The Visual Computer manuscript No.
(will be inserted by the editor)

Gaël Sourimant · Luce Morin · Kadi Bouatouch

Gps, Gis and Video fusion for urban modeling

Abstract 3D reconstruction of urban environments is a widely studied subject since several years, since it can lead to many useful applications: virtual navigation, augmented reality, architectural planning, etc. One of the most difficult problem nowadays in this context is the acquisition and treatment of data if very large scale and precise reconstruction is aimed. In this paper we present a system for computing geo-referenced positions and orientations if images of buildings from non calibrated videos. Providing such information is a mandatory step to well conditioned large scale and precise 3D reconstruction of urban areas. Our method is based on the fusion of multimodal datasets, namely GPS measures, video sequences and rough 3D models of buildings.

Keywords City modeling · image-based modeling · geo-localization · virtual reality

1 Introduction

The recent success of google-earth has shown that adding photo-realistic texture on a 2D map adds a lot of sense for the user compared with a traditional synthetic and symbolic 2D map. The 3D functionalities offered by this popular tool, such as 3D navigation and 3D modeling of buildings, are also reasons for its success. However, the provided 3D models of buildings show little realism (they are gray parallelepipeds). As for the aerial pictures superimposed on the 2D maps, one would like 3D models of buildings to provide a feeling of reality, as in a picture taken from an airplane or from the ground. 3D modeling of urban environments has many other applications such as games, virtual tourism, geopositioning or virtual reality. Unfortunately, manual modeling by an expert graphic designer is a tedious and expensive process which can not be extended to large-scale urban areas.

In this paper, we present a system for computing geo-referenced positions and orientations of images of building. Our approach is based on the fusion of multimodal datasets, namely ground based video images together with GIS data composed of a set of buildings, each being described by its ground footprint and its elevation. This type of GIS data exists in a large scale for any large city in France[1]. However, such modeling performs well for aerial visualization, but it is not visually satisfying for ground navigation in the 3D model. Video and GIS data give complementary information: video provides photorealism, geometrical details, precision in the fronto-parallel axes ; GIS provides a "clean" and complete geometry of the scene, structured into individual buildings. We also add a GPS acquisition synchronized with video acquisition. In order to combine these three types of data, the first step is to register the data in the same coordinate system. The registration step is indeed a bottleneck in the whole modeling system, as it requires geometric correspondences between very non similar data contents.

The remaining of the paper is organized as follows. Section 2 presents related works on automatic, or semi-automatic city modeling, based either on aerial or ground imaging, or even both. Section 3 gives an overview of our approach and the particular datatypes we use. The whole registration procedure is explained in 4 and results on geo-localization of images are presented in 5, leading to the conclusion in section 6

Gaël Sourimant
IRISA/INRIA
Tel.: +33 2.99.84.72.72
Fax: +33 2.99.84.71.71
E-mail: gael.sourimant@irisa.fr

Luce Morin
IRISA/Université Rennes1
Tel.: +33 2.99.84.73.45
Fax: +33 2.99.84.71.71
E-mail: luce.morin@irisa.fr

Kadi Bouatouch
IRISA/Université Rennes1
Tel.: +33 2.99.84.72.58
Fax: +33 2.99.84.71.71
E-mail: kadi.bouatouch@irisa.fr

[1] 3D visualization of french cities will be publicly available in 2007 according to the French National Geographic Institute (IGN)

where we discuss the limitations of our framework and leads for future works.

2 Related works and proposed approach

Several studies have addressed automatic or semi automatic 3D modeling of urban environments from acquired data. Depending on the desired level of detail and accuracy of the resulting models, different types of dataset have been used. For large scale and rough reconstruction, aerial and satellite images are used, whereas for precise but more narrow range reconstruction one could use ground pictures or video sequences. A third type of approach combines both these aerial- and ground-based approaches to perform accurate and large scale city reconstruction.

Aerial-based city reconstruction. Prior work on urban modeling often imply modeling from aerial and satelite imaging. These methods are generally decomposed into two steps. First, buildings are detected using segmentation or lines extraction / connection algorithms [6,10,21], and then building models are effectively computed using standard stereovision [6] or registration between the images and simple 3D primitive models fused in a CSG tree structure [18]. The resulting models give a good prior of the city shape, and can be used in frameworks where the goal is to enhance them using ground imaging. It has been used in previous studies together with automatic texture generation from semantic information (date of building construction, number of stages) to automatically recover 3D models of entire cities [13]. These approaches generally provide however rough 3D models of buildings, in the sense that though global shape is well estimated (dimensions, angles, etc.) no façade textures nor geometric details can be extracted from this kind of images.

Ground based city reconstruction. Ground based reconstruction refers to 3D modeling of buildings using data acquired within urban areas : still images or video sequences. This kind of approach enables the retrieval of photometric information (buildings textures, etc.) as well as local geometry (positions and shapes of doors, windows, etc.).
Methods for modeling 3D scenes from sets of images have been widely studied in computer vision, either in *Structure from Motion* (SfM), which allows to retrieve camera calibration, pose estimation and 3D scene geometry [9,11], or in *Image-based Modeling* (IbM) which is the process of creating 3D models directly from a collection of input images. An example of IbM framework is the semi-automatic Façade system [3] which was used to model parts of the UC Berkeley campus. Several works address geometric retrieval in local areas [22,15], whereas other projects aim at large scale urban modeling from ground images. In the MIT City Scanning Project [19], calibrated hemispherical images of buildings are used to extract planes corresponding to façades, which are then textured and geometrically refined using pattern matching and computer vision methods. In the UrbanScape project [1], a fully automated system for accurate and rapid 3D reconstruction of urban environments from video streams is designed, one of its goals being real-time reconstruction using both the CPU and the GPU. The 4D Cities project [14,12] aims at creating time-varying 3D models from a large collection of images taken from different locations, at very different times.
The main drawback is that it remains a local approach, requiring a great level of effort both on acquisition and on data analysis in the case of large scale reconstruction.

Taking advantage of both approaches. Few studies take advantage of both aerial and ground reconstruction. An interesting approach is the one by Frueh's and Zakhor's [7], where they jointly use camera and laser acquired data for city modeling. A vertical laser measures a dense point cloud of the façades while a video camera is used for texturing. An horizontal laser camera is used to estimate the acquisition position and is registered with aerial edge maps of the buildings to ensure global consistency of the different 3D models computed. The main drawbacks of such procedure is the complexity of the acquisition system together with a long and tedious post processing of the data, *i.e.* mainly to treat the massive number of measured 3D points.

Proposed approach. We solve the problem in a coarse-to-fine way, in the sense that we start from existing rough and non textured 3D models of the buildings (which are expressed in a geo-referenced coordinate frame), and add gradually information from video sequences, thanks to computer vision- and robotics-based algorithms. Such an approach allows to compute the corresponding real geometric details and textures that will enhance the coarse original 3D models.
Contrary to aerial-based solutions which only provide a rough structure of buildings shape, our framework starts with such models and aims at refining their geometry by adding photometric and geometric details thanks to video images. By using geo-referenced rough 3D models of the buildings extracted from a GIS database, our approach permits to build more easily large scale city models than classical ground-based methods, the initial modeling providing a useful global consistency between the different models. Finally, our approach uses these ground data (video images) and aerial data (GIS models) to perform a consistent global reconstruction with a more simple acquisition system (only a handheld video camera and a GPS receiver) and moreover an simpler post-processing procedure than methods described in [7].

3 Data types and system overview

3.1 Data types

In this section, we review some information on the different datasets used in the proposed reconstruction framework,

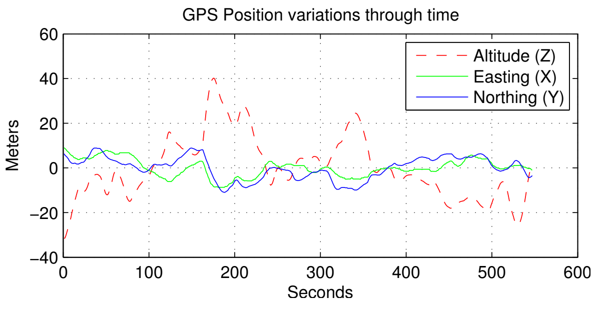


Fig. 1 GPS position measures in meters *vs.* time, for a fixed point

in order to provide a good understanding basis for the next sections. The datasets on which is based our method are the following: GIS databases which give the original geo-referenced 3D models of the buildings, videos from which we extract RGB images for texturing and luminance information for features extraction/tracking, and finally GPS measures that are recorded simultaneously with images, which will provide a first approximation for geo-localizing these images. We remind here some particularities of GPS and GIS.

Gps. The GPS (*Global Positioning System*) gives position measures with limited accuracy. Satellite signals can be reflected off walls, mountains, and slowed down by atmospheric conditions, etc., leading in a five meters precision in 95% of the time. In order to estimate the error variation of GPS measures through time, an acquisition was made at the exact same spot, in poor recording conditions (just next to high buildings), during approximately 10 minutes. Figure 1 shows the position variations, decomposed into easting (X), northing (Y) and altitude (Z) in the standard geographic UTM coordinate system. Values are centered at their mean for variation comparison purpose. As we can see, standard deviation in altitude is much higher ($\sigma_Z = 14.02m$), and then less reliable than variation in the horizontal plane ($\sigma_X = 3.92m$, $\sigma_Y = 5.05m$). GPS data can thus only provide an initial estimate of the camera path with limited accuracy.

Gis. The GIS acronym, standing for *Geographic Information System*, refers to a collection of any form of geographically referenced information. In our system, we use a database where each building is described by its altitude, its height, and its footprint expressed as a closed list of 2D points, which XY coordinates are expressed in the UTM coordinate system. This database provides a coarse estimation of the scene geometry, the buildings being modeled by simple polyhedrons (see figure 2: the left part represents footprints from top view ; the right part shows the resulting rough 3D models computed from the buildings footprints and elevations). One may already see the drawbacks of such building models: they are geometrically poor (no façade details, no roof modeling), photometrically null (they do not provide any information about the building textures), and moreover can be erroneous (bad height estimation, lack of sub-parts of buildings, etc.). This is the reason why we introduce video data to enhance those models.

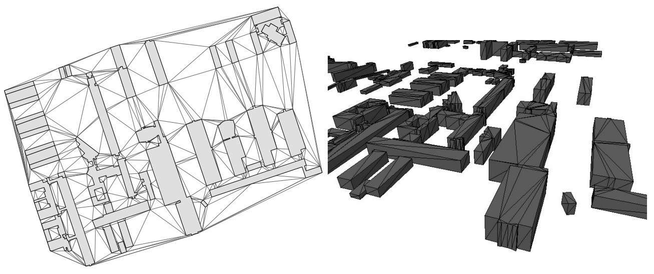


Fig. 2 GIS representation used as an initial buildings modeling

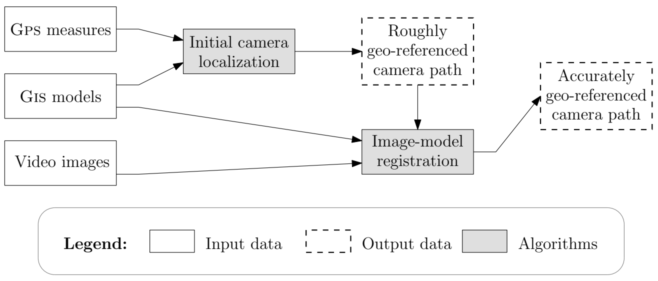

GPS measures
GIS models
Video images
Initial camera localization
Roughly geo-referenced camera path
Image-model registration
Accurately geo-referenced camera path
Legend:
Input data
Output data
Algorithms

Fig. 3 High-level principle of our method

3.2 System overview

The datasets fusion principle is outlined on figure 3. The first step of our framework, outlined on this figure 3 by the box labeled *Initial camera localization* (see section 4.1), consists of using GPS data together with the GIS database so as to find a first approximation of the camera location with regard to the buildings. As an output, we associate rough camera position and orientation with each image of the video sequence. Since we only need at this point an approximate measure of the camera pose with regard to the surrounding buildings, a few meters error on the position estimate does not cause any nuisance.
The next step, outlined on figure 3 by the box labeled *Image-model registration* (see section 4.2), consists in relating images and 3D model primitives so as to get as an output accurate poses of the camera, for each image in the video. The camera pose is initialized with the estimated positions given by the GPS measures, and the final step of our system consists in registering the projection of the model onto the images by modifying the position and orientation of the virtual camera. Many algorithms for image-model registration already exist in the literature. The one we present here has the particularity to be adapted to urban reconstruction, contrary to state-of-the art methods.
Once both registrations are performed, the geo-localization of the real camera is deduced from the newly computed pose of the virtual camera. It is these poses together with the projected simple models on the images that will provide a well

conditioned front-end to accurate geometry computation and texture extraction.

4 Image-model registration

We have seen that the satellite position measures will only lead to an initial estimation of the position of the camera. This estimate will be used as a guess for the computation of a more accurate camera pose, as will be shown in 4.2.

4.1 Initial camera localization based on Gps

Registering GPS measures and the GIS database aims at providing the camera position with respect to the buildings coordinate system. The different problems we have to cope with to get this initial estimation are mainly the difference between GPS and GIS coordinate systems, camera orientation estimation, and camera altitude estimation.

Geographic to UTM conversion. The very first task we have addressed while registering the GIS database and the GPS positions is to express both of them in the same coordinate system. On one hand the GPS position measures are expressed with latitude/longitude couples, while on the other hand the point positions in the GIS database are given in meters, in a UTM coordinate system. The conversion between one system to the other can be quite complex, depending on the user's position on the globe, the precise shape of Earth, etc. Equations converting coordinates from one system to the other are precisely described in [17].

Camera position. As seen in section 3.1, even if the horizontal precision can give a good estimate of the camera locations with regard to the GIS database, vertical estimation of the position is untrustworthy, which may lead to an unusable initial projected model with regard to the images. To cope with this problem, we chose to simply drop the vertical GPS measured position and use instead a ground estimation by putting the camera about one meter and a half above it. To compute the height estimate of a given point within the GIS database, we use a Delaunay triangulation in which each building ground corner acts as a triangle vertex, as it is illustrated on figure 2 (left part). Ground height is then computed as the intersection of this generated surface and the 3D vertical line intersecting the XY plane at the position given by the GPS measure, in UTM coordinates.

Camera orientation. The camera orientation has to be estimated for each frame, but absolutely no information about it is provided by the acquired data, because the GPS receiver only records a position and an altitude, and we made the choice of a simply usable system. Since it is more likely (because more natural) that the camera motion would be forward rather than backward, we initialize the orientation for a measure at time t with the vector $(p_{t+1} - p_t)$, p_t being the measured position for the measure at time t.

4.2 Registering Gis and Video

The use of GPS data has provided a rough estimate of camera parameters (position and orientation). To be accurate enough for data fusion, this first estimate has to be refined using video data. Registration of video and GIS consists in finding camera parameters expressed in the GIS coordinate system, for each video frame. If correctly registered, the GIS superimposes with buildings in the video frame when the GIS 3D models are rendered using the obtained camera parameters. Registration is decomposed into two steps. First, a semi-automatic process performs registration between the 3D model and the first image of the video sequence. Then, aligning the projections of the model on the following images amounts to a tracking problem. The following presents the theoretical background used for model-image registration and then describe more precisely the initial and tracking steps.

Theoretical background: camera model. The pinhole camera model is used to model the camera. With the hypothesis of corrected input images with regard to radial distortion, the homogeneous 2D projection $\mathbf{p}$ of an homogeneous 3D point $\mathbf{P}$ is given by the following equation :

$$\mathbf{p} = \mathbf{K}.{}^c\mathbf{M}_o.\mathbf{P} \tag{1}$$

$$\text{with}\,\mathbf{K} = \begin{bmatrix} \frac{f}{p_x} & 0 & u_0 \\ 0 & \frac{f}{p_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \text{and}\,{}^c\mathbf{M}_o = \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix}$$

where p_x and p_y are the width and height of the pixels, $[u_0\, v_0]^\top$ are the image coordinates of the principal point, and f is the focal length. Thus f_x and f_y are the focal length measured in width and height of the pixels. The camera pose ${}^c\mathbf{M}_o$ is defined by the camera 3×3 orientation matrix $\mathbf{R}$ and the 3×1 position vector $\mathbf{t}$.

Theoretical background: Visual Virtual Servoing. Aligning a projection of a 3D object on an image of the same object has been widely studied in the context of Computer Vision and Robotics. The most popular algorithms are those of Dementhon, namely POSIT [5] and SoftPOSIT [4]. Comport *et al.* also proposed relevant approaches based on visual servoing [2], which aims at searching for the pose of the viewed object (in images) by modifying the pose of the virtual (3D model) object. Our solution to compute the pose of the camera and register the GIS-based 3D models to the images is based on such a virtual visual servoing approach.
Pose computation is considered here as a global non-linear problem, consisting in relating a set of 3D primitives to their corresponding 2D projections in the image plane. The goal is to minimize the error between observed data $\mathbf{s}^*$ (in the images), and the position of the same information $\mathbf{s}$, computed by projection of the observed primitives onto the image plane. The camera pose ${}^c\mathbf{M}_o$ is thus estimated as:

$${}^c\tilde{\mathbf{M}}_o = \operatorname{argmin}(\|\mathbf{s}({}^c\mathbf{M}_o) - \mathbf{s}^*\|^2) \tag{2}$$

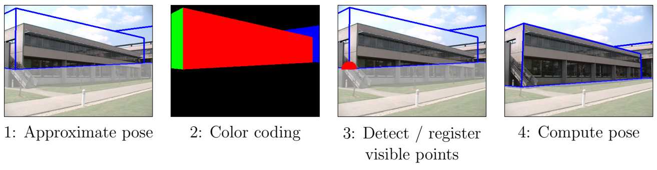


Fig. 4 Compute pose for the first image

This is solved in an iterative process, the pose being initialized to ${}^{ci}\mathbf{M}_o$, and updated until convergence to ${}^{cf}\mathbf{M}_o$ thanks to the control law:

$$v = -\lambda(\mathbf{L_s})^{+}(\mathbf{s}({}^{c}\mathbf{M}_o) - \mathbf{s}^*) \tag{3}$$

v being a pose vector defined by $\mathbf{R}$ and $\mathbf{t}$, λ a scalar and $\mathbf{L_s}$ the Jacobian of the minimization function. This method is generic regarding the primitive types, provided that the projection errors can be computed from image data. In our case, we use 2D interest points, so $\mathbf{s}^*$ represents a set of 2D points $\mathbf{p}_i$, and $\mathbf{s}$ is the set of corresponding projected 3D points $\mathbf{P}_i$, for a given pose ${}^{c}\mathbf{M}_o$ and a given internal parameters matrix $\mathbf{K}$. If N is the number of points in consideration, we have $\mathbf{s}^* = \{\mathbf{p}_i | i \in 1 \dots N\}$ and $\mathbf{s} = \{\mathbf{K}.{}^{c}\mathbf{M}_o\mathbf{P}_i | i \in 1 \dots N\}$. Finally, we see that if we can provide correspondences between 2D image points and 3D model points on the GIS database, we are able to compute the pose for the current image expressed in the GIS coordinate system.
Position accuracy computed thanks to a virtual visual servoing approach is very sensitive to errors introduced either by primitives extraction (noise in images, illumination variations, occlusions, 3D measure of model points, ...) or by primitives misregistration. The solution we use to ensure robustness of the control law is to introduce directly in it M-estimators, which allow to quantify a confidence measure in each visual information we use. The new control law is then:

$$v = -\lambda(\mathbf{D}\mathbf{L_s})^{+}\mathbf{D}(\mathbf{s}({}^{c}\mathbf{M}_o) - \mathbf{s}^*) \tag{4}$$

where $\mathbf{D} = diag(w_1, w_2, \dots, w_N)$ is a diagonal matrix holding the weights w_i corresponding to the confidence we have in each visual information. They are computed using the Cauchy robust function. Finally, to ensure that a sufficient number of visual information would not be rejected by the robust estimator, we check that the matrix $\mathbf{D}\mathbf{L_s}$ is of full rank (*i.e.* rank 6 since the pose has 6 degrees of freedom: 3 for translation and 3 for orientation), using a SVD decomposition.

Pose computation for the first image. We describe here the semi-automatic method for registering the GIS with the first frame of the video. The successive steps of the complete procedure are illustrated on figure 4.
At this point, only a rough position and orientation of the camera are available for this frame. The user is first asked to correct these values thanks to an OpenGL interface, showing both the image and the GIS 3D buildings. The latter is rendered in wireframe mode with a virtual camera. The user translates and rotates the virtual camera manually so that the projected GIS is visually similar to the image content.

This initial camera pose is refined using 2D-3D correspondences. The only 3D points which can be reliably extracted from the GIS database are the buildings corners (*i.e.* the bottom and roof points belonging to the building's footprint). Buildings corners which are visible in the rendered wireframe are automatically detected using the following color coding procedure. A polygonal version of the GIS database is stored in the OpenGL graphics memory, each façade being assigned a unique RGB color :

$$R = b_i \div 256 \quad G = b_i \, mod \, 256 \quad B = f_i$$

where b_i is the building's index and f_i is the façade's index within the building. The black color represents the 'no building' information. This color model is rendered in the OpenGL back buffer with the current approximate pose. Reading the back buffer once allows to identify which façades and buildings are currently viewed, and the couple (b_i, f_i) allows then direct access in the GIS database to the 3D coordinates of the façade corner points. Corner points projecting outside the image or occluded by another façade are discarded.
For each selected 3D point X_i the interface displays a marker in the GIS model, and waits for the user to click on the corresponding image point x_i. Once all 2D-3D correspondences have been given, pose is computed using a virtual visual servoing algorithm thanks to equation 3. Four 2D-3D correspondences at least are needed to perform the registration, the result being better in case of non coplanar points.

Tracking. Once pose has been computed for the first image of the video, registering GIS and images becomes a tracking problem, and as such, we treat it in a fully automatic way, using a similar virtual visual servoing approach. Let I_t be an image for which registration with the 3D model has been computed, and I_{t+1} be the following image for which we are trying to compute the pose. As usual, we need for this image I_{t+1} correspondences between 2D and 3D points. This is done in a *point transfer* scheme, using data extracted from I_t. The complete tracking procedure is summarized in algorithm 1 and illustrated on figure 6.
First of all, 2D points are extracted from image I_t. For feature extraction and tracking, we use an implementation of Kanade-Lucas-Tomasi (KLT) feature tracker[2] [20]. Because all extracted points may not belong to a building, they have to be classified into on- and off-building points. No explicit depth estimation is performed to check whether the 2D extracted features intersect the GIS model. Instead they are assigned to their corresponding z-buffer value, which is computed by OpenGL to display the 3D model registered to the image (see figure 6(a)). If this value is zero, then the point is considered as an off-building point, and vice versa. We have then at this point correspondences between 2D and 3D points, for image I_t, *which is already registered* with the GIS model. One can already see that we are not limited here to use buildings corners as 3D information, since image model-registration gives potentially depth information

[2] http://www.ces.clemson.edu/~stb/klt/

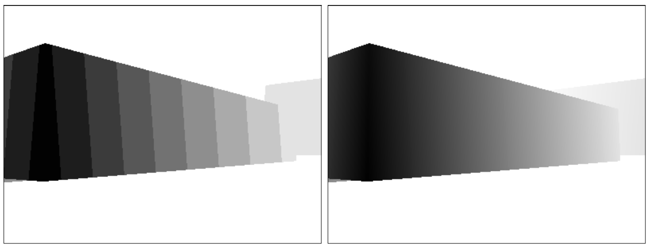

(a) $\pi_n = 1$ and $\pi_f = 75$ (b) $\pi_n = 20$ and $\pi_f = 75$

Fig. 5 Influence of clipping planes positions on depth estimation

for each pixel lying in the model projection. Actually, for low-resolution images (400×300 pixels here), one can often expect to find about 100 or 150 features.

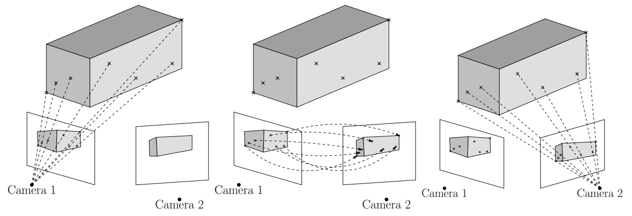
Because the estimation is generally unstable since features often lie on a single façade plane, the ground estimate (see section 4.1) is used to introduce new 2D-3D correspondences which are globally on a plane orthogonal to the façade planes.

Moreover, we take into consideration the way OpenGL stores z-buffer values to get more accurate measures for the 3D points. In our case, few precision is generally provided to the façcade points if we use standard clipping planes values. To prevent this, we let the user define the far clipping plane value π_f as a parameter but we move the near one π_n to rendered building point which is the closest to the camera. The depth value $z(p)$ for a feature point p is then computed from the corresponding z-buffer value $z'(p)$ using the mapping function described in equation 5.

$$z = (\pi_f \pi_n)/(\pi_f - z'(\pi_f - \pi_n)) \tag{5}$$

Figure 5 provides a comparison between stored z-buffer values for standard clipping planes values (left column) and adapted ones (right column). As we can see, the further π_n is from the camera center, the higher is the precision for points far from it. On the figure, the closest point is approximately at 23 meters away from the camera.

Using the KLT, we track the 2D features from image I_t to I_{t+1} (see figure 6(b)). If $\mathbf{x}_t$ represents the 2D points extracted from I_t and $\mathbf{X}$ their corresponding 3D position, since we have correspondences between $\mathbf{x}_t$ and $\mathbf{x}_{t+1}$ we can deduce 2D-3D correspondences for I_{t+1}, between $\mathbf{x}_{t+1}$ and $\mathbf{X}$. Using them into equation 4 permits to compute the camera pose for I_{t+1} (figure 6(c)). The process is repeated until pose has been computed for all images in the video. However, the KLT tracker looses points throughout the registration process. To cope with this, we introduce a measure criterion on the lost points. If we loose a certain percentage of points (typically 60%), we extract new interest features and read the corresponding z-buffer values, for the last registered image. We keep however the points we did not lose, and constrain the new points to be far enough in the image from the old ones.



(a) (b) (c)

Fig. 6 Tracking pose throughout the video

Algorithm 1 Track pose throughout the video

```
Require: Pose has been estimated for the first image $I_0$
 1: Extract feature points $\mathbf{x}$ from image $I_0$
 2: Store number of feature points $n$
 3: Read z-buffer $\mathbf{Z}$ and find the closest point to camera
 4: Place near plane $\pi_n$ at this point
 5: Read $\mathbf{Z}$ again
 6: for all $x_i \in \mathbf{x}$ do
 7:     if corresponding z-buffer value $z_i = 1$ then
 8:         Remove $x_i$ from $\mathbf{x}$
 9:     else $\{x_i$ on a building$\}$
10:         Deduce 3D position $X_i$ of $x_i$ from $z_i$
11:         Store $X_i$ into the list $\mathbf{X}$
12:     end if
13: end for
14: for all $x_i \in \mathbf{x}$ do
15:     Track $x_i$ in $I_1$
16:     if Tracking is successful then
17:         Store correspondence $(x_i, x'_i)$
18:     end if
19: end for
20: Compute pose for $I_1$ using $\mathbf{x}'$ and $\mathbf{X}$
21: for all $I_t$ in image sequence, $t > 1$ do
22:     $\mathbf{x} \leftarrow \mathbf{x}'$
23:     Track $x_i \in \mathbf{x}$ in $I_{t+1}$
24:     Count number of remaining points
25:     if Too many points were lost then
26:         Add new 2D-3D correspondences using procedure described
            from lines 1 to 13 on image $I_t$
27:     end if
28:     Compute pose for $I_{t+1}$ using $\mathbf{x}'$ and $\mathbf{X}$
29:     $t \leftarrow t + 1$
30: end for
```

5 Experiments

We present in this section some experiments of our method on several building façades. After giving some details about camera calibration, results are given for two test sequences. The following results have been computed on a Pentium IV running at 2.5 GHz with 512 Mo of RAM, and using a nVidia Quatro2 EX graphic card for rendering.

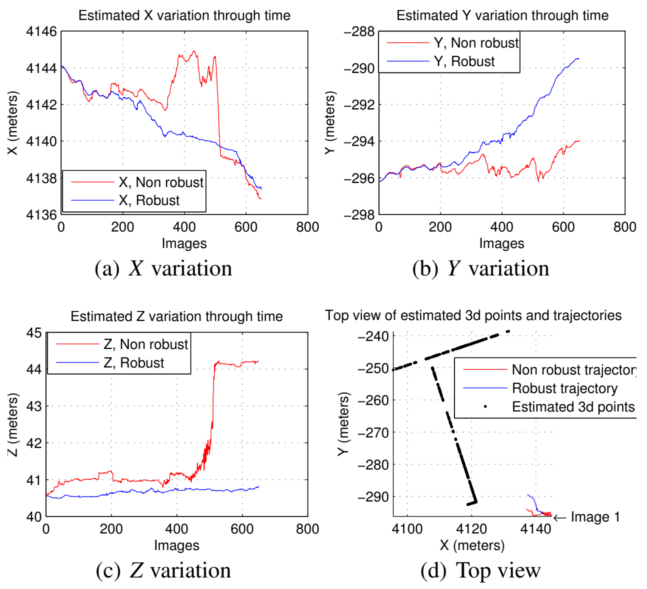
Camera calibration. In our context, we do not need extremely accurate intrinsic calibration, thanks to the ratio between pixel size and dimensions of projected model (see also [8]). We set the principal point coordinates to $[0 0]^\top$. As for the focal length, we can use parameters given by the device constructor, or even EXIF[3] information stored in the images,

[3] *Exchangeable Image File Format*

like in [16].

Tracking results. The test sequence presented in this section is composed of low-resolution images (400×300 pixels). It has been acquired with a digital camcorder, and contains 650 images of several façades. The motion of the camera is generic and does not target any particular façade, which makes tracking even more difficult. Two tracking results are presented. First, a simple visual servoing tracker has been used, and is labelled as *non robust*. Only façade points are used, no z-buffer optimization is performed, and the non robust version of the control law (equation 3) is used to compute the pose for each image. Though this state-of-the-art approach performs well in the case where the camera always aims at the tracked object, an important drift is introduced when this tracked object is only partially visible, disappears in several frames or when there are many reflections within the viewed scene. We therefore present tracking results using the *robust* model-tracker which is described in section 4.2. Once correspondences are manually provided for the first image, the pose itself is computed in approximately 0.2 seconds. Tracking results are presented on figure 7. The estimated (X,Y,Z) positions of the camera are given for both trackers on figures 7(a) 7(b) 7(c). A top view of the estimated trajectory in the UTM coordinate system together with the positions of the measured 3D points is also illustrated on figure 7(d). Finally, a rendering of the GIS model superimposed on the corresponding images is presented on figure 8. Tracking is computed in 441 seconds for the non robust version and 637 for the robust one. One can note that the different improvements we brought make the tracking more robust and less sensitive to drift than the simple visual servoing algorithm. It is particularly clear on the curve of the estimated altitude (7(c)), which is not supposed to vary more than a few centimeters. We can notice however that though seriously attenuated, drift in pose estimation is still noticeable and has to be lowered.
Tracking results in a simple case are are presented on figure 9. Here, only a single façade is viewed and tracked throughout the whole sequence, and no non-modelised obstacle (such as poles, pedestrians, etc.) hide any part of the façade. One can see here that robust and non-robust results are quite similar, since we are in a quasi ideal test case.
Results shown on figure 10 depict quite the same test scenario, except for the fact that a car is hidding some part of the tracked façade. 2D tracked points that belong to this car are always assigned wrong 3D coordinates and thus make the non-robust tracker fail (here the tracker is completely lost after the 110^{th} frame). On the contrary, the robust tracker succeeds to get rid of these false 2D-3D point correspondences and tracks correctly the façade for all images.
Videos for these registration results are available online[4].

[4] http://www.irisa.fr/temics/staff/sourimant/tracking



(a) X variation (b) Y variation (c) Z variation (d) Top view

Fig. 7 Tracking results for the image sequence *Ifsic*

6 Conclusion and future work

We presented in this paper a methodology for registering different kind of data, as a mandatory step to large-scale city modeling, by interpreting GPS measures with regards to a GIS database so as to get a coarse estimation of the camera pose, and then by refining these estimates using suitable visual virtual servoing algorithms. We have then computed geo-referenced poses of the camera, which provides us with useful information for future geometric refinement of the GIS-based 3D models, using directly the registered image sequences.
However, there is still room for improvement for this part of the work. First, we would want to get rid of the manual part of the pose initialization process, by developing a fully automatic procedure to perform this computation. Moreover we could use such registration process to reduce drift introduced during the tracking phase. Such a procedure is currently studied.
In the near future, we plan to take advantage of this method by using the images registered with the GIS database to enhance the coarse polyhedral 3D models we just used, and more precisely compute their local geometric details and real texture information.

Acknowledgements The fund support from France Telecom R&D for this work is gratefully acknowledged. The authors would like to thank especially Christian Bouville, Jerome Royan, Eric Marchand and the SIMILAR European Network of excellence, for the helpful discussions and advices they gave.

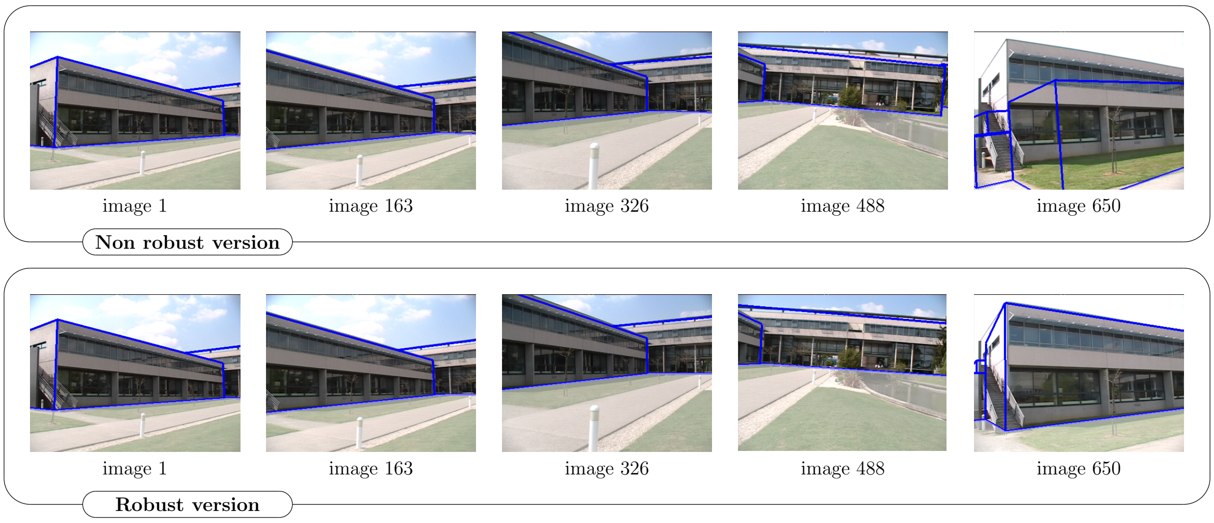


Fig. 8 Visual tracking results with superimposed 3D model (*Ifsic*)

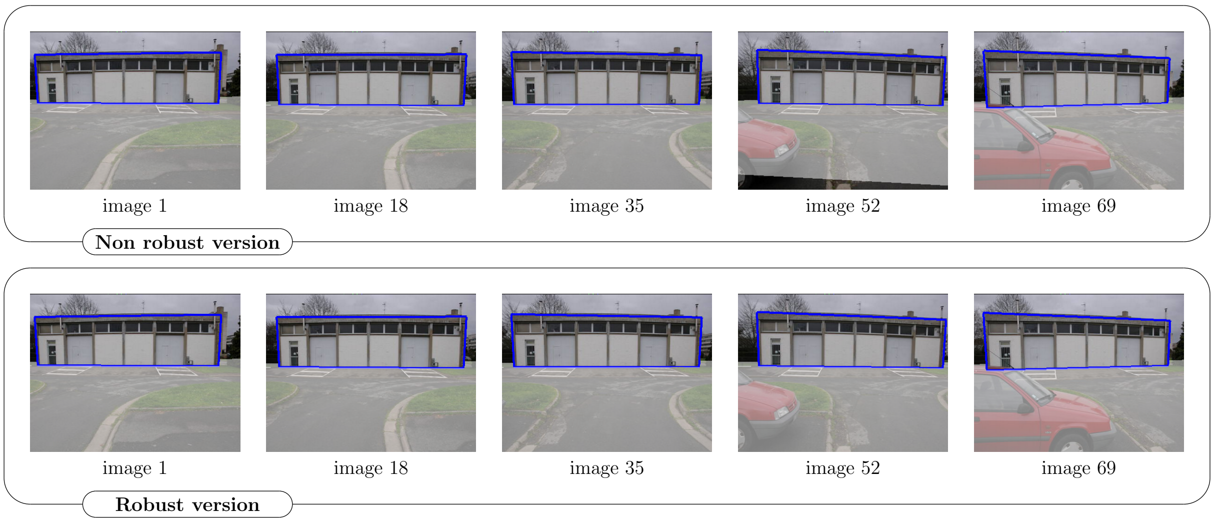


Fig. 9 Visual tracking results with superimposed 3D model (*Beaulieu*)

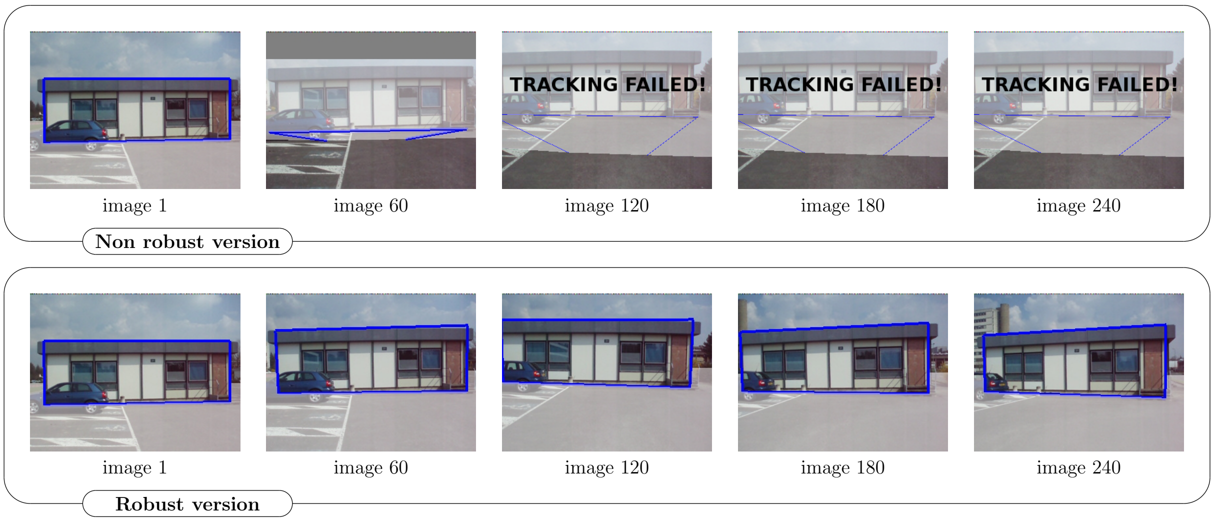


Fig. 10 Visual tracking results with superimposed 3D model (*Beaulieu2*)

References

1. Akbarzadeh, A., Frahm, J.M., Mordohai, P., Clipp, B., Engels, C., Gallup, D., Merrell, P., Phelps, M., Sinha, S., Talton, B., Wang, L., Yang, Q., Stewéius, H., Yang, R., Welch, G., Towles, H., Nisté, D., Pollefeys, M.: Towards urban 3d reconstruction from video. In: 3DPVT (2006)
2. Comport, A., Marchand, E., Pressigout, M., Chaumette, F.: Real-time markerless tracking for augmented reality: the virtual visual servoing framework. IEEE Trans. on Visualization and Computer Graphics (2006)
3. Debevec, P.E., Taylor, C.J., Malik, J.: Modeling and rendering architecture from photographs. In: ACM SIGGRAPH (1996)
4. DeMenthon, D., David, P., Samet, H.: Softposit: An algorithm for registration of 3d models to noisy perspective images combining softassign and posit. Tech. rep., Center for Automation Research (2001)
5. DeMenthon, D., Davis, L.S.: Model-based object pose in 25 lines of code. In: European Conference on Computer Vision (1992)
6. Elaksher, A.F., Bethel, J.S., Mikhail, E.M.: Reconstructing 3d building wireframes from multiple images. In: Proceedings of the ISPRS Commission III Symposium on Photogrammetric Computer Vision (2002)
7. Frueh, C., Zakhor, A.: Automated reconstruction of building facades for virtual walk-thrus. In: Proceedings of the SIGGRAPH 2003 Conference on Sketches & Applications (2003)
8. Gomez, J.F.V., Simon, G., Berger, M.O.: Calibration errors in augmented reality: A practical study. In: ISMAR '05: Proceedings of the Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality (2005)
9. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, ISBN: 0521540518 (2004)
10. Hofman, A.D., Maas, H.G., Streilein, A.: Knowledge-based building detection based on laser scanner data and topographic map information. In: Proceedings of the ISPRS Technical Commission III Symposium on Photogrammetric Computer Vision (2002)
11. Pollefeys, M., Gool, L.V.: Visual modeling: from images to images (2002)
12. Quennesson, K., Dellaert, F.: Rao-blackwellized importance sampling of camera parameters from simple user input with visibility preprocessing in line space. In: 3DPVT (2006)
13. Royan, J.: Visualisation interactive de scènes urbaines vastes et complexes à travers un réseau. Ph.D. thesis, Laboratoire France Telecom R&D (2005)
14. Schindler, G., Krishnamurthy, P., Dellaert, F.: Line-based structure from motion for urban environments. In: 3DPVT (2006)
15. Schindler, K., Bauer, J.: A model-based method for building reconstruction. In: Proceedings of the ICCV Workshop on Higher-Level Knowledge in 3D Modeling and Motion (2003)
16. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: exploring photo collections in 3d. In: ACM SIGGRAPH (2006)
17. Snyder, J.P.: Map projections - A working manual. US Geological Survey Professional Paper 1395 (1987)
18. Suveg, I., Vosselman, G.: Localization and generation of building models. In: Proceedings of the ISPRS Technical Commission III Symposium on Photogrammetric Computer Vision (2002)
19. Teller, S., Antone, M., Bodnar, Z., Bosse, M., Coorg, S., Jethwa, M., Master, N.: Calibrated, registered images of an extended urban area. Int. J. Comput. Vision (2003)
20. Tomasi, C., Kanade, T.: Detection and tracking of point features. Tech. rep., Carnegie Mellon University (1991)
21. Tupin, F., Roux, M.: Detection of building outlines based on the fusion of sar and optical features. PandRS (2003)
22. Werner, T., Zisserman, A.: Model selection for automated architectural reconstruction from multiple views. In: Proceedings of the British Machine Vision Conference (2002)

Gaël Sourimant Gaël Sourimant obtained his MS. Degree in Computer Science in 2004 from University of Rennes, France. He is currently doing his PhD Thesis at the University of Rennes, in the Temics project, on data fusion in the context of city modeling. His research interests are computer vision, virtual and augmented reality, image processing and synthesis.

Luce Morin Luce Morin was born in Grenoble, France. She received the engineer degree in physics and digital images from ENSPS, Strasbourg in 1989 and the PhD degree from INPG, Grenoble in 1993. Since then she has been an Assistant Professor at the University of Rennes, France, where she teaches computer science, image processing and computer vision. Her research activities, as a member of the Temics project in the IRISA/INRIA Rennes laboratory, deal with 3D modelisation for video sequence communication.

Kadi Bouatouch Kadi Bouatouch is an electronics and automatic systems engineer (ENSEM 1974). He was awarded a PhD in 1977 and a higher doctorate on computer science in the field of computer graphics in 1989. His is working on global illumination, lighting simulation for complex environments, GPU-based high fidelity real-time rendeirng, parallel graphics algorithms, augmented reality and compter vision. He is currently Professor at the university of Rennes 1 (France) and researcher at IRISA (Institut de Recherche en Informatique et Systmes alatoires). He is member of Eurographics, ACM and IEEE. He was and is member of the program committees of several conferences and workshops and refree for several Computer Graphics journals like: The Visual Computer, IEEE Computer Graphics and Applications, IEEE trans on Visualization and Computer Graphics, IEEE trans on image processing, etc. He also acted as a referee for many conferences and workshops.