Markerless Vision-based Augmented Reality for Urban Planning

Ludovico Carozza,

ARCES, University of Bologna, Italy

David Tingdahl,

ESAT/IBBT/VISICS, KU Leuven, Belgium

Frédéric Bosché*

Heriot-Watt University, Edinburgh, Scotland

&

Luc van Gool

Computer Vision Lab, ETH Zurich, Switzerland
& ESAT/IBBT/VISICS, KU Leuven, Belgium

Abstract: *Augmented Reality (AR) is a rapidly developing field with numerous potential applications. For example, building developers, public authorities and other construction industry stakeholders need to visually assess potential new developments with regard to aesthetics, health & safety, and other criteria. Current state-of-the-art visualization technologies are mainly fully virtual, while AR has the potential to enhance those visualizations by observing proposed designs directly within the real environment.*

A novel AR system is presented, that is most appropriate for urban applications. It is based on monocular vision, is markerless and does not rely on beacon-based localization technologies (like GPS) or inertial sensors. Additionally, the system automatically calculates occlusions of the built environment on the augmenting virtual objects.

Three datasets from real environments presenting different levels of complexity (geometrical complexity, textures, occlusions) are used to demonstrate the performance of the proposed system. Videos augmented with our system are shown to provide realistic and valuable visualizations of proposed changes of the urban environment. Limitations are also discussed with suggestions for future work.

1 INTRODUCTION

Public authorities, building developers and other construction industry stakeholders need to assess the impact of potential urban developments with regard to aesthetics, health & safety, buildability and many more criteria. Current state-of-the-art visualization technologies are mainly fully virtual. In comparison, Augmented Reality (AR) has at least one crucial advantage, namely that designs can be visualized directly within the real environment instead of within an entirely virtual world (De Filippi and Balbo, 2011), (Woodward et al., 2010).

This article presents an AR system aiming towards applications in which the urban environment is augmented with *virtual static and dynamic objects*, such as buildings and people. The distinctive characteristics of the proposed system are that it is markerless and does not rely on any local or global positioning technology, or any inertial sensors; the system only uses digital images. In addition, the proposed system can accurately generate occlusions of the augmenting material (i.e. the inserted virtual objects) by the real static environment. This significantly contributes to highly realistic outputs – although occlusions by real dynamic objects cannot be addressed by our current system.

The rest of the article is organized as follows. Section 2 reviews existing related work in AR and localization (the most challenging task for AR systems). Then, Sections 3 to 6 detail the approach developed in the proposed system. A detailed analysis of the system's performance with multiple experiments conducted using real data follows in Section 7. Finally, Section 8 draws conclusions and adds considerations on open issues and future developments.

It is important to note that, although the proposed system is presented in the context of visualization of urban developments, it is certainly applicable in other contexts.

2 BACKGROUND: AR AND LOCALIZATION

The most challenging task for AR systems is the accurate localization of the viewpoint within the given scene. Localization refers to the calculation of both the location and orientation of the viewpoint. The problem of real-time localization in environments with different complexity has been extensively investigated in recent years for its key role played in Robotics and AR (Azuma et al., 2001), (Durrant-Whyte and Bailey, 2006).

Localization can be performed with different – often integrated – sensors such as global and local positioning technologies (e.g. GPS, WIFI), Inertial Measurement Units (IMU) and environment sensors (Azuma et al., 2001), (Comport et al., 2006), (Shin and Dunston, 2009), (Sanghoon and Omer, 2011), (Behzadan and Kamat, 2010). The latter – e.g. digital cameras, radars, laser scanners – enable environment *mapping* and subsequently *localization*. During *mapping*, features are extracted from sensed data and localized, creating environment landmarks. During *localization*, features are similarly extracted from the sensed data and matched to the mapped features. The matches are used to estimate the pose of the sensor. In Robotics, localization is also often performed jointly with mapping of unknown environments, following a Simultaneous Localization and Mapping (SLAM) approach (Leonard and Durrant-Whyte, 1991), (Durrant-Whyte and Bailey, 2006), (Klein and Murray, 2007), (Newcombe and Davison, 2010).

2.1 Offline mapping

When the environment is *known* a priori, then it is possible to map it offline. Offline mapping can be achieved in different ways:

- *Fiducial markers* can be introduced and localized in the environment, so that online localization can be achieved by simply recognizing them using an appropriate sensing pipeline (Sanghoon and Omer, 2011), (Yakubi et al., 2011).
- *Physical features* of the scene can be learnt and mapped offline using environment sensing, and online localization is achieved by using a similar sensing pipeline (Reitmayr and Drummond, 2006), (Gordon and Lowe, 2004).

Fiducial marking typically leads to more accurate localization results, but it requires intrusive and accurate positioning of markers within the environment. Markerless systems, on the other hand, are not invasive, but may result in less reliable positioning (Reitmayr and Drummond, 2006). Within markerless systems, vision-based mapping and localization is very popular, because (1) digital cameras are robust, compact and inexpensive (Neumann et al., 1999); and (2) Structure-from-Motion (SfM) algorithms are providing a sound tool for scene mapping (Pollefeys et al., 2004).

Vision-based localization systems can use several types of features to map the environment, such as lines, points or even registered images (Gordon and Lowe, 2004), (Karlekar et al., 2010), (Reitmayr and Drummond, 2006), (Simon et al., 2000), (Inam, 2009). For instance, Inam (2009) uses line features to locate robots with respect to soccer goal posts in the RoboCup contest, while Wolf et al. (2005) present a strategy based on image retrieval for the localization of a robot in an indoor environment.

2.2 Localization

During *online* localization, *robustness* and *stability* in the camera pose estimation throughout its motion are challenging requirements (Comport et al., 2006). This is particularly true in AR applications: since the visualization of the real scene is to be augmented with virtual objects in a scene-consistent way, any localization error can result in gross errors in the augmented imagery. As a result, for improved robustness but also efficiency, tracking strategies are commonly implemented, e.g. Kalman Filtering (Bishop andWelch, 2001), (Capp et al., 2007). Non-deterministic approaches using Monte-Carlo Localization are also used (Wolf et al., 2005), (Inam, 2009).

Furthermore, it is possible during online processing to incrementally expand the initial scene maps by identifying new reliable features with SLAM-type approaches (Saeki et al., 2009).

3 OVERVIEW OF PROPOSED SYSTEM

We propose a *markerless monocular vision-based* approach for localization within an urban scene. Since in our context the environment can be visited beforehand, a map of the environment can be learnt offline. Consequently, our system particularly relates to the works of Gordon and Lowe (2004) and Newcombe and Davison (2010). Like them, we do not rely on any global positioning or inertial sensors. And like Newcombe and Davison (2010), our mapping stage simultaneously performs a dense 3D mesh reconstruction of the scene, so that static occlusions of the learnt scene on the inserted virtual objects can be taken into account when augmenting the target imagery. However, compared to Newcombe and Davison (2010) who focus on SLAM, we work with less controlled outdoor scenes and do not aim to reconstruct them in real-time. Additionally, our context is different, as the scale of the scene is typically set by the augmentations (i.e. the inserted virtual objects) and the positioning and scaling of the 3D reconstructed scene with respect to these augmentations must be performed very accurately. In the gaming AR example presented in Newcombe and Davison (2010) this is not an issue: scaling and even localization of the augmenting material with respect to the real environment is not critical and thus can be defined arbitrarily.

Our system uses a two-stage approach (see Figure 1):

1. ***Offline* learning/training stage**. During this stage, the 3D scene is first learnt and then *augmented* with virtual elements. For mapping the scene, Speeded-Up Robust Features (SURF) (Bay et al., 2006) are extracted from a set of *training images* and assigned 3D coordinates by using a SfM algorithm followed by a robust Euclidean bundle adjustment algorithm. This process constructs a *map of 3D-referenced visual features*, also called *database* hereafter. Subsequently, a dense reconstruction of the scene is produced, resulting in a 3D mesh of the scene. This mesh is used (1) offline to augment the scene with virtual objects, and (2) online to compute occlusions.
2. ***Online* processing stage**. During this stage, images of a *target image sequence* (e.g. a video sequence) are processed sequentially. For each target image, SURF features are first extracted from it and efficiently matched with the mapped ones. Correspondences are then used to estimate the camera pose within the learnt scene using a robust approach. Finally, the target image is augmented by projecting on it the virtual scene objects, taking into account occlusions of the virtual objects by the reconstructed scene.

The processes used in the offline and online stages are detailed in Sections 4 to 6. Note that the online stage is entirely automated, while the offline is almost entirely automated. The only manual step is the insertion of virtual elements in the reconstructed 3D scene model.

4 OFFLINE LEARNING/TRAINING STAGE

4.1 Learning the scene

The input to the learning stage includes a set of images of the scene of interest, called *training images*, with corresponding camera intrinsic parameters. The aim is to build a map of 3D-referenced SURF features (Bay et al.,

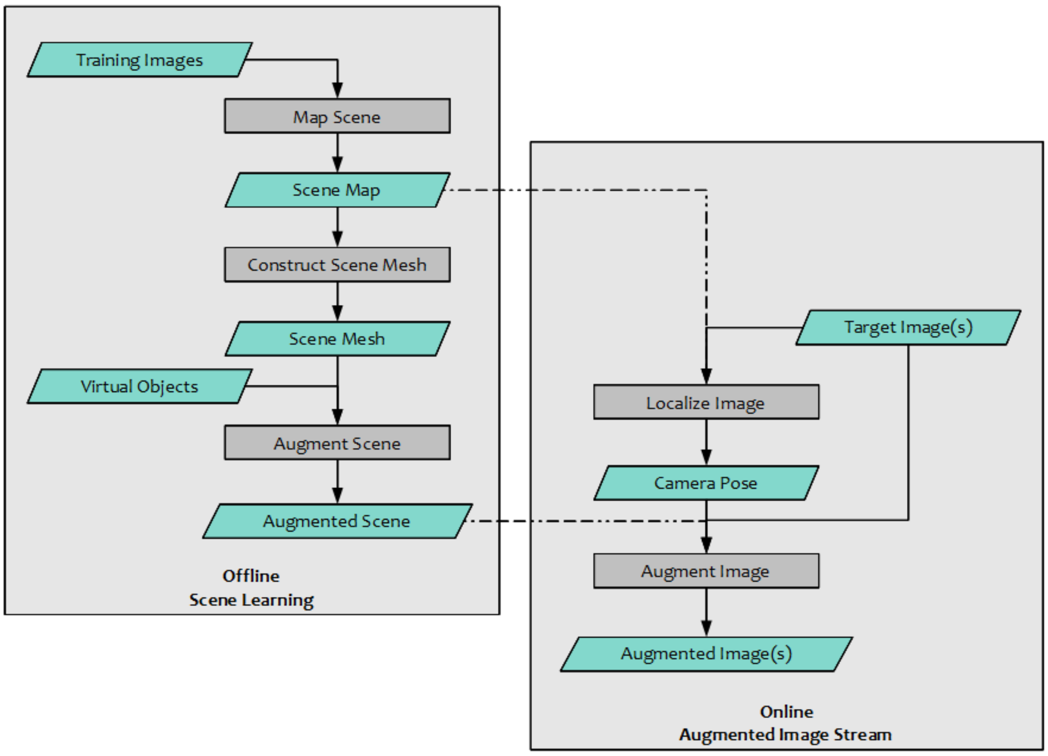


Figure 1 Overview of the proposed image-based AR system.

2006) extracted from the training images. This can be performed in different ways, including:

- **Using a prior non-textured 3D model of the scene**: Through a Graphical User Interface (GUI), for each training image the user manually matches several 3D model points with their corresponding image points. From these 2D-3D correspondences and knowing the camera intrinsic parameters associated with the image, the pose of the associated camera within the 3D scene model is estimated using the 3-point algorithm (Haralick et al., 1994). Knowing this camera pose with respect to the 3D scene model, the 3D coordinates of SURF features extracted from the training image are calculated by reprojecting them onto the scene model.
- **Using SfM and Bundle Adjustment**: SfM enables the 3D-registration of the training images' cameras with respect to one another. SURF features are used in an initial sparse matching step to find corresponding points between images that are triangulated into a 3D point cloud. A subsequent robust Euclidean Bundle Adjustment from candidate views directly registers the already extracted SURF features in the reconstructed Euclidean 3D reference frame to build the *map of 3D-referenced features*. This approach, summarized in Figure 2, is fully automated. We use the ARC3D framework (Tingdahl and Van Gool, 2011) for 3D reconstruction and self-calibration.

  Note that the robustness of SURF descriptors to scale changes allows to relax some constraints about camera motion during the reconstruction process (normally constrained to turn around the scene to be reconstructed), permitting the combination of camera paths at different distances from the building.

These two training approaches were tested with image sets acquired from the main square of the EPFL campus in Lausanne (Figure 3(a)). When these sequences were acquired, camera motion was not constrained in any particular way. For the first approach, a rough, but typical, untextured 3D CAD model of the scene was used (Figure 3(b)). The results however showed that the quality of such coarse 3D models is too low for successful image registration: the manual feature matching leads to registrations that were found too inaccurate, as can be seen with the reprojected model wireframe in Figure 3(c). Additionally, this model-based training approach requires a potentially long mapping procedure involving substantial human interaction. For these reasons, the second approach based on SfM was preferred (whose results with the EPFL dataset are reported in Section 7.2).

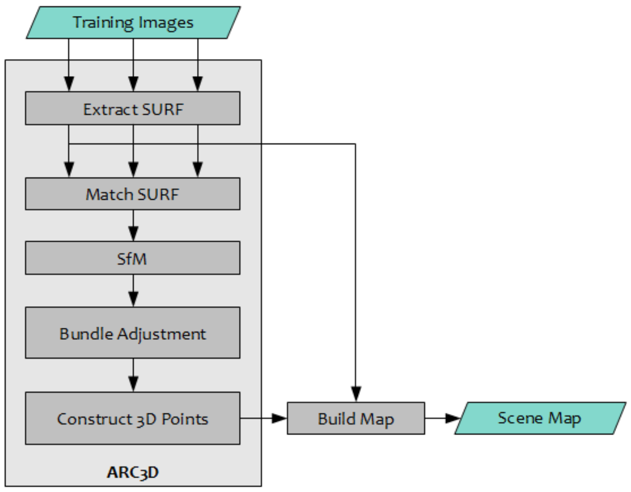


Figure 2 Outline of the offline training stage. SURF descriptors and 3D points reconstructed by ARC3D on a sequence of training images are used to build the map of 3D-referenced SURF features.

4.2 Augmenting the scene

As previously mentioned, we use the ARC3D framework (Tingdahl and Van Gool, 2011) for mapping the scene. Similar work can also be found in (Zhang and Elakser, 2012).

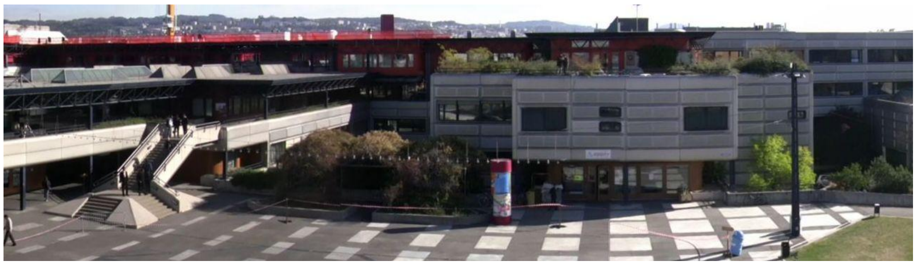
ARC3D actually provides us with an important feature that is of particular interest to our system. In addition to the 3D reconstruction, using the same input images ARC3D enables a dense reconstruction of the acquired scene in the form of a 3D mesh. Compared to the point cloud of the reconstructed map, this mesh offers two advantages:

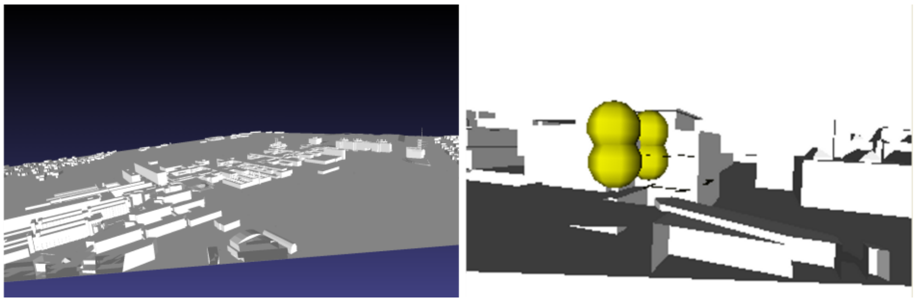
1. It visually simplifies the manual insertion of virtual objects in the scene.
2. During online processing, it enables the computation of occlusions of the virtual objects by the reconstructed scene.

Figure 4 shows an example of the dense reconstruction of the EPFL campus scene augmented with a virtual building.

Note that in the case when a virtual object is aimed to replace an existing one (e.g. a building is planned to be demolished and replaced by a new one), the user must remove from the ARC3D-reconstructed mesh the parts corresponding to the objects to be replaced. This ensures that occlusions caused by the objects to be replaced are not taken into account when augmenting the target images with the new objects (see Section 6 and example in Figure 11).

The organization of the map data (i.e. 3D-referenced feature points and corresponding SURF descriptors) must take into account the way the data is utilized and the constraints that are faced during the online camera pose estimation; it should thus prefer some aspects rather than


(a)


(b)

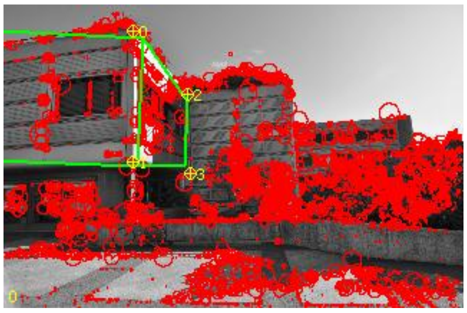

(c)

Figure 3 Illustration of the limitations of the mapping approach using a non-textured 3D model of the scene. (a) View of the EPFL campus scene; (b) Non-textured 3D model of the EPFL campus in Lausanne; (c) Reprojection of the wireframe lines of the 3D model of a building during the training stage.

others. The strategy followed is presented in the following section along with the online localization procedure.

5 ONLINE LOCALIZATION

5.1 General approach

During *online* operations, the system processes the *target image sequence* (e.g. images from a video sequence). For each target image, SURF features are extracted and matched with the SURF descriptors in the database (using the Euclidean distance in a 64-dimensional space). Matched descriptors allow the system to establish correspondences, called *matched 3D points*, between the 2D image coordinates of the target image features and the 3D coordinates associated to the matched map features. Knowing the camera intrinsic parameters, the camera pose is then estimated from these correspondences by wrapping the 3-point algorithm (Haralick et al., 1994) in a Random Sampling and Consensus (RANSAC) framework (Fischler and Bolles, 1981). This results in an initial pose estimation that is subsequently used in a *Guided Refinement (GR)* process. In this process, the frustum-culled database 3D points are reprojected on the image plane of the target image, and matches to the target image descriptors are identified within a radius of ρ_{2D} pixels (we use $5 \leq \rho_{2D} \leq 15$ pixels). This enables reassessing all initial matches and identifying additional ones. A refined pose estimation is then obtained by putting all matches into a Levenberg-

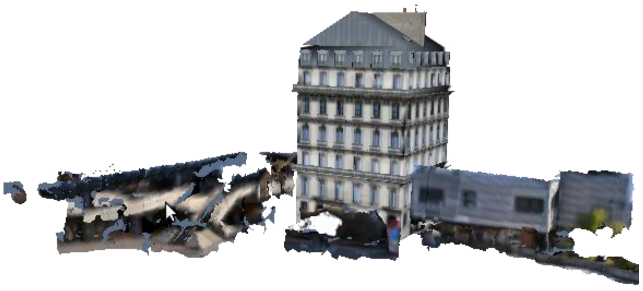


Figure 4 Dense reconstruction of the EPFL campus scene obtained using ARC3D (Tingdahl and Van Gool, 2011). The scene is then augmented with a virtual Parisian-styled building.

Marquardt non-linear pose optimization algorithm (Nocedal and Wright, 2006).

5.2 Matching process optimization

As the number of training images increases, along with their sizes, the map of 3D-referenced features can rapidly become very large (thousands of features). As a result, to avoid latency in the system, a strategy must be in place to prevent brute force matching during online operations.

We first implement a commonly used strategy that consists in partitioning the SURF descriptor space into a k-d tree (Beis and Lowe, 1997), (Gordon and Lowe, 2004), so that only the subspace associated with the target descriptor is visited, effectively reducing the computational payload.

Secondly, we propose to discard SURF features with low reliability, or *strength*, from the database and from the target image. SURF strength is associated to the Hessian response (Bay et al., 2006). It depends on the scene feature, the viewpoint and the lighting condition of the image, thus capturing the *repeatability* of the feature compared to the other ones in that particular image. For the target image, we select the S_{target} strongest features for matching (we use S_{target} = 1, 500 by default).

For the database, the situation is more complex as the features effectively come from multiple training images within which they present varying strengths. In order to obtain a *global SURF strength*, we propose to assign to each 3D reconstructed point the average of the strengths of the SURF features corresponding to that point. This way, SURF descriptors can be sorted globally and only those with a high average repeatability can be retained. Of course, more sophisticated algorithms for weighting the different strengths could be implemented. The actual number of database features retained for matching then depends on three different configurations, as detailed below.

Additional heuristics can be used to increase matching performance depending on three different configurations, or modes, that can be encountered: (1) *Pose initialization*; (2) *Pose tracking*; (3) *Pose resetting*. The cases when these modes are considered and the methods applied for optimizing matching in each case are detailed in the following sub-sections. Figure 5 summarizes the on-line localization processing strategy.

5.2.1 Pose initialization. This mode is applied when no reliable information on previous camera poses is available, i.e. if the pose of the camera has not been successfully tracked within the last *n* images (we use *n* = 20). Note that this mode is applied at least once, on the very first target image.

Using only the general approach described earlier, the (globally) strongest map features would be used for matching. The issue is that if too few features are used, these may not sufficiently cover the entire scene for a sufficient number of matches to be obtained, and consequently an initial pose to be confidently estimated. Therefore, it must be ensured that enough of the strongest database features are used and these also cover the 3D scene as uniformly as possible. We refer to this criterion as *spatial spread*.

To achieve this spread, we arrange the 3D-referenced feature points into an octree, where each cell represents a partition (cuboid) of the 3D scene. A *full octree* is first populated with all the 3D feature points, splitting each cell once the number of points it contains reaches a threshold N'_{max} (we use N'_{max} = 400).

Then, we construct a *pruned octree*. This is done by removing from the full octree all the cells with a side smaller than V_{min}. For each removed sub-tree, all its feature points are aggregated in the parent cell (i.e. the first cell without a side smaller than V_{min}). Finally, for each of those parent cells (which are now leaf cells), the features are sorted according to their global strength and only the N_{max} strongest ones are retained. Those 3D feature points represent a sub-set of the original point set that is as homogeneously spatially spread as possible. In our setup, we use the parameter values N_{max} = 200 and V_{min} = 5m. Note that N_{max} can be automatically derived from V_{min}. Note also that the full and pruned octrees both only need to be calculated once offline.

In summary, in initialization mode, the search for feature matches is performed on the pruned octree constructed offline, and with the strongest features of all its leaf nodes arranged in a k-d tree.

5.2.2 Pose tracking. This mode is applied when the poses of the two previous target images were successfully calculated. This means that some knowledge about previous camera poses is available. Assuming linear camera dynamics, a prediction of the pose of the current camera can be made using an Extended Kalman Filter (EKF) (Bishop and Welch, 2001). The target image features may then be matched only against the database features that lay inside the predicted frustum, resulting in a significant simplification of the matching stage. Near and far culling planes with distances set to 0m and 50m respectively are added to the camera frustum for culling. Note that, for additional speed-up, culling is done on the full octree rather than directly on all the 3D points. Furthermore, only the $S_{frustum}$ strongest features of the points in the cells intersecting the predicted view are considered for matching and organized in a k-d tree. We use $S_{frustum} = S_{target}$.

To prevent the system from considering unreasonable predictions, we reject any prediction with a change in camera orientation larger than $\delta\alpha_{max}$, and the next pose is then re-computed in Reset mode. We use $\delta\alpha_{max}$ = 0.1 rad ($\simeq$ 5 deg).

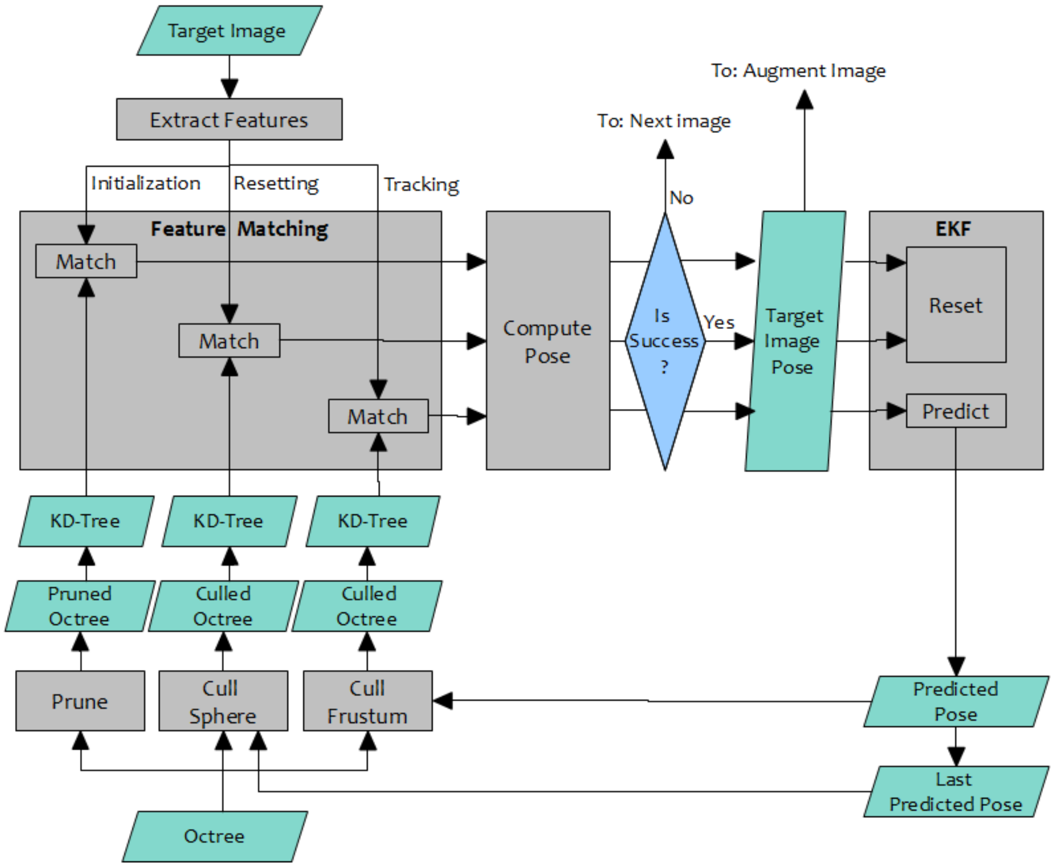


Figure 5 Outline of the on-line processing stage, emphasizing the strategies chosen to reduce the search space for efficient feature matching.

5.2.3 Pose resetting. This mode is applied when pose tracking has failed, but was successful for one of the last *n* images. Using the location component of the pose of the last successfully tracked image, we then cull the full octree using a sphere centered at that location and with a radius of ρ_{sphere} = 50m. Here as well, only the S_{sphere} strongest features of the points located in the sphere are considered for matching and organized in a k-d tree. We use $S_{sphere} = 4S_{frustum}$.

6 ONLINE IMAGE AUGMENTING STAGE

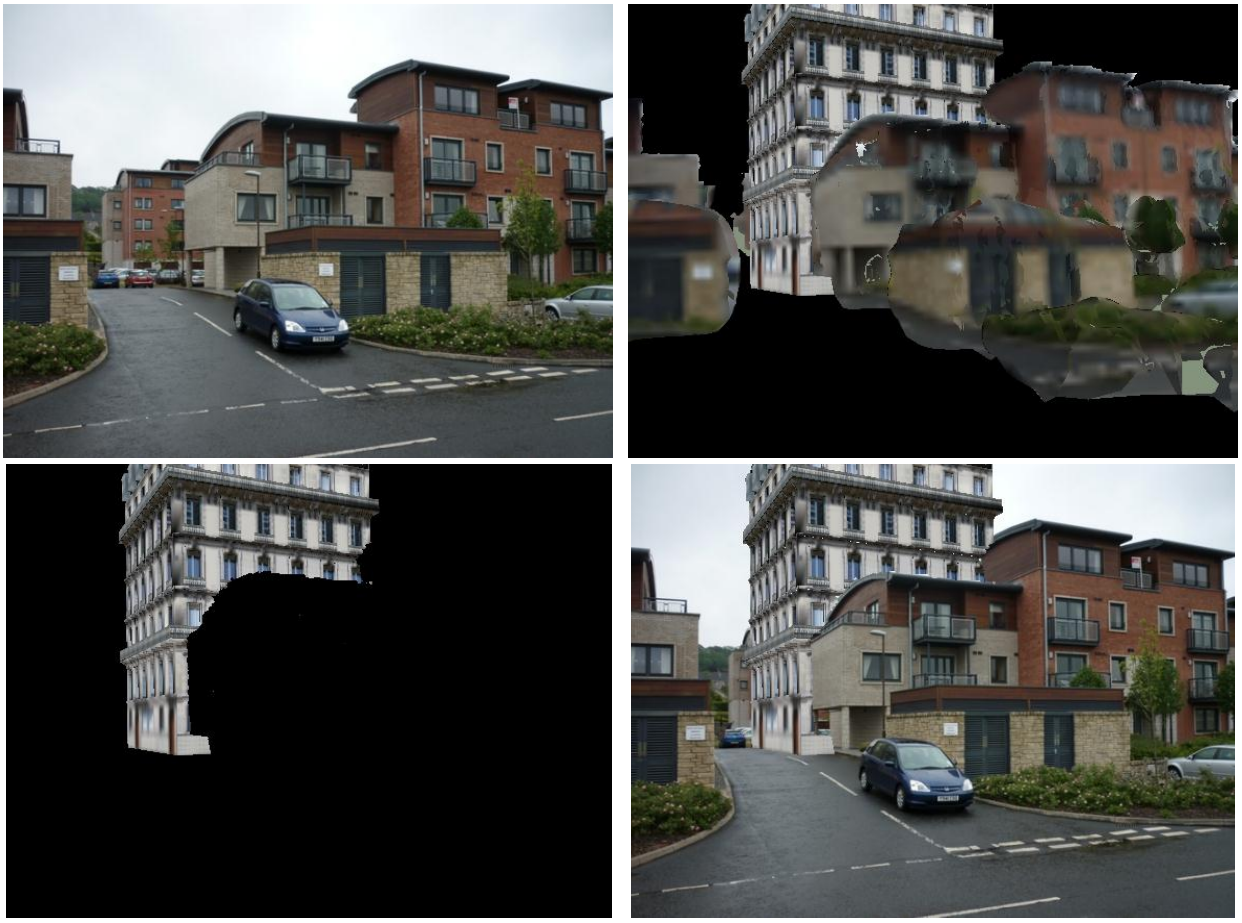
If the system has confidently calculated the pose of the camera for a given target image, this image is augmented with the projection of the virtual objects. In order for this projection to take into account occlusions of the inserted virtual objects by the reconstructed 3D scene, the following procedure is used (illustrated in Figure 6). Using the calculated pose and the target image intrinsic parameters, a virtual camera is positioned in the augmented 3D scene, which leads to a virtual image. Then, for each pixel in the virtual image, if the first intersected object is one of the inserted virtual objects, then the color value of the corresponding pixel in the real target image is changed to the color obtained at the point of intersection. Otherwise (i.e. if there is no intersection or the first intersection is the reconstructed scene model), the pixel in the real target image remains unchanged.

7 EXPERIMENTS AND FINDINGS

We now present results of several experiments. The first experiment aims at demonstrating the general performance of the proposed system. Additional experimental results are then shown on two other datasets, including the EPFL scene for which the use of a pre-existing coarse 3D CAD model proved unsuccessful (see Section 4.1).

7.1 1st experiment

7.1.1 Dataset. In a first experiment, 22 training images were taken by a person walking around a residential neighborhood. The pictures have a 2048x1536 resolution (~ 3M pixels) (see Figure 7). The processing of the images resulted in two files: the list of 3D-referenced SURF features; and the reconstructed 3D mesh of the scene, which

(a)  (b)

(c) (d)

Figure 6 The process to augment a target image: (a) Target image; (b) Virtual image obtained from the estimated pose and the target image's internal parameters; (c) Texture to be superimposed (overlaid) on the target image; (d) Augmented target image.

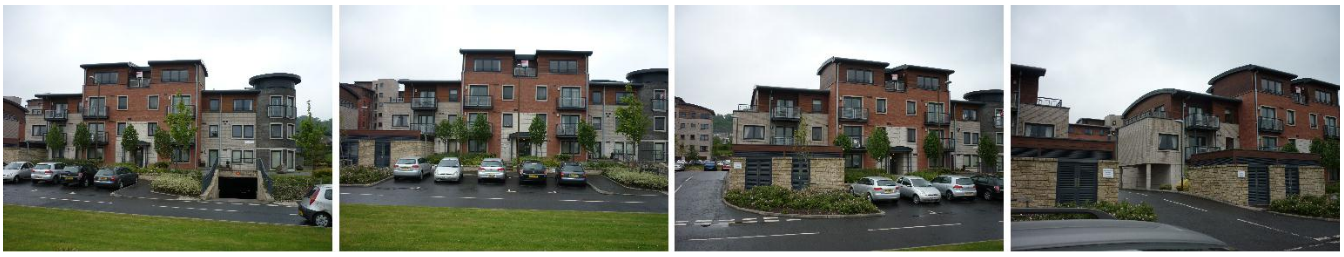


Figure 7 Experiment 1: Four of the 22 training images used to reconstruct the database of 3D-registered SURF features and the dense 3D model of the scene.

was then augmented with an additional building (see Figure 8). Later on, a set of target images was acquired using the same digital camera and the same resolution (2048x1536). In order to simulate a video sequence, the 120 target images were acquired in 'burst mode'.

A second training and target image dataset was then created by downsampling all the initial images to a resolution of 640x480, and processing them the same way. The reason for building these two training and two target datasets was to test (1) the impact of image resolution on the offline and online stages, as well as (2) the impact of resolution differences between the training and target images.

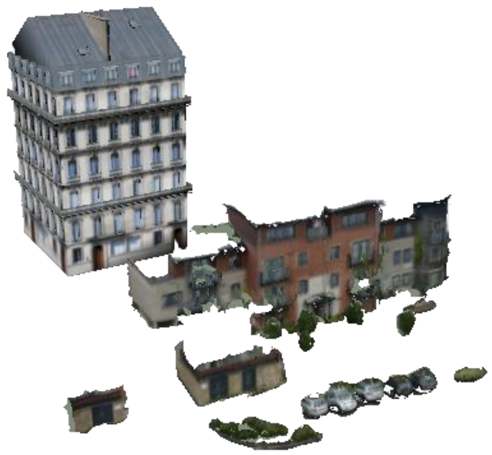


Figure 8 Experiment 1: Augmented 3D scene.

7.1.2 Results. Figure 9 shows the results obtained for some of the input target images. The results shown are for the training and target images all having a resolution of 2048x1536. Remember that any error in the pose estimation should lead to gross errors in the augmented images, that would be particularly obvious in the case of occlusions. With this in mind, a visual analysis of the results shows that all poses were successfully calculated.

It should be noted that in this experiment, at four occasions when a target image was processed in tracking mode, a sharp acceleration in the camera orientation occurred and the EKF prediction resulted in a change of camera orientation slightly larger than $\delta\alpha_{max}$ – the reason for these large orientation changes between consecutive images is the low-frequency of the experienced data acquisition (~ 1 fps). This resulted in the estimated pose being rejected by the system and recalculated in reset mode. In all cases, this recalculation was successful. So, the system seems to achieve accurate pose estimations and effectively recover from tracking failures.

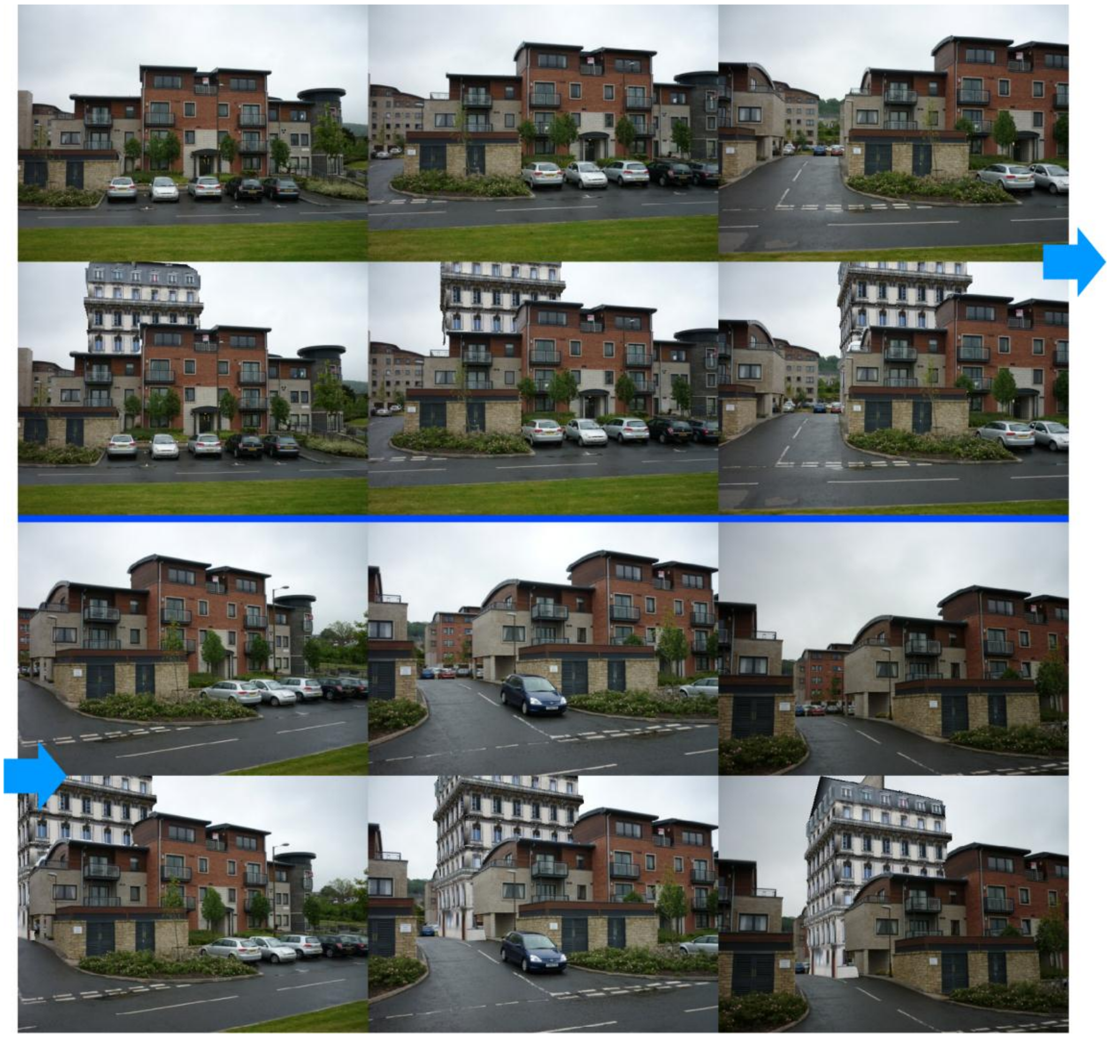


Figure 9 Experiment 1: 6 of the 120 target image stream before (lines 1 and 3) and after being augmented (lines 2 and 4).

Tables 1, 2, 3 present quantitative results on the pose estimation performance. The three tables report results for experiments conducted with and without two options:

- *Multiple Matching (MM)*: a target image feature can be matched with several database (DB) feature descriptors. Such strategy may be considered due to the possible presence of numerous self-similarities in the scene (in terms of geometry and texture), which could result in false yet strong matches that could in turn yield false yet well-supported camera pose estimations.
- *Guided Refinement (GR)*: the guided refinement stage described in Section 5 aims at increasing the number of matches, so that the correct pose is more likely to be recovered. This guided refinement is however not mandatory and can be disabled.

Table 1 reports the average numbers of features used for matching and the average numbers of matches obtained for the three possible modes. Table 2 reports the success rate in pose calculation. Both the *software success rate* (i.e. when the software found a sufficient number of geometrically consistent matches) and the *visual success rate* (i.e. human control on whether the estimated pose is correct) are reported. Finally, Table 3 reports the average processing times obtained for the different modes.

The analysis of these results shows that, as expected, MM and GR tend to improve the number of (good) matches. However, in this experiment at least, the improvement is not critical, since it does not significantly

Table 1
Experiment 1: Statistics of the pose calculation performance for the three possible modes: *Initialization*, *Tracking* and *Reset*. The columns *Matching* and *GR* report the results obtained after the initial matching stage and the guided refinement stage respectively. The columns *DB* and *Match* report the number of map features used for matching and the number of matches achieved.

MM	*GR*	*Initialization*				*Tracking*				*Reset*			
		Matching		*GR*		*Matching*		*GR*		*Matching*		*GR*	
		DB	*Match*	*DB*	*Match*	*DB*	*Match*	*DB*	*Match*	*DB*	*Match*	*DB*	*Match*
No	No	2,675	114	N/A		1,500	72	N/A		6,000	85	N/A	
No	Yes	2,675	114	1,500	232	1,500	72	1,500	156	6,000	85	1,500	170
Yes	No	2,675	119	N/A		1,500	88	N/A		6,000	88	N/A	
Yes	Yes	2,675	119	1,500	286	1,500	88	1,500	217	6,000	88	1,500	214

Table 2
Experiment 1: Statistics of the pose calculation performance.

MM	*GR*	*Software Success Rate*			*Visual Success Rate*		
		Initial.	*Tracking*	*Reset*	*Initial.*	*Tracking*	*Reset*
No	No	100% (1/1)	95% (118/124)	100% (6/6)	100% (1/1)	100% (124/124)	100% (6/6)
No	Yes	100% (1/1)	97% (120/124)	100% (4/4)	100% (1/1)	100% (124/124)	100% (4/4)
Yes	No	100% (1/1)	97% (120/124)	100% (4/4)	100% (1/1)	100% (124/124)	100% (4/4)
Yes	Yes	100% (1/1)	98% (121/124)	100% (3/3)	100% (1/1)	100% (124/124)	100% (3/3)

Table 3
Experiment 1: Average computation times for pose calculation using training and target images having all 2048x1536 resolution.

MM	*GR*	*Mean Processing Time (s)*		
		Initial.	*Tracking*	*Reset*
No	No	1.34	2.16	1.18
No	Yes	2.00	2.81	1.81
Yes	No	1.37	2.17	1.18
Yes	Yes	1.99	2.81	1.81

Table 4
Experiment 1: Comparison of the pose estimation performance for different combinations of sizes of the training and target images. Small (S) images have the size 640x480, and large (L) images have size 2048x1536. The last four lines of this table correspond to Table 2.

Image Size		*MM*	*GR*	*Software Success Rate*			*Visual Success Rate*		
Training	*Target*			*Initial.*	*Tracking*	*Reset*	*Initial.*	*Tracking*	*Reset*
S	S	No	No	100%	97%	100%	100%	100%	100%
		No	Yes	100%	97%	100%	100%	100%	100%
		Yes	No	100%	97%	100%	100%	100%	100%
		Yes	Yes	100%	96%	100%	100%	100%	100%
S	L	No	No	100%	91%	43%	100%	88%	45%
		No	Yes	100%	88%	54%	100%	88%	54%
		Yes	No	100%	88%	70%	100%	88%	70%
		Yes	Yes	100%	95%	75%	100%	85%	75%
L	S	No	No	100%	98%	100%	100%	98%	100%
		No	Yes	100%	95%	100%	100%	99%	100%
		Yes	No	100%	94%	100%	100%	94%	88%
		Yes	Yes	100%	93%	100%	100%	94%	89%
L	L	No	No	100%	95%	100%	100%	100%	100%
		No	Yes	100%	97%	100%	100%	100%	100%
		Yes	No	100%	97%	100%	100%	100%	100%
		Yes	Yes	100%	98%	100%	100%	100%	100%

improve the pose estimation success rates. In addition, the use of GR significantly impacts the average processing time with an average increase of ~40%. In any case, Table 3 indicates that the current implementation of the proposed tracking system does not enable processing speeds that would support real-time applications (given the current parameters and large image resolutions). The issue of processing speed is investigated further below.

Table 4 compares the performance achieved by the system for training and target images with different resolutions: the original resolution 2048x1536 and the sub-sampled resolution 640x480. The table shows that the system clearly performs best when the training and target images have a similar size, which is generally not surprising. But, these results also show that small training and target images actually achieve similar pose estimation performance as large images, with the advantage of faster processing times (see Table 5). In the case of different training and target image resolutions, the results indicate that the scale-invariance property of SURF features can be put to the limit if the difference in image resolution is significant (given that the two datasets are acquired from the same distance to the scene). Nonetheless, one can also note that the performance seems better when the training images are larger than the target images, than when it is the opposite.

Table 5 presents computational times similarly to those in Table 3 but for training and target images with resolution 640x480. It appears that, although the number of image pixels is effectively reduced by a factor of 10, the computational efficiency does not improve that significantly. This is due to the fact that, although the computation of the SURF features for the target image is significantly sped up, the parameters S_{target}, $S_{frustum}$ and S_{sphere} remain unchanged, so that the number of matches calculated remains fairly constant.

Table 5
Experiment 1: Average computational times for pose calculation using training and target images having all resolution 640x480.

MM	*GR*	*Mean Processing Time (s)*		
		Initial.	*Tracking*	*Reset*
No	No	0.58	0.75	0.75
No	Yes	1.25	1.22	1.21
Yes	No	0.55	0.57	0.56
Yes	Yes	1.20	1.20	1.20

Finally, Table 6 presents computational times achieved with training and target images with resolution 640x480 and with $S_{target} = S_{frustum} = 500$ and $S_{sphere} = 2,000$. It appears that the processing times are further, but not that significantly, decreased. Note that, for this dataset at least, this computational improvement was not achieved at the cost of pose estimation quality. A visual analysis of the augmented target images showed that the pose estimations were all just as good as the ones obtained with higher values for S_{target}, $S_{frustum}$ and S_{sphere}, and images with higher resolutions.

Table 6
Experiment 1: Average computational times for pose calculation using training and target images having all resolution 640x480. Compared to Table 5, the following parameter values are changed: $S_{target} = S_{frustum} = 500$ and $S_{sphere} = 4S_{frustum} = 2,000$.

MM	*GR*	*Mean Processing Time (s)*		
		Initial.	*Tracking*	*Reset*
No	No	0.35	0.26	0.34
No	Yes	0.45	0.39	0.46
Yes	No	0.34	0.28	0.34
Yes	Yes	0.45	0.40	0.47

7.2 Further experiments

Figures 10 and 11 show experimental results obtained for two other datasets. The results shown in Figure 10 are obtained with the EPFL dataset presented in Section 4 and for which tracking with an existing coarse 3D CAD model had proven unsuccessful. This scene, which is much less structured than the one in the first experiment, is thus more challenging — a scene's *structural complexity* refers to geometric complexity, amount of uniform textures, and amount of geometrical/texture repetitions. The system nonetheless successfully augmented the subsequently acquired target video images.

Figure 11 shows results illustrating a case when a building is planned to be demolished and a new building is planned to be constructed. The structural complexity of this Edinburgh New Town dataset can be roughly estimated to lay between the complexities of the scenes of the first two experiments, mainly because it contains buildings with well textured façades, but also with numerous self-similarities that can impact the performance of the system.

Overall, all these results demonstrate the localization performance and potential of the proposed AR system.

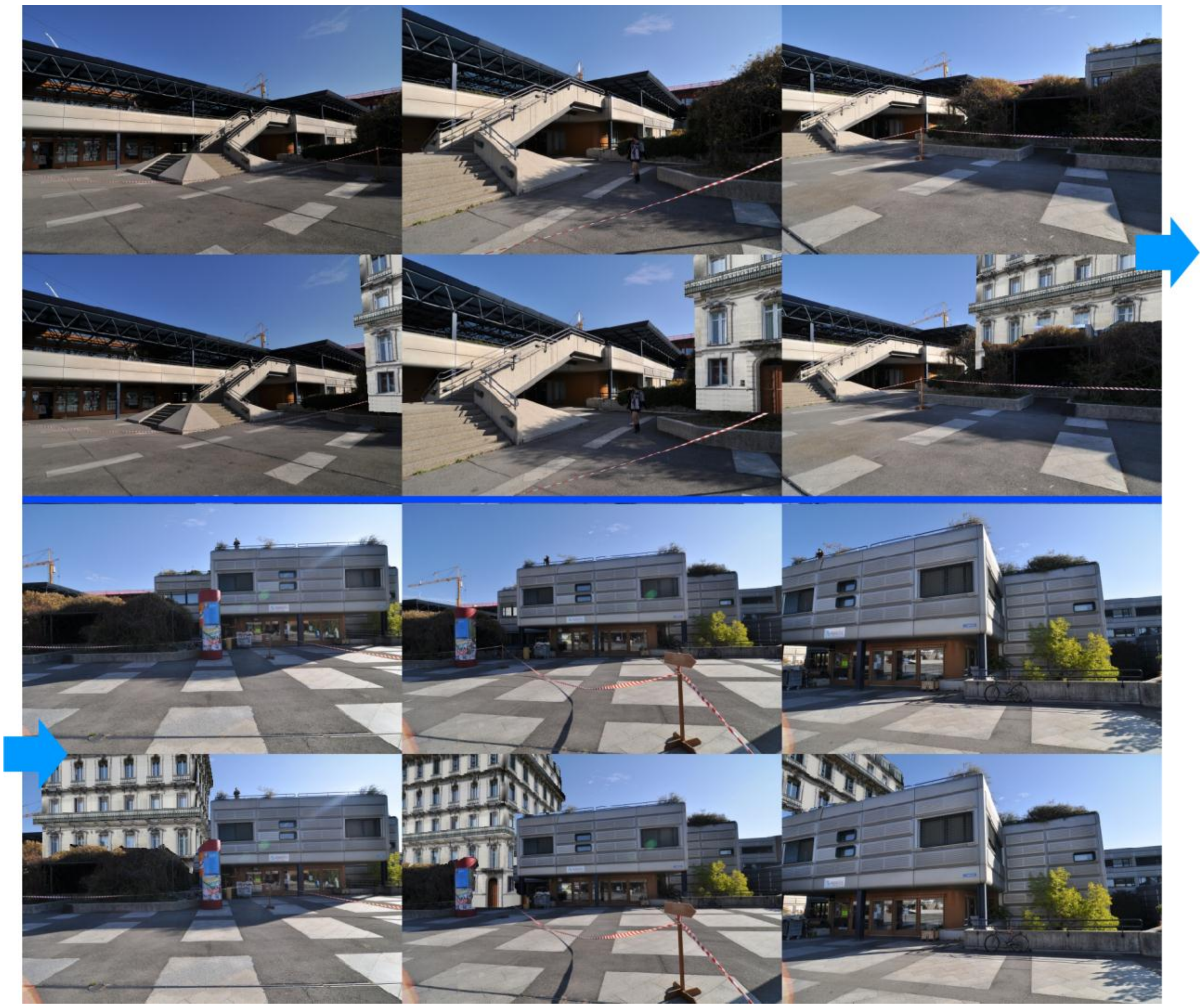


Figure 10 Experiment 2: Six of the processed target images of the EPFL dataset before (lines 1 and 3) and after being augmented (lines 2 and 4).

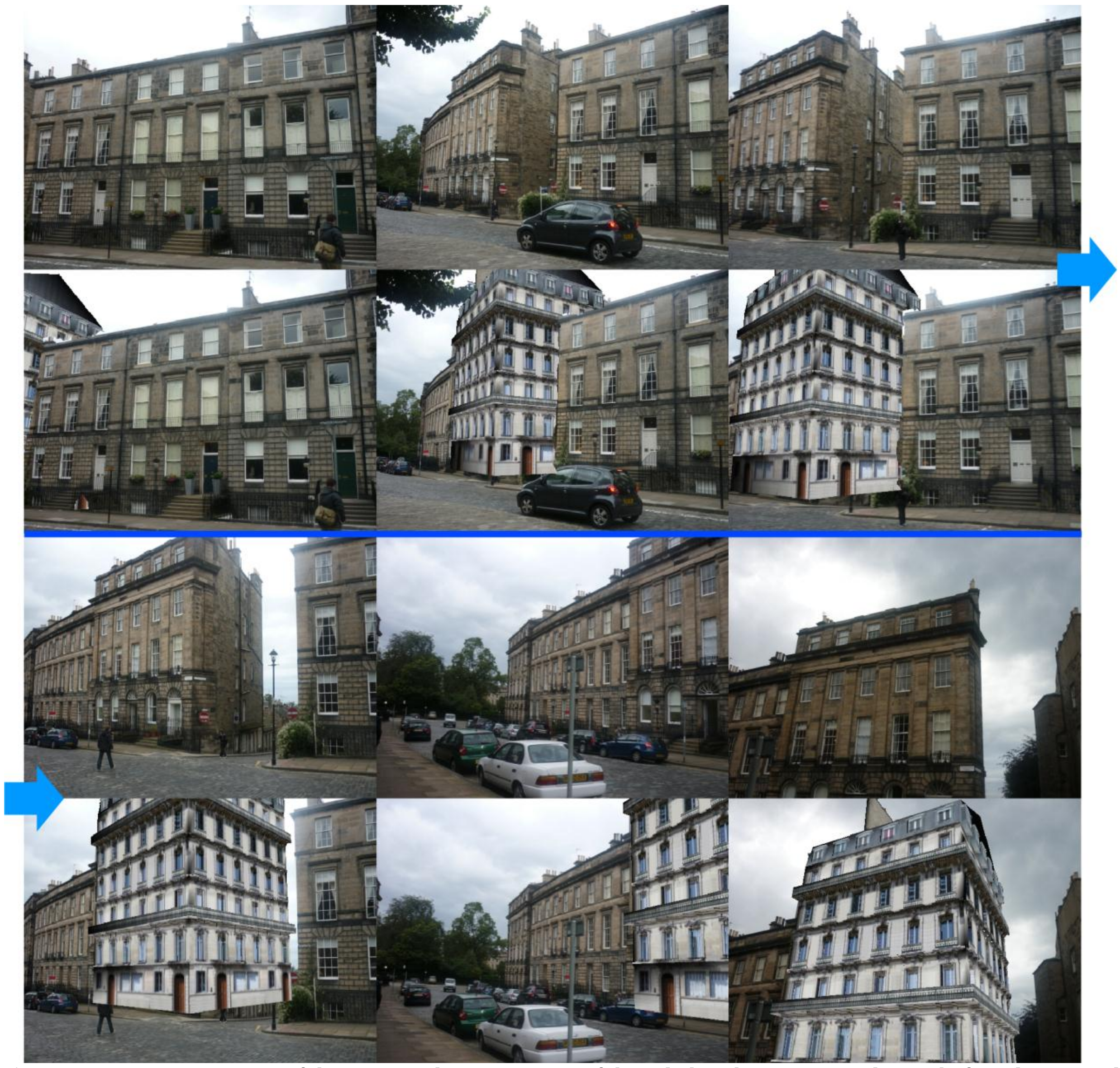


Figure 11 Experiment 3: Six of the processed target images of the Edinburgh New Town dataset before (lines 1 and 3) and after being augmented (lines 2 and 4).

8 CONCLUSIONS AND FUTURE WORK

A markerless monocular vision-based augmented reality system has been presented, with the aim of providing stakeholders of urban developments with a tool enabling them to assess planned constructions directly within their environment. The system has two stages: In a first offline stage a map of the urban scene is constructed comprising a database of 3D-referenced SURF features and a dense mesh reconstruction. The latter is used offline to augment the scene with the virtual objects, and online to calculate occlusions of those virtual objects by the existing environment. In the second, online stage, target stream images are processed. For each target image, the camera pose is computed through SURF matching and a robust pose estimation procedure combining 3-point, RANSAC, and Levenberg-Marquardt non-linear optimization algorithms.

The performance of the system was successfully demonstrated on real imagery from three different scenes. Nonetheless, improvements could be made in several areas:

- The system in its current implementation only achieves up to 3fps which is not fast enough to consider a real-time AR system. While some improvement could be achieved by varying some parameters (e.g. S_{target}, $S_{frustum}$, S_{sphere}), transferring some data processing to a GPU is also of interest here because our feature matching processes (the most time-consuming part) are well suited for parallel processing. Nonetheless, as shown in this paper, the system may already be used in an "offline" manner by augmenting a recorded video.

- Compared to other commonly used approaches, such as the one proposed by Woodward et al. (2010), our tracking strategy does not rely on tracking image features (e.g. using the Kanade-Lucas-Tomasi Feature Tracker (KLT) (Shi and Tomasi, 1994)). Instead, it tracks the camera. While camera tracking may be more robust with respect to sharp changes in camera motion (through the *Reset* mode implemented here), the overall tracking is likely not as efficient (database culling must be performed and a fairly large amount of matching has to be conducted at each iteration). A hybrid system could thus be envisaged.
- Similarly, our system does not rely on beacon-based positioning or inertial sensors. While using a single source of data (images) makes our system much simpler, it could nonetheless benefit from such sensory data, especially to support the *Initalization* or *Reset* modes.
- Further culling of the database features may also be achieved by considering some feature visibility criterion, as suggested by Wolf et al. (2005).
- While the system is designed to handle scenes of the size of a urban neighborhood, further testing needs to be conducted using larger scenes.
- Since the focus of this work is on urban augmented reality, positioning techniques based on planar structures, as suggested by Simon et al. (2000), could be investigated.

In addition, it must be emphasized that there is one limitation that is inherent to all vision-based localization approaches, which is that they perform adequately only when the scene is sufficiently structured. While urban environments are generally sufficiently structured (as shown in the experiments), other environments can be much more challenging (e.g. for some "greenfield" developments).

Finally, an interesting area of further research, that could potentially address some of the limitations of the current system, is the use of textured GIS level 2 (and 3) urban 3D models – e.g. city models encoded in CityGML (Open Geospatial Consortium, 2012). There is an increasing public and commercial interest for such models, that are now being developed around the world. The advantage of such models with regard to the proposed system is that scenes' maps of 3D-referenced features could be built by extracting features directly from the textured models, which would simplify the learning stage. In addition, the models could be directly used for calculating occlusions.

ACKNOWLEDGMENTS

The authors would like to acknowledge the support of the Swiss National Science Foundation (SNSF) with the project Aerial Crowds #CRSI20 122696, as well as the European Space Agency (ESA) with the project Wear++ #. Ludovico Carozza would like to thank ARCES - University of Bologna for having supported his research period at ETH, Zurich, through the "Marco Polo" scholarship.

REFERENCES

Azuma, R., Baillot, Y., Behringer, R., Feiner, S., Julier, S. & MacIntyre, B. (2001), Recent advances in augmented reality, *Computer Graphics and Applications, IEEE*, **21**(6), 34-47.

Bay, H., Tuytelaars, T. & Van Gool, L. (2006), SURF: Speeded Up Robust Features, *Lecture Notes in Computer Science, Proceedings of the European Conference on Computer Vision (ECCV)*, **3951**, 404-417.

Behzadan, A.H. & Kamat, V.R. (2010), Scalable algorithm for resolving incorrect occlusion in dynamic augmented reality engineering environments, *Computer-Aided Civil and Infrastructure Engineering*, **25**, 3-19.

Beis, J. & Lowe, D.G. (1997), Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Puerto Rico, pp. 1000-1006.

Bishop, G. & Welch, G. (2001), An introduction to the Kalman Filter, *SIGGRAPH, Course 8*, University of North Carolina at Chapel Hill, USA.

Capp, O., Godsill, S.J. & Moulines, E. (2007), An overview of existing methods and recent advances in sequential Monte Carlo, *IEEE Proceedings*. **95**, 899-924.

Comport, A.I., Marchand, E., Pressigout, M. & Chaumette, F.(2006), Real-time markerless tracking for augmented reality: The virtual visual servoing framework, *IEEE Transactions on Visualization and Computer Graphics*, **12**(4), 615-628.

De Filippi, F. & Balbo, R. (2011), Planning for real: ICT as a tool in urban regeneration, *The Built & Human Environment Review*, **4**, 67-73.

Durrant-Whyte, H. & Bailey, T. (2006), Simultaneous localization and mapping (SLAM): Part I the essential algorithms, *Robotics and Automation Magazine*, **13**(2), 99-110.

Fischler, M.A. & Bolles, R.C. (1981), Random sample and consensus: A paradigm for model fitting with application to image analysis and automated cartography. *Communications of the ACM*, **24**, 381-395.

Gordon, I. & Lowe, D.G. (2004), Scene modelling, recognition and tracking with invariant image features, *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, Arlington, VA, USA, pp. 110-119.

Haralick, R.M., Lee, C.N., Ottenberg, K. & Nolle, M. (1994), Review and analysis of the three point perspective pose estimation problem, *International Journal of Computer Vision*, **13**, 331-356.

Inam,W. (2009), Particle filter based self-localization using visual landmarks and image database, *IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, Daejeon, South Korea, pp. 246-251.

Karlekar, J., Zhou, S.Z., Lu, W., Loh, Z.C. & Nakayama, Y. (2010), Positioning, tracking and mapping for outdoor augmentation, *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*. Seoul, Korea, pp. 175-184.

Klein, G. & Murray, D. (2007), Parallel tracking and mapping for small AR workspaces. *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, Nara, Japan.

Leonard, J.J. & Durrant-Whyte, H. (1991), Mobile robot localization by tracking geometric beacons, *IEEE Transactions on Robotics and Automation*, **7**(3), 376-382.

Neumann, U., You, S., Cho, Y., Lee, J. & Park, J. (1999), Augmented reality tracking in natural environments, *IEEE International Symposium on Mixed Realities (ISMR)*.

Newcombe, R.A. & Davison, A.J. (2010), Live dense reconstruction with a single moving camera. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, USA, pp. 1498-1505.

Nocedal, J. & Wright, S.J. (2006), *Numerical Optimization*, 2nd Edition, Springer.

Open Geospatial Consortium (2012), *City Geography Markup Language (CityGML) Encoding Standard*, available at https://portal.opengeospatial.org/files/?artifact id=47842.

Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J. & Koch, R. (2004), Visual modeling with a hand-held camera, *International Journal of Computer Vision*, **95**(3), 207-232.

Reitmayr, G. & Drummond, T.W. (2006), Going out: Robust tracking for outdoor augmented reality, *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, Santa Barbara, CA, USA, pp. 109-118.

Saeki, K., Tanaka, K. & Ueda, T. (2009), LSH-RANSAC: An incremental scheme for scalable localization, *IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, pp. 3523-3530.

Sanghoon, L. & Omer, A. (2011), Augmented reality-based computational fieldwork support for equipment operations and maintenance, *Automation in Construction*, **20**(4), 338-352.

Shi, J. & Tomasi, C. (1994), Good features to track, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, pp. 593-600.

Shin, D.H. & Dunston, P.S. (2009), Evaluation of augmented reality in steel column inspection, *Automation in Construction*, **18**(2), 118-129.

Simon, G., Fitzgibbon, A. & Zisserman, A. (2000), Markerless tracking using planar structures in the scene, *IEEE and ACM International Symposium on Augmented Reality (ISAR)*, pp. 120-128.

Tingdahl, D. & Van Gool, L. (2011), A public system for image based 3D model generation, *Lecture Notes in Computer Science. Computer Vision/Computer Graphics Collaboration Techniques 5th International Conference, MIRAGE 2011*, **6930**, pp. 262-273.

Wolf, J., Burgard, W. & Burkhardt, H. (2005), Robust vision-based localization by combining an image retrieval system with monte carlo localization, *IEEE Transactions in Robotics*, **21**(2), 208-216.

Woodward, C., Hakkarainen,M., Korkalo, O., Kantonen, T., Aittala, M., Rainio, K. & Kähkönen, K. (2010), Mixed reality for mobile construction site visualization and communication, *10th International Conference on Construction Applications of Virtual Reality (CONVR2010)*, Sendai, Miyagi, Japan, pp. 35-44.

Yakubi, N., Miyashita, K. & Fukuda, T. (2011), An invisible height evaluation system for building height regulation to preserve good landscapes using augmented reality, *Automation in Construction*, **20**, 228-235.

Zhang, C. & Elaksher, A. (2012), An Unmanned Aerial Vehicle-Based Imaging System for 3D Measurement of Unpaved Road Surface Distresses, *Computer-Aided Civil and Infrastructure Engineering*, **27**(2), 118-129.