an equally precise estimation of the absolute orientation of the imaging device. Therefore, hereinafter we will assume that the users are capturing images, typically with a tablet or a smartphone, for which they want to obtain the precise estimation of the full 6 degrees of freedom (*dof*) –3 for (X, Y, Z) position and 3 additional for the (α, β, γ) Euler angles for the rotation matrix– of the device with respect to some conveniently chosen world reference coordinate system.

In order to achieve that goal, a keystone in our approach is the use of the *Scale Invariant Feature Transform* (SIFT) [20], a well known technique in computer vision research. Our proposal is also based on the use of structure from motion (*SfM*) techniques to run off-line 3D maps reconstructions of the environment from the features extracted using the training images captured along the area of interest. Finally, several resection techniques have also been evaluated to estimate the full 6 *dof*, depending on whether we know some intrinsic camera parameters, like focal length.

Our multisensor approach makes use of additional sensors data, like RSSIs and measurements from the digital compass or built-in accelerometer, in order to limit the amount of information to be examined, to simplify the training process and to reduce the computational load of the smartphones. For many indoor location-based services, the training phase requires an important time effort, since they are based on a laborious work of collecting RSSI and images at each discrete point where the mobile devices might be located. However, our solution follows a different approach based on a continuous video recording and data acquisition process. We have also distributed the required functionality between the mobile devices and dedicated servers. The system can currently complete an entire cycle, from obtaining data from sensors to giving a precise estimation, in less than 400 ms.

Several augmented-reality applications do not require a continuous real time response, such as those designed to display location-aware reminders or notes, to obtain information about who is behind the door of a particular laboratory, or to visualize additional objects of interest virtually added to the scene. Our system obtains a good trade-off between a fine-grained accuracy in location and an acceptable response time, making it ideal for such kind of applications.

II. Related work

Indoor positioning is an active research field. Many types of signals (radio, images, sound) and methods have been used to infer location.

Paying attention to radio signals, pattern recognition methods, among many others, estimate locations by recognizing position-related patterns. Fingerprinting [5] is based on radio maps containing patterns of RSSIs, which are obtained using 802.11, Zigbee, Bluetooth or any other widespread wireless technology. The analysis of RSSI patterns is a technique that has been examined by several authors [4], [13], obtaining an accuracy ranging from 0.5 to 3 meters.

However, better results can be obtained by integrating the information captured by multiple sensors. For example, SurroundSense [3] is a mobile phone-based location system for indoor environments via ambience fingerprinting. However, the optical recognition techniques proposed in SurroundSense are simply based on images of the floor to extract information about light and color. With SIFT, our work avoids typical variations related to light, color or scale, and provides much more robustness and better accuracy, by exploiting the superior discriminative power of the extracted visual features.

Other works also use WiFi signals and image recognition to estimate positions, such as [15] and [22]. However, they are based on two-dimensional landmarks that must be placed in the scenario of interest, which involves the inclusion of obtrusive elements.

In [1], [28], the authors performed a preliminary analysis of how techniques like SIFT can be used to improve the accuracy in location based systems. As we will show, using our 3D model we are able to provide a higher accuracy and to reduce the required time for the training process.

III. System overview

This section depicts the design of our proposal, covering the training phase and the on-line phase. Figure 1 provides a general overview of the different operations.

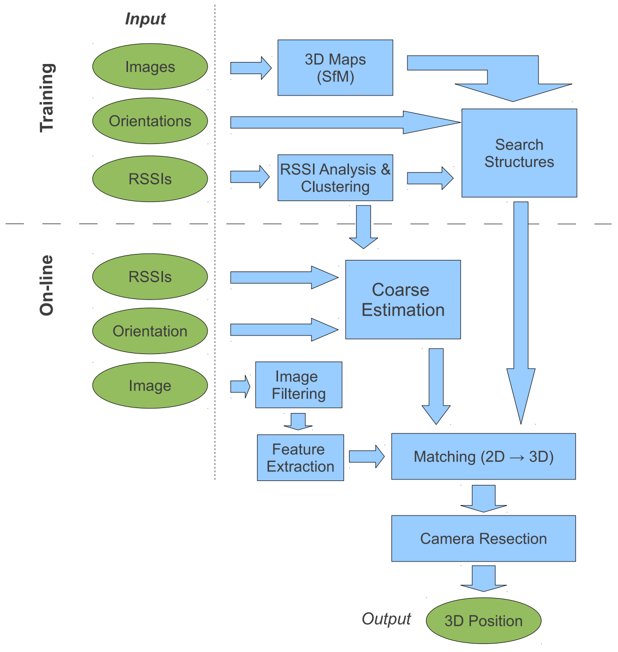

Fig. 1. System overview

During an initial training phase, images, 802.11 signals and orientations are captured along the entire scenario. Furthermore, we have considerably reduced the training time since images are captured as part of a video recording. Then, using SIFT features extracted from those images, and visual structure from motion techniques, we generate 3D maps of the scene ready to be used for full localization (this process is detailed in Section VIII-B). 802.11 signals are also analyzed to obtain probability distributions of the RSSIs read at different reference positions.

During the on-line phase, multiple sensors will be used to determine an initial coarse estimation that indicates a cluster of physical points where the device seems to be. To take advantage of this first estimation, we previously divided the 3D model during the training phase according to both the geographical areas where signals showed a similar behavior and to the different meaningful orientations.

Next, making use of the image captured by the smartphone, we perform a filtering process based on a gradient model to discard blurry and uniform images. Then, if the image is valid, we extract its SIFT features and perform a matching process against the features contained in our partitioned 3D model to determine correspondences between 2D pixels and 3D points. In Section VI we will describe the different approaches we have analyzed to perform this matching process.

Finally, and relying upon computer vision techniques, we perform a complete camera resection process to estimate the current 3D position of the device. A better accuracy is achieved when cameras have been previously calibrated, that is, we know intrinsic parameters (like focal length), but our technique also works accurately when it is necessary to carry out a camera auto-calibration process. Section VII describes this camera resection process and includes a detailed analysis of the different parameters that we have considered.

IV. A multisensor approach

In recent years, the wide adoption of smartphones equipped with several sensors has simplified the process of obtaining context information. Despite we use the images from the camera as the main piece of data to infer accurate positions, we can benefit from other sensors in order to limit the amount of information to be examined, to simplify the training process or to reduce the computational load of the smartphones. We believe this multisensor approach is crucial to accomplish a feasible image-based location based service.

On the one hand, in relation to the generation of search structures and context models, the main drawback of most of the indoor positioning systems is the scalability issues that arise when modeling large scenarios. However, several solutions can be envisioned following a multisensor approach in order to reduce the necessary time effort and the required processing during the on-line phase. For example, many previous works have demonstrated that significant accuracy can be obtained in indoor scenarios by means of fingerprinting techniques [5]. Using some of those techniques, during the training phase we will be able to divide the scenario into different zones according to the characterization of the 802.11 signals. We can obtain clusters of physical points [19] where signals show a similar behavior, and they are used to partition the 3D model into different submodels. We have confirmed in previous works [24] that a cluster hit percentage of 90% can be obtained using probabilistic techniques, and in 5% of the remaining cases we obtain positions located in an adjacent cluster. Additionally, while we are recording the training video, we also save information related to the current orientation of the training device using the data provided by the built-in digital compass, since the different video frames will be tagged with their corresponding orientation.

During the on-line phase, we will determine a coarse-grained geographical zone where the device is located using the obtained RSSI. That zone is related to different 3D submodels, and one of them will be selected according to the orientation of the device in order to find correspondences between the features of the current image and those contained in the submodel. In this way we reduce the amount of information (image features) to check against during the on-line phase, which improves performance.

On the other hand, the information from the different sensors will also be used to reduce the computational cost on the smartphone, to minimize the amount of information transmitted and, therefore, to preserve the smartphone battery life. For example, we have developed a motion estimator based on accelerometer readings to determine whether the user location has to be recalculated because the user is moving. Access points are also analyzed locally in order to determine whether we are in a location supported by our system. Images are also preprocessed locally to check how blurry they are.

V. Using SIFT-based 3D Models

As we mentioned above, a keystone in our approach is the use SIFT, a well known and widely adopted technique in computer vision research. It provides a method for extracting a collection of *visual features* from images, which are invariant to image translation, scaling and in-plane rotation, and partially invariant to illumination changes and affine distortion suffered in different imaging conditions and/or varying camera viewpoints. These features are fairly repetitively detected, while being also well localized in both the spatial and scale domains of the input images. Moreover, a large number of them can be obtained for every image (typically a few hundreds, depending on several factors such as the image size, blurring and, of course, the scene itself). The strictly local nature of the computation for each feature, together with the possibility of globally computing consistencies among the whole list of obtained features, make this technique rather robust to problems like occlusion, clutter, or moderate image noise.

Each obtained SIFT feature is characterized by a 4-D *position vector* (x, y, σ, θ), where (x, y) is the position in the image, σ is the feature scale (i.e. size), and θ is a dominant orientation. Additionally, each feature is also attached a highly distinctive 128 dimensional *description vector*, which depends on the overall photometric structure of the local image environment of the feature. This descriptor allows a single feature to be correctly matched with a high probability against a large database of features, providing a powerful basis not only for visual recognition of objects or places [8], but also –and what is more relevant for this work– in the context of simultaneous localization and three-dimensional scene reconstruction [26].

We have already used this technique in [24], performing a matching process between the features of the image obtained by the device to be localized and a geo-tagged database of features that were previously extracted using representative images taken in a discretized grid of our application environment. These images were captured in all possible directions of movement to estimate the device location in an orientation-independent way. The system was able to approximate fairly well the current location if a good amount of matchings with some image in the database was found.

Nevertheless, while this technique provided acceptable results in a moderate response time (3 seconds), it suffered from three main drawbacks: first, the obtained accuracy depended also on the granularity defined when the environment was modeled as a grid of discrete points; second, this *matching only* approach provided approximate 3D position of the sensor, but no attempt to precisely estimate the orientation was made in that work (beyond the orientation provided by the original training images); and third, but not less, the training phase made it necessary to capture several images at each one of the points in a discretized grid of our environment, a rather cumbersome process which could become unfeasible in terms of time effort and needed human resources when modeling large scenarios such as a university campus or an airport.

To overcome these issues, in this work we combine the use of SIFT features with state-of-the-art techniques in three dimensional computer vision, which are able to reconstruct a real world scene given just a set of images of it. Taking advantage of the remarkable advances in the field of applied projective geometry in the last decade, nowadays it is possible to obtain an accurate 3D model of an arbitrary scene using only a sufficiently large number of images taken from different viewpoints of it. This can be done even in conditions such as the completely uncalibrated case, in which neither the real position nor the internal parameters of the cameras are known in advance. The myriad of techniques involved in this complex process, generically known in the computer vision literature as *Structure from Motion* (SfM), go from feature extraction and matching to estimation of the geometric relationships between the multiple views of the scene, passing through multiple intermediate matrix algebra procedures and both linear and non-linear optimization techniques. Each of these computing stages have themselves generated an enormous amount of research in the last decade, so their thorough descriptions are out of the scope of this paper, but the remarkable book [14] –the *de-facto bible* in the field– is a good reference for those interested in the underlying details of this research field.

In any case, and just to get an idea of the functioning and power of such techniques, we briefly summarize the overall

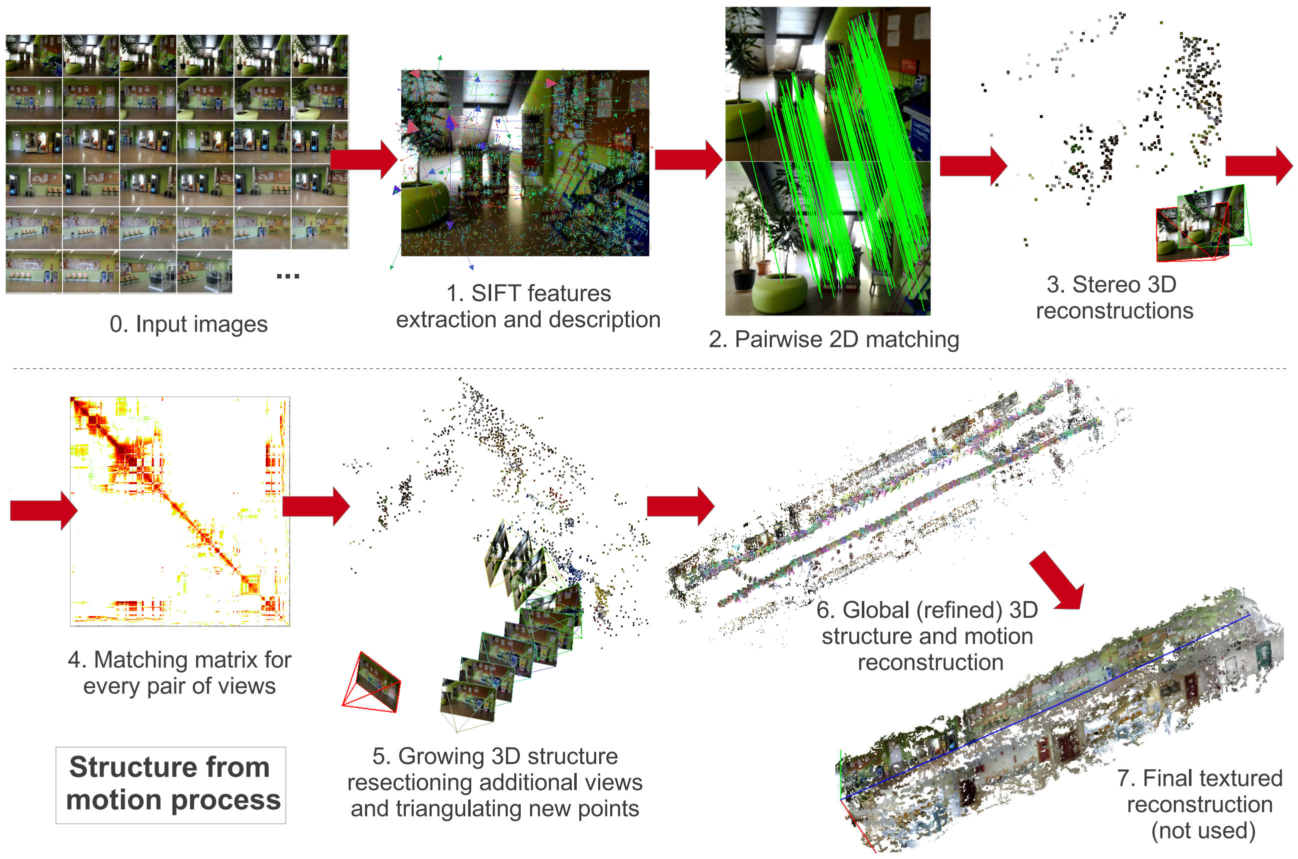
0. Input images
1. SIFT features extraction and description
2. Pairwise 2D matching
3. Stereo 3D reconstructions
4. Matching matrix for every pair of views
Structure from motion process
5. Growing 3D structure resectioning additional views and triangulating new points
6. Global (refined) 3D structure and motion reconstruction
7. Final textured reconstruction (not used)

Fig. 2. Illustration of the *SfM* 3D reconstruction technique. All figures obtained using Wu's VisualSfM software on our test environment (see also sec. VIII).

process followed by a typical *SfM* system in Figure 2. We use here, as the illustrating example, the 3D modeling process that we performed in our own test environment (though more details on the whole procedure will be given in Section VIII-B). During the training phase, a video is recorded capturing all the details of our environment. The video is then processed in order to extract images at a particular frame rate, which will be used as input for the SfM process (step 0 in figure 2). The use of video recording not only saves a lot of time and effort during training, but it also guarantees better 3D reconstructions, since overlapping images and a sequential order facilitates the posterior matching and reconstruction stages. Taking as input only that set of images, and by extracting (step 1) and matching (step 2) the SIFT features among them, the aforementioned SfM techniques are able to finely estimate the 3D position of visually relevant points in the scene, as well as the position and orientation of the cameras from where the pictures were taken. In order to do that, the procedure starts from individual stereo pairs (step 3), which are augmented later with new cameras by an incremental resectioning/triangulation procedure (step 5). This incremental process is guided by the global set of pairwise matchings matrix found among all the images (step 4). The final step in every SfM process, known as *bundle adjustment* [27], is a global nonlinear optimization stage which minimizes the reprojection error of all the reconstructed 3D points to the 2D image position of the original SIFT features in the images, thus producing an accurate 3D reconstruction which includes both the 3D points and the full 6 *dof* position of the cameras (step 6). Though not shown in the figure, each 3D point is attached the set of 128-dimensional descriptors as extracted from the corresponding images where that point appeared in the field of view, something that will be key for our posterior localization process. Moreover, though not strictly needed in our system, the set of points can be augmented with texture patches to get a more realistic 3D reconstruction (step 7). As we can see, the obtained reconstructions provide a fairly accurate metric reconstruction of the environment, up to an arbitrary global euclidean transform, which is conveniently defined choosing an adequate world reference coordinate system (red-green-blue axis in the figure).

VI. Matching

Once we have obtained the 3D representation of our scenario, we need a mechanism to look for coincidences between the features extracted from the 2D images captured during the on-line phase and those features that made up the 3D model. In order to perform that matching process efficiently, we considered different search structures that might be suitable to represent the 3D model, that is, the set of typically several thousands of 3D points with corresponding hundreds of thousands of features extracted from the training images. Since each feature is defined as a 128-dimensional vector, there is no algorithm able to identify the exact nearest neighbors in such a high dimensional space that is more efficient than exhaustive search. An alternative is to consider greedy algorithms which, though can be much faster in very large datasets (by using precomputed algorithmic structures, typically search trees), do obtain only acceptable approximations to the set of *k* nearest neighbors for each input feature.

In this work we have analyzed two different algorithms to match features, both based on the *k-nearest neighbor* technique. In all cases we have also partitioned our space according to the zones determined by the analysis of the RSSIs signals and meaningful orientations, as we described in section IV. The first option is based on the proposal made by Arya and Mount [2], which uses *kd-trees* and *bd-trees* supporting both exact and approximate nearest neighbor searching in spaces of various dimensions. The kd-tree version implemented by the *approximate nearest neighbor* (ANN) library [21] was exhaustively tested in [24], where we provided an analysis of the different parameters that impact on the speed and reliability of the obtained results. Those parameters are the distance ratio R (between the closest and the next closest neighbor) to discard false matchings, and the error bound ϵ that controls the maximum ratio between the distance to the reported point and the true nearest neighbor for approximate searches.

Although these implementations have been successfully tested in many applications where the database of features was huge (up to millions of them), the moderately large size of a typical individual zone in our case led us to test a brute force alternative: since our complete 3D model is made up of $\sim$ 250K descriptors in a real 128-dimensional space, the tree structures (one per each defined zone) used by ANN did not perform as fast in terms of response time as an exhaustive *exact nearest neighbor* (ENN) implementation of the brute force search algorithm running on a graphics processing unit (GPU). To take advantage of the enormous power of these relatively inexpensive computing platforms, we slightly modified the algorithm proposed in [9] in order to adapt its functionality to our zone-based space search. In addition to a better accuracy, the use of this algorithm resulted in an increased matching performance of up to 70% in terms of response time with respect to the original ANN implementation. Section VIII-C will show additional quantitative and qualitative information about our matching process.

VII. Camera Resection

Camera *resection* (sometimes generically referred to as camera *calibration*) is the process by which we can determine the $\mathtt{P}_{3\times 4}$ *projection matrix* from a given set of matching 3D scene points and the corresponding 2D pixels in an image. This matrix defines the algebraic relationship $\mathtt{P}\boldsymbol{X}_i = \boldsymbol{x}_i$ for every $\boldsymbol{X}_i \leftrightarrow \boldsymbol{x}_i$ correspondence when both the pixels $\boldsymbol{x}_i$ and the 3D points $\boldsymbol{X}_i$ are given in homogeneous coordinates [14]. $\mathtt{P}$ can be factorized as $\mathtt{P} = \mathtt{K}_{3\times 3}\mathtt{R}_{3\times 3}[\mathtt{I}|-C]_{3\times 4}$, in a way that the internally encoded relationship with the rotation $\mathtt{R}$ and 3D position $C = (c_x, c_y, c_z)^\top$ of the camera in the reference 3D coordinate system is made explicit. This factorization depends as well on the camera intrinsic calibration matrix $\mathtt{K}$, which contains internal camera parameters such as the focal length and the aspect ratio (for example, a very simple calibration matrix which is commonly used is $\mathtt{K} = diag(f, f, 1)$, with f the focal length in pixel units). In this section we detail the entire resectioning process that we follow in this work, and which is summarized in Figure 3.

```
Input: $2D \leftrightarrow 3D$ correspondences
1: $\mathtt{P}, inliers \leftarrow DLT(correspondences)$          ▷ RANSAC
2: if camera intrinsic known then          ▷ Precalibrated case
3:     $\mathtt{K}, \mathtt{R}, C \leftarrow ExteriorOrientation(\mathtt{K_{fixed}}, inliers)$
4: else          ▷ Autocalibration
5:     $\mathtt{K}, \mathtt{R}, C \leftarrow QR_{Decomposition}(P)$
6: end if
7: $\mathtt{R_{opt}}, C_{opt} \leftarrow NonLinearOptimization(\mathtt{K}, \mathtt{R}, C)$
Output: $\mathtt{R_{opt}}$ and $C_{opt}$ (camera pose in world coordinates)
```

Fig. 3. Resection algorithm

The overall process starts by running a robust implementation of the so called *Direct Linear Transformation* (DLT) algorithm [14], which allows us to obtain the camera matrix $\mathtt{P}$ that minimizes the reprojection error for the maximum number of *inlier* correspondences. In this context, inliers are matching pairs of 3D$\leftrightarrow$2D points which can be fitted to the model up to a given error tolerance. Then, and if the camera had been precalibrated (that is, we knew in advance the calibration matrix $\mathtt{K}$), we would discard $\mathtt{P}$, using only the obtained inlier set to get the desired camera pose $(\mathtt{R}, C)$ using an *exterior orientation* algorithm. Otherwise, we still can perform a camera *autocalibration*, using the previously estimated $\mathtt{P}$ to solve the factorization problem by carrying out the simple and well known matrix algebra operation of QR decomposition [10]. These two alternatives give us the possibility to treat both the cases of knowing the camera internal parameters in advance (feasible for most common devices), as well as the completely uncalibrated case, which is solved at the price of a slightly lesser precision. The final pose is obtained by refining the former solution using some standard non-linear optimization procedure [23].

A. Direct Linear Transformation

The $\mathtt{P}$ matrix that we must first estimate is an homogeneous entity with 12 elements, so that it has in general 11 *dof*. Each 3D$\leftrightarrow$2D correspondence $(x, y, z) \leftrightarrow (p, q)$ sets up the homogeneous restriction $\mathtt{P}(x, y, z, 1)^\top = \lambda(p, q, 1)^\top$, which is equivalent to $(p, q, 1) \times \mathtt{P}(x, y, z, 1) = (0, 0, 0)$. Namely,

$$\begin{bmatrix} \mathbf{0}^\top & -\boldsymbol{X}_i^\top & q_i\boldsymbol{X}_i^\top \\ \boldsymbol{X}_i^\top & \mathbf{0}^\top & -p_i\boldsymbol{X}_i^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{P}^1 \\ \boldsymbol{P}^2 \\ \boldsymbol{P}^3 \end{bmatrix} = \mathbf{0} \qquad (1)$$

where $\boldsymbol{X}_i = (x_i, y_i, z_i, 1)^\top$ and $\boldsymbol{x}_i = (p_i, q_i, 1)^\top$ $\forall i$ are the homogeneous coordinates of the corresponding pair of points, and $\boldsymbol{P}^k = (p_{4k-3}, p_{4k-2}, p_{4k-1}, p_{4k})^\top$ for $k = 1, 2, 3$, are the three rows of $\mathtt{P}$. Taking into account that we must solve for 11 unknowns (*dof* of $\mathtt{P}$), and since each $2D \leftrightarrow 3D$ correspondence gives rise to two independent equations over the elements of P, it is necessary to make use of a minimum of 6 of these correspondences to solve for the complete system. From the solution for the 12 unknowns $p_1 \cdots p_{12}$ we can easily reconstruct a canonical version of the homogeneous matrix $\mathtt{P}$:

$$\mathtt{P} = \begin{bmatrix} \frac{p_1}{p_{12}} & \frac{p_2}{p_{12}} & \frac{p_3}{p_{12}} & \frac{p_4}{p_{12}} \\ \frac{p_5}{p_{12}} & \frac{p_6}{p_{12}} & \frac{p_7}{p_{12}} & \frac{p_8}{p_{12}} \\ \frac{p_9}{p_{12}} & \frac{p_{10}}{p_{12}} & \frac{p_{11}}{p_{12}} & 1 \end{bmatrix} \tag{2}$$

The former procedure can be easily adapted to the overdetermined case (more than 6 correspondences). Since points are measured inexactly due to noise, some of these correspondences may not be fully compatible with any projective transformation, and therefore the best possible $\mathtt{P}$ has to be determined, which is found by minimizing the reprojection error in the minimum square sense.

The SIFT matching process frequently "contaminates" the set of real correspondences with some subset of *outliers* (wrong correspondences), which would seriously distort the obtained solution if not adequately filtered. To solve this problem, a robust estimation procedure is needed, and in this work we use the well known RANSAC algorithm [7]. This algorithm works by finding the best transformation matrix that minimizes the reprojection error taking into account only inlier (i.e. correct) correspondences. To filter the outlier (incorrect) correspondences, RANSAC iteratively selects a random subset of six elements from the original set of correspondences, from which it computes a tentative $\mathtt{P}$. Since the number of available correspondences is notably higher than this required minimum of six, at each RANSAC iteration we perform an incremental process that iteratively adds the correspondences identified as inliers and recalculates P until no more inliers are detected. One correspondence is identified as an inlier if its reprojection error does not exceed a specified threshold. This iterative procedure, which will try several tentative $\mathtt{P}$ matrices before success, finishes when either it obtains a sufficient percentage of inliers over the total number of input correspondences, or the number of iterations exceeds a given threshold. In Section VIII-C we will show the influence of all these parameters in both the accuracy and performance of the results.

The $\mathtt{K}$, $\mathtt{R}$ and C values can be easily obtained from the QR decomposition of the first three columns of $\mathtt{P}$. Therefore, at least theoretically, this generic procedure could be applied without a previous knowledge of the calibration matrix of the camera. However, the main problem with this approach is that the solution values that we obtain (mainly $\mathtt{K}$ and C) can be severely coupled, that is, are not mutually independent from each other. Particularly, in some ill-conditioned cases such as scenes with a dominant plane (a not so uncommon case in typical indoor scenarios, where large walls can cover most of the image), the autocalibration process can fail to give an accurate solution. As Section VIII-C shows, the QR decomposition can still be a good alternative when the camera intrinsic parameters are completely unknown, except under the aforementioned ill-posed conditions. In the following section we show how this a priori knowledge of calibration largely improves the quality of the results.

B. Exterior Orientation

In many cases, the on-board camera specifications (mainly focal length and aspect ratio) of typical smartphones and tablets are known, so the calibration matrix $\mathtt{K}_{\mathtt{fixed}}$ can be determined. Fiore [6] developed an alternative linear method to obtain only the exterior orientation of the camera, which, using this knowledge of the intrinsic parameters, results in a much more precise estimation of $\mathtt{R}$ and C. Moreover, it performs remarkably well even in the planar case which poorly conditioned the problem for the DLT algorithm, as commented in the previous subsection. Fiore's algorithm works in two stages, first estimating the unknown depths ζ_i for each homogeneous 2D point $\boldsymbol{x}_i$, i.e., first we get an intermediate set of (inhomogenous) 3D points $\zeta_i \mathtt{K}_{\mathtt{fixed}}{}^{-1}\boldsymbol{x}_i$, and then what is left is an *absolute 3D orientation* problem, whose solution yields the desired values of $\mathtt{R}$ and C.

More specifically, given a number of input 2D$\leftrightarrow$3D point correspondences, $\boldsymbol{x}_i \leftrightarrow \boldsymbol{X}_i$, and the intrinsic camera matrix $\mathtt{K} = \mathtt{K}_{\mathtt{fixed}}$, we are required to find a rotation $\mathtt{R}$, a vector C (attitude and position of the camera) and a subsidiary depths vector $\boldsymbol{\zeta} = (\zeta_1, \cdots, \zeta_i, \cdots, \zeta_n)^\top$ (where n is the number of correspondences) such that:

$$\zeta_i \boldsymbol{x}_i = \mathtt{KR}[\mathtt{I}| - C]\boldsymbol{X}_i \Leftrightarrow \mathtt{K}^{-1}\zeta_i \boldsymbol{x}_i = \mathtt{R}[\mathtt{I}| - C]\boldsymbol{X}_i \quad \forall i \tag{3}$$

In order to solve for the unknown values of ζ_i, we first compose the matrix $\mathtt{M}$, whose columns are formed with the homogeneous coordinates of the 3D points,

$$\mathtt{M}_{4\times n} = [\boldsymbol{X}_1 \cdots \boldsymbol{X}_n] \tag{4}$$

and another matrix $\mathtt{N}$, arranging in this case the homogeneous coordinates $\boldsymbol{x}_i = (p_i, q_i, 1)^\top$ of the 2D points in the following way:

$$\mathtt{N}_{3n\times n} = \begin{bmatrix} \boldsymbol{x}_1 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{x}_2 & \dots & \vdots \\ \vdots & & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \dots & \boldsymbol{x}_n \end{bmatrix} \tag{5}$$

Then, taking the *singular value decomposition* (SVD) [10] of $\mathtt{M}$, we obtain the factorization $\mathtt{M} = \mathtt{U}_{4\times 4}\mathtt{D}_{4\times n}\mathtt{V}^T_{n\times n}$, where $\mathtt{U}$ and $\mathtt{V}$ are orthogonal and $\mathtt{D}$ is diagonal. Being r the rank of $\mathtt{M}$, we denote by $\mathtt{V}_2$ the submatrix of size $n \times (n-r)$ which results from taking only the last $n-r$ columns of $\mathtt{V}$. This submatrix spans the null space of $\mathtt{M}$, so that $\mathtt{MV}_2 = \mathbf{0}$.

Given the former definitions, it can be shown [6] that the depth vector $\boldsymbol{\zeta}$ can be recovered linearly (up to a scale factor), just by solving the following null-space problem:

$$((\mathtt{V}_2{}^\top \otimes \mathtt{K}^{-1})\mathtt{N})\boldsymbol{\zeta} = \mathbf{0} \tag{6}$$

Where the $\otimes$ symbol stands for the Kronecker product [16] of matrices. Once the depths have been obtained, our initial problem (3) is now reduced to find the correct alignment of two sets of 3D points, an instance of the well known absolute 3D orientation problem.

This last stage of the exterior orientation algorithm proceeds as follows. Let W_i be the 3D depth recovered points, $W_i = (\zeta_i \cdot (x_i/f), \zeta_i \cdot (y_i/f), \zeta_i)^\top$ (where we have used the $\mathtt{K}_{\mathtt{fixed}} = diag(f, f, 1)$ assumption), and Y_i the likewise inhomogeneous original 3D points, $Y_i = (x_i, y_i, z_i)^\top$. Then, given these two sets of 3-D points W_i and Y_i, we are required to find the rotation matrix $\mathtt{R}$, the vector C and the scalar s, such that:

$$W_i = s(\mathtt{R}Y_i + C) \quad \forall i = 1...n \tag{7}$$

Now, centering the point clouds on their respective means, i.e., $\overline{W}_i = W_i - \frac{1}{n}\sum_{j=1}^{n} W_j$ and $\overline{Y}_i = Y_i - \frac{1}{n}\sum_{j=1}^{n} Y_j$, the scale s is very easy to obtain as $s = \frac{||\overline{W}_i||}{||\overline{Y}_i||}$, for every i. In practice, s can be averaged for the set of points $\forall i = 1 \dots n$, and then used to scale accordingly the values of $\overline{Y}_i$.

With the sets of points adequately scaled and centered, we are left with the problem of estimating the unknown rotation between $\overline{W}_i$ and $s\overline{Y}_i$. This is known as the *orthogonal Procrustes* problem [12]. Being $\mathtt{W}_{3\times n}$ and $\mathtt{Y}_{3\times n}$ the matrices formed by stacking the points $\overline{W}^i$ and $s\overline{Y}^i$ respectively, the solution is found using again the SVD decomposition. The sought rotation is given by:

$$\mathtt{R} = \mathtt{V}' \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & Det(\mathtt{U}'\mathtt{V}'^\top) \end{bmatrix} \mathtt{U}'^\top \tag{8}$$

where $\mathtt{U}'_{3\times3}\mathtt{D}'_{3\times3}\mathtt{V}'^{\top}_{3\times3}$ stands now for the SVD of the matrix $\mathtt{YW}^\top$, and the determinant of $\mathtt{U}'\mathtt{V}'^\top$ is always 1 or -1.

Finally, once we have calculated the rotation matrix $\mathtt{R}$, we can obtain the translation vector C from (7) simply this way:

$$C = \frac{1}{s}\left(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{W}_i\right) - \mathtt{R}\left(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{Y}_i\right) \tag{9}$$

As we will demonstrate in Section VIII-C, this precalibrated exterior orientation procedure ensures both robustness and accuracy in the estimated camera pose, even in environments where we have to deal with essentially planar scenes.

C. Nonlinear Optimization

The methods described in the previous subsections are linear, so in fact they minimize an *algebraic* instead of a *geometric error* [14]. Thus, though essentially correct, the obtained solutions for the $(\mathtt{K}, \mathtt{R}, C)$ decomposition of the camera matrix are slightly suboptimal. Especially when we know the camera intrinsic parameters ($\mathtt{K}_{\mathtt{fixed}}$) with a high degree of accuracy, a final polishing of the solution by any standard nonlinear procedure which aims to minimize the geometric reprojection error is recommended (line 7 of algorithm in figure 3). Several alternatives are also available for this task [23], though in this work we simply use an straightforward iterative procedure implemented in the GNU scientific library. Our experiments show that this final optimization, to a greater or lesser extent, always improves the linear estimation.

D. From 3D model to real world coordinates

Obviously, in the absence of any real world reference, the SfM techniques can determine the 3D reconstruction only up to an arbitrary global euclidean transform. Nevertheless, our LBS must of course attach this 3D model to some convenient coordinate system defined for our physical environment, so that we obtain the localization results in real world coordinates. We can also define the desired augmented reality structures on this world coordinate system. Given a minimum of three ground control points (though, for numerical stability, it is convenient to use a few more), it is possible to estimate the corresponding parameters s', $\mathtt{R}'$ and C' of the needed *3D reconstruction* $\rightarrow$ *world* euclidean transformation solving again an absolute 3D orientation problem analogous to the one mentioned in subsection VII-B. Those parameters will be used to globally transform the whole 3D model to the desired final real world coordinates, where all the posterior resectioning process will take place.

VIII. Experimental Analysis

This section describes the experimental environment where we have carried out a set of tests with the aim of validating the proposal described in this paper. We also outline the software and hardware elements as well as the methodology to perform the training process. Finally, we discuss the details of one specific validation (among the several tests we performed) and their related results.

A. Experimental Environment

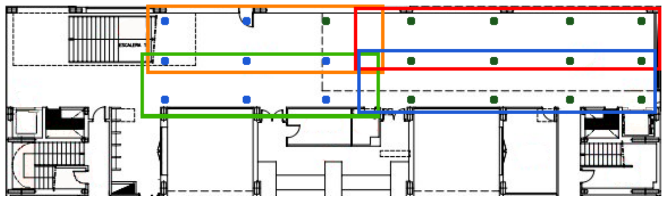

Fig. 4. Experimental environment. Colored dots indicate the RSSI clusters. Colored rectangles identify the 3D sub-models defined including orientations.

The testbed where our experiments were conducted is located on the ground floor of our Faculty. As it is shown in Figure 4 it is a 220 m^2 open space area. To build the fingerprinting map based on RSSI we took advantage of the access points deployed throughout the dependencies.

The training video and RSSI observations were captured using a Samsung Galaxy Tab Plus 7.0 with Android OS 3.2 and experiments were carried out making use of a Samsung Galaxy SII smartphone with Android OS 4.0.3. We made use of our own application, LOCUM-Trainer, to perform a fast training process that did not compromise the performance and accuracy of the localization system. Instead of defining a discrete space model where RSSIs, orientations, and images are acquired, we scanned the RSSIs of the different access points while we were video recording the environment and registering the orientation of the device. The operator just had to label the current zone where he was moving around (the map division into zones was done prior to the data capture). In this way, we obtained a lightweight fingerprinting map of RSSIs and a set of images (frames extracted from the recording) geo-tagged with the correspondent zone and orientation. Then, a clustering process was performed using the techniques described in Section IV in order to group these sample points into clusters according to the RSSIs and to the different orientations. These clusters determined the geographical areas dividing the space search for image analysis, as it was mentioned in Section VI.

During the on-line phase we extracted SIFT features to look for correspondences with the points of the 3D model. For image processing purposes we used the SIFTGPU library, a GPU Implementation of SIFT developed by Changchang Wu [30]. SIFTGPU requires a high-end GPU which is not available in smartphones, so we use a server with a nVidia GeForce GTX580 with 512 cores, supporting both GLSL (OpenGL Shading Language) and CUDA (Compute Unified Device Architecture). This GPU was installed in a Intel(R) Core(TM) i7-2600K CPU 3.40GHz. This server is responsible for estimating the location of the different devices. Additionally we have also used a modified version of the CUDA ENN software developed by V. Garcia [9] to perform the matching process, and our own implementation of the resection algorithm described in VII, making use of the QVision library, available in *SourceForge*.

A client-server architecture was designed to reduce the computational cost, to minimize the amount of information transmitted and, therefore, to preserve the smartphone battery life. During the on-line phase the smartphone uses different threads running in parallel in order to capture images and to obtain information from RSSIs, accelerometer and digital compass. We check whether the image is useful to avoid unnecessary transmissions, that is, it is focused and rich enough to allow the extraction of SIFT features. For this purpose, smartphones use the Sobel operator [11], an efficient linear filter operation which responds to sharp gray level changes (edges) in images. Then, making use of specific APIs, the smartphone is able to send the image to the server along with the RSSI information and its orientation. The motion estimator is locally used to detect whether a new image has to be sent to the server.

B. Building 3D maps

As we mentioned above, our proposal is mainly focused on the image analysis. Thus, we needed to obtain a set of images to build our 3D model. To minimize the training time, we recorded a 7’24” long video, capturing every detail of our environment. Since there are not many objects in the center of the hall, the video was recorded paying attention to the four main walls, with the camera situated at shoulder level. During our tests we learnt that a better reconstruction is obtained when the scene is recorded with the camera oriented to the walls and also including a second round using a 45 degrees angle of view. Once the video was recorded, and after several tests varying the frame extraction rate, we extracted one image per 24 frames, thus obtaining 524 different images. Those images were good enough to build the 3D model which was used for the following tests.

Our 3D model was built using the VisualSFM software [29], a visual structure from motion application that makes use of a set of input images to run 3D reconstructions. VisualSFM relies on the SIFTGPU library to extract the SIFT features and to perform the pairwise image matching process. The bundle adjustment is implemented using a GPU version that exploits the hardware parallelism.

In addition to the 3D model, VisualSFM also generates an output file containing the technical information about the model. This information includes the camera parameters, such as focal length (in our case, the focal length was manually established before the reconstruction took place), quaternion rotation, camera center and radial distortion, estimated for each one of the images (considered as individual cameras) included in the reconstruction. Moreover, it also contains information about each 3D point of the model, identified by its coordinates (x, y, z) and a set of SIFT features defining the point.

Finally, using the information extracted from the 3D model and according to the zones dividing our map, we built the search structures that contain the descriptors of the 3D points belonging to each submodel.

C. Validation results

In this section we present the experimental results of our proposal. A set of 20 images (640×480 pixels) is our validation data, and they were obtained while walking along our scenario using the rear camera of a Samsung Galaxy SII. Since the camera was previously calibrated, we knew its intrinsic parameters (focal length and aspect ratio). Results shown in Table I were obtained performing the resection process making use of the Exterior Orientation technique.

TABLE I
ESTIMATION ERROR (CM) AFTER THE RESECTION PROCESS USING EXTERIOR ORIENTATION AND DIFFERENT CONFIGURATION PARAMETERS

Nº Correspondences		**40**		**50**		**60**	
Minimum Inliers		**10**	**30**	**10**	**30**	**10**	**30**
Iterations	**Reprojection Error (px)**						
100	**4**	19.5	9.0	25.2	9.6	39.9	9.4
	5	14.4(*)	10.8	25.8	9.8	32.5	10.6
	6	16.3	10.5	27.2	10.1	37.08	15.0
200	**4**	10.8	8.6(**)	20.0	10.1	29.1	9.2
	5	14.6	10.7	15.9	9.7	18.1	10.9
	6	14.0	10.0	15.9	10.2	32.5	12.4
300	**4**	12.4	9.1	15.3	10.2	19.3	10.0
	5	14.1	10.6	11.6	9.5	23.4	11.0
	6	12.3	9.9	14.7	10.0	19.8	12.5

We initially evaluated the different configuration parameters related to camera resection process. As mentioned in Section VII, those parameters are: the number of correspondences used

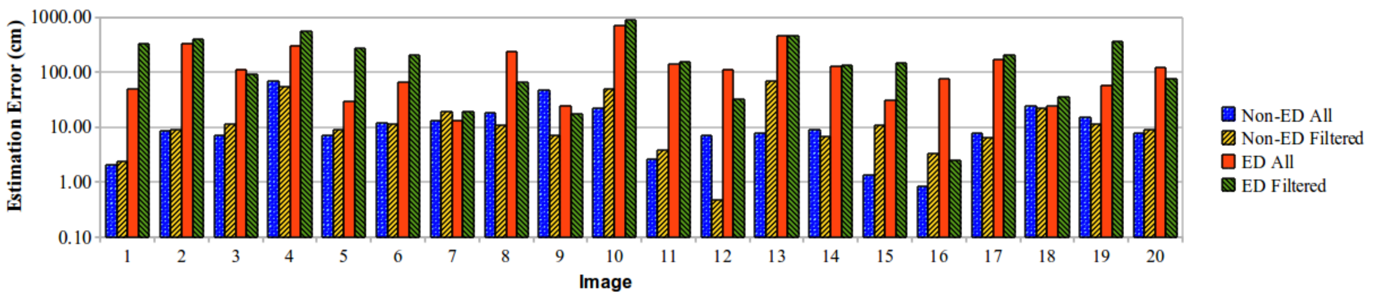

Fig. 5. Accuracy results for each validation image using Early-Detection (ED) and Non-Early Detection (Non-ED) versions of RANSAC, together with different sets of descriptors (using all of them or a filtered subset).

as input by the RANSAC process; the number of iterations performed by the RANSAC process; the reprojection error threshold used to classify a correspondence as inlier; and the minimum number of inliers required to consider the estimated camera resection matrix as valid. Table I shows part of the different combinations we have experimented. It is worth mentioning that RANSAC introduces a certain randomness to the process, and these values may slightly vary among different executions. Twenty independent executions per each parameter combination are averaged to provide stable results.

As we can see, there are several sets of parameters providing, on average, an estimation error under 15 cm. The choice of the most appropriate configuration depends on the associated processing time. For example, we consider that the set of parameters that offers the best trade-off between accuracy and performance is the one marked as (*) in Table I. Despite this option supposes a slight loss of accuracy in relation to other combinations, the involved time for the resection process is reduced down to 37% in relation to more accurate options. Moreover, this choice is less restrictive than the one providing the best accuracy results, since requiring a fewer number of inliers we were able to perform a valid camera resection in all the tests instead of the 90% of success obtained when using the most accurate option (**). It is worth noting that despite the selected set of parameters worked properly for our environment, it may differ from one scenario to another.

Furthermore, we have evaluated additional images captured with the aim of validating our proposal under difficult conditions. Some of them contain information about areas of the scenario that were not completely modeled. Other were taken with an inclination angle that exceeds the invariance angle of the SIFT descriptors used to build the 3D model. Taking into account these drawbacks, we were able to obtain an accuracy ranging from 0.8 to 3.7 meters. The analysis of these images demonstrates the robustness of this technique even when the problem is ill-conditioned, but also some of its limitations.

Once we have selected the best configuration of parameters, we have also evaluated how the number of descriptors in the search space influences on the accuracy and performance of the resection process. We have reduced the number of descriptors by filtering those defining the same 3D point and whose camera poses are closer than a specified threshold (100 cm). This reduction is based on the idea that each 3D point is formed by a set of SIFT descriptors, but some of them were obtained from nearby cameras, thus introducing some redundancy to the matching process because of their visual similarity. In this way, we have reduced the number of descriptors from 254631 to 86088. On the other hand, we have also evaluated the use of an Early-Detection (ED) version of the RANSAC process that may save up to 60% of processing time. It is based on the idea that once the threshold of inliers has been exceeded, it might not be necessary to execute the rest of RANSAC iterations since the obtained camera matrix P might be good enough to estimate the camera pose accurately.

Figure 5 shows the histogram that reflects the accuracy obtained for each validation image when combining the above mentioned proposals. Despite its better performance, ED shows less accuracy than the complete version of RANSAC, 159 cm and 14.4 cm respectively on average. Moreover, we can observe how accuracy is barely decreased when a filtered set of descriptors is used. In those cases, the mean estimation error is 16.36 cm, but the required matching time (Figure 6) is reduced down to 67% in relation to the whole set of descriptors.

A zone-based search space (the multisensor approach) reduces (62%) the time required to carry out the matching process, as Figure 6 shows. We also observe that the time needed to complete the whole cycle from sensor data acquisition to 3D estimation is around 345 milliseconds. We have detailed the transmission time of sensor data from the smartphone, the required time at the server to perform the SIFT extraction, to run the matching process, and to apply the resection technique.

It is worth mentioning that in those cases where the intrinsic camera parameters are unknown, we can make use of the DLT algorithm described in Section VII to carry out the resection process. A mean estimation error around 78 cm is obtained and the elapsed time is similar to the one obtained with Exterior Orientation. Therefore, our proposal is able to address reasonably well even those situations where there is no previous knowledge of the camera configuration.

IX. Conclusions

As emphasized during the paper, the multisensor approach is crucial to accomplish a robust and scalable image-based location based service. By incorporating SfM techniques we have

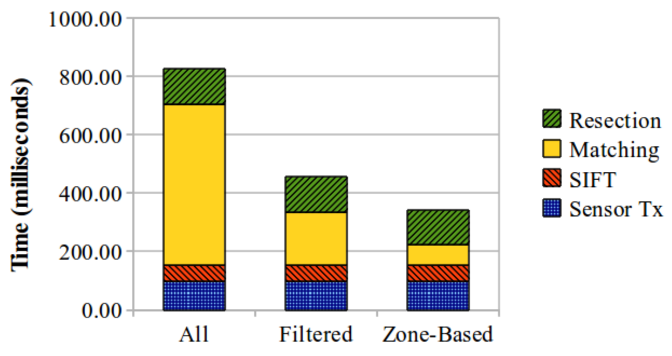

Fig. 6. Performance using different alternatives to build the search space.

remarkably augmented both the scalability and precision of the training phase, not only eliminating the need to discretize the environment at a given granularity, but also improving the quality of the location estimations. All these advantages are obtained using a medium number of training images from a video recording, a key fact that makes the training phase definitely easier and thus much more easy to deploy. The result is an accurate 3D model of the environment ready to perform the desired full 6 *dof* accurate localization process for any new input image.

Locations are obtained with a mean error of 15 cm, and with the additional bonus of estimating the rotation with a similar precision. Moreover the mean time required to complete an entire cycle is 345 milliseconds, making this proposal able to support those applications requiring high accuracy and rapid response, such as augmented reality applications. Additionally, the multisensor approach provides mechanisms to work in difficult scenarios such as those presenting similar areas from a visual perspective (floors, rooms). In those cases, SIFT features can fail to disambiguate, but other signals like RSSIs can be crucial to perform an accurate resection process.

For all these reasons, we find our approach a valuable contribution for this kind of location-based services, since we have obtained a good trade off between a fine-grained accuracy and an acceptable response time using data from multiple sensors.

X. Acknowledgments

This work was supported by the Spanish MICINN, Plan E funds, under Grant TIN2009-14475-C04-02/01.

References

[1] I. Arai, S. Horimi and N. Nishio. Wi-Foto 2: Heterogeneous Device Controller Using Wi-Fi Positioning and Template Matching. In Proc. Pervasive 2010.
[2] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions. In ACM-SIAM Sympos. Discrete Algorithms, 1994.
[3] M. Azizyan, I. Constandache, and R. Roy Choudhury. SurroundSense: Mobile Phone Localization via Ambience Fingerprinting. In Proc. of the 15th MOBICOM, pages 261-272, 2009.
[4] P. Bahl and V. N. Padmanabhan. RADAR: An In-Building RF-based User Location and Tracking System. In 19th Joint Conference of the IEEE Computer and Communications Societies, volume 2, pages 775-784, 2000.
[5] G. Chandrasekaran, M. A. Ergin, J. Yang, S. Liu, Y. Chen, M. Gruteser, and R. P. Martin. Empirical Evaluation of the Limits on Localization Using Signal Strength. In Proc. SECON 2009, pages 333-341, 2009.
[6] P. Fiore, ”Efficient linear solution of exterior orientation”, IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 23, pp 140-148, 2001.
[7] M. A. Fischler, R. C. Bolles, ”Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography”. Communications of the ACM, vol. 24, pages 381-395, 1981.
[8] F. Frauendorfer, C. Wu, J. M. Frahm and M. Pollefeys, Visual Word based Location Recognition in 3D Models Using Distance Augmented Weighting. In Proc. of the 4th 3DPVT, 2008.
[9] V. Garcia and E. Debreuve and M. Barlaud. Fast k nearest neighbor search using GPU.In Proceedings of the CVPR Workshop on Computer Vision on GPU, Anchorage, Alaska, USA, pages 1-6, June 2008.
[10] Gene H. Golub, Charles F. Van Loan, ”Matrix Computations”, 3^{rd} edition, Johns Hopkins Studies in Mathematical Sciences, 1996
[11] R.C. Gonzalez and R.E. Woods. Digital Image Processing, 2nd edition. Prentice Hall, 2002.
[12] J. C. Gower, G. B. Dijksterhuis, ”Procrustes Problems”, Oxford University Press, 2004.
[13] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki. Practical Robust Localization over Large Scale 802.11 Wireless Networks. In Proc. MOBICOM 2004, pages 70-84, 2004.
[14] R. Hartley, A. Zisserman, ”Multiple View Geometry in Computer Vision”, 2^{nd} edition. Cambridge University Press, 2011.
[15] K. Hattori, R. Kimura, N. Nakajima, T. Fujii, Y. Kado, B. Zhang, T. Hazugawa, and K. Takadama. Hybrid Indoor Location Estimation System Using Image Processing and WiFi Strength. In Proc. WNIS, pages 406-411, 2009.
[16] R. A. Horn, C. R. Johnson, ”Topics in Matrix Analysis”, Cambridge University Press, 1991.
[17] K. Kanatani. Geometric Computation for Machine Vision. Oxford University Press, 1993.
[18] K. Konolige and K. Chou. Markov Localization using Correlation. In Proc. IJCAI 1999, pages 1154-1159, 1999.
[19] H. Lemelson, M. B. Kjaergaard, R. Hansen, and T. King. Error Estimation for Indoor 802.11 Location Fingerprinting. In Proc. LoCA 2009, pages 138-155, 2009.
[20] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision, Volume 60, pages 91-110, 2004.
[21] D. M. Mount and S. Arya. ANN, a Library for Approximate Nearest Neighbor Searching. CGC Workshop on Computational Geometry, 1997.
[22] A. Mulloni, D. Wagner, I. Barakonyi, and D. Schmalstieg. Indoor Positioning and Navigation with Camera Phones. IEEE Pervasive Computing, 8:22-31, 2009.
[23] J. Nocedal and S. Wright, ”Numerical Optimization”, Springer, New York, 1999.
[24] A. J. Ruiz-Ruiz, O. Canovas, R. A. Rubio, P. E. Lopez-de-Teruel. Using SIFT and WiFi Signals to Provide Location-Based Services for Smartphones. In Proc. of MobiQuitous, pages 37-48, 2011.
[25] A. J. Ruiz-Ruiz, O. Canovas. Integrating probabilistic techniques for indoor localization of heterogeneous clients. In Proc. of JITEL 2011.
[26] S. Se, D. G. Lowe and J. J. Little. Vision Based Global Localization and Mapping for Mobile Robots. IEEE Transactions on Robotics, 21, 3, pages 364-375, 2005.
[27] B. Triggs, P. F. Mclauchlan, R. I. Hartley, A. W. Fitzgibbon, ”Bundle Adjustment – A Modern Synthesis”. Lecture Notes in Computer Science, Vol. 1883, 2000.
[28] M. Werner, M. Kessel and C. Marouane. Indoor Positioning Using Smartphone Camera. In IPIN 2011.
[29] C. Wu, ”VisualSFM: A Visual Structure from Motion System”, http://www.cs.washington.edu/homes/ccwu/vsfm/, 2011.
[30] C. Wu, ”SiftGPU: A GPU implementation of Scale Invaraint Feature Transform (SIFT)”, http://cs.unc.edu/ ccwu/siftgpu, 2007
[31] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz, ”Multicore Bundle Adjustment”, CVPR 2011