Using dense point clouds as environment model for visual localization of mobile robot

Nathan Crombez, Guillaume Caron, El Mustapha Mouaddib

The 12th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 2015)
October 28 ∼ 30, 2015 / KINTEX, Goyang city, Korea

Using dense point clouds as environment model for visual localization of mobile robot

Nathan Crombez, Guillaume Caron, El Mustapha Mouaddib

University of Picardie Jules Verne
MIS Laboratory
33 rue Saint Leu, 80039 Amiens Cedex 1, France
Email: {nathan.crombez,guillaume.caron,mouaddib}@u-picardie.fr

Abstract -

Camera 3D pose estimation, to be consistent and precise, can benefit from two things: a 3D model of the environment, as it is well known, and the photometric appearance of the environment. The latter recently received more attention from the research community. However, it is mainly tackled for conventional cameras and using 3D models obtained from their images and for this purpose. In parallel, recent tools like 3D laser scanners have been more and more improved and are now able to rapidly generate an accurate and colored dense point clouds of a scene. We propose in this paper to tackle wide field of view camera 3D pose estimation using intensities of the whole image and surrounding datasets previously acquired by a 3D laser scanner. The direct use of image intensities withdraws features detection and matching issues and ensures more consistency than using geometric features. The performance of the approach is proven in simulation and real experiments in indoor and outdoor situations.

Keywords - Localization, virtual visual servoing, point clouds, laser scanners

1. Introduction

Robot or vehicle localization, using embedded devices only, is still a challenging issue. Tackling the problem using vision is full of potential since images bring a lot of information about the environment. The current work is in this field, even if it proposes a more general method applicable to any non uniform object visual pose estimation.

Using a conventional camera can lead to precise pose estimation, up to the availability of any visual features in the camera field of view. To correct this issue, a panoramic or omnidirectional camera should be used. Thanks to its very wide field of view, the probability of seeing visual features dramatically increases. Furthermore, it maximizes the number of sensed features too, leading obviously to more constraints for motion or pose estimation.

In recent years, 3D laser scanners have been more and more improved and have now digital cameras included which assign a color to each acquired points. It is now possible to rapidly extract a geometric and visual representation of an environment. Thanks to these equipments, the number of 3D point cloud models of cities and historical monuments is increasing. This work is focused on camera localization in this kind of environments where GPS can be defeated due to occlusion, or multi-reflections. So using a large field of view camera on top of a mobile robot (Fig. 1), the goal of this work is to estimate its pose in a previously scanned environment using its images. The problem very recently received attention from researchers in perspective vision, exploiting normalized mutual information (NMI) [23] or point features [12]. However in our case, point features are not usable and we can exploit the point intensities provided by the scanner. That is why, to estimate the pose of the camera, and therefore localize the robot, we propose to rely on 3D model-based tracking and pose estimation principles but using the photometric feature rather than a geometric one and a 3D point cloud as model.

(a) (b) (c)

Fig. 1 Material: the mobile robot Pioneer 3-AT with a FishEye Camera pointing up (a) and examples of FishEye digital images acquired outside (b) and inside (c)

A way to do 3D model based visual tracking and pose estimation is to optimize the pose minimizing the error between the projection of the model of a real object and corresponding image measurements. The pose optimization process is often preceded by a linear estimation step which has well been tackled in the literature for points, in perspective vision [11], [22] or in omnidirectional vision [20], and other features as lines [2]. This kind of method can be used with feature points or lines shared by real images and virtual images of the 3D model for the first image of a sequence for initialization. For following images, as we assume being in a tracking process, the optimal pose of the previous image is used as the initial pose for the current image.

So, when an initial guess of the pose is available, matched features which are shared by images and the 3D model, are used to compute a cost function leading to a non-linear system with respect to the pose of the 3D object. The linearization of the system by a Taylor development allows to design a Gauss-Newton or Levenberg-

Marquardt optimization process using feature points [15], [10] or lines [21].

However, in our case, detection, matching and tracking of such geometric features is very difficult. The digital camera included to the scanner has a low resolution and a low intensity dynamics. So, when the points sampling rate is greater than the digital camera resolution, a 3D point and its nearest neighbors are colorized with the same RGB value. This phenomenon creates a blur effect on the colored point clouds. We also have to place the laser scanner at different geographical positions to avoid occlusions in large environments, we have to make several acquisition stations. Despite fast acquisitions, these point clouds have been acquired at different times of the day, so at different sun exposures. Because of that, two points side by side from two separate stations may have a completely different color. These problems are visible on the virtual images generated by projecting our 3D point clouds. Moreover, dealing with point clouds implies that virtual images can contain empty pixels when not any 3D point is projected in. Because of that, interest points detection and the matching (ASIFT [18], KAZE [1], Brisk [14]) between a digital image and a virtual one failed or gave us inaccurate results. For the same reasons, extracting lines are also a tricky operation considering that kind of images.

The photometric feature received recently much attention from the research community, mainly due to its potential of precision but also since it withdraws feature detection and matching issues. The photometric feature has been used to track planes in a 2D motion estimation scheme in perspective [5] and in omnidirectional [17] vision or in the visual control of a robot under the visual servoing framework in [8] (perspective vision) and [6] (omnidirectional vision).

The current work proposes to adapt the use of the photometric feature for 3D model based pose estimation in wide angle vision. For this, we use a dense point cloud as model of an environment which has not only a geometric structure but also a photometric value for each 3D point.

The rest of the paper is then organized as follow. First, the section 2 describes the point clouds database generation. After this, the pose estimation method itself is presented in section 3. Finally, simulation results and real experiments are presented in section 4 before conclusion.

2. Organized Point Cloud Database

For large scanned environments, such as cities or historical monuments, the number of 3D points can rapidly become enormous (several millions). Thus, generating virtual images using all the 3D points of these models would be really time-consuming operations. This is why we propose a solution for only considering the useful 3D points of the model depending on the robot pose.

2.1 Camera model

The wide field of view camera that we use is a fisheye camera. A fisheye camera involves an ultra wide-angle lens that generates strong visual distortions to create an hemispherical image. We use the unified spherical projection model which is well suited for single viewpoint camera but which can be also used for modelling the fisheye projection [3], considering additional radial distortion parameters.

Following the spherical model, a 3D point $\mathbf{X} = (\mathbf{X},\mathbf{Y},\mathbf{Z})^{\mathbf{T}}$ is first projected onto a unitary sphere, centered at $(0,0,\xi)^T$. The obtained point is then perspectively projected on the normalized image plane as $\mathbf{x} = (\mathbf{x},\mathbf{y})$:

$$\mathbf{x} = \mathbf{pr}_\xi(\mathbf{X}) \quad \text{with} \quad x = \frac{X}{Z+\xi\rho} \quad \text{and} \quad y = \frac{Y}{Z+\xi\rho} \ , \quad (1)$$

and $\rho = \sqrt{X^2+Y^2+Z^2}$. An image point is obtained from a 3D point using $pr_\gamma(\mathbf{X}) = \mathbf{K}\mathbf{pr}_\xi(\mathbf{X})$, knowing intrinsic parameters $\gamma = \{p_x, p_y, u_0, v_0, r_1, r_2, \xi\}$, where r_1 and r_2 are radial distortion parameters and some others compose the matrix $\mathbf{K}$:

$$\mathbf{K} = \begin{pmatrix} p_x & 0 & u_0 \\ 0 & p_y & v_0 \\ 0 & 0 & 1 \end{pmatrix}. \quad (2)$$

2.2 Organized Point Cloud

We note $P_{i,j}(\mathbf{X},\mathbf{I})$ the j^{th} 3D point from the point cloud acquired by the station i where $\mathbf{X}$ is its coordinates and I its intensity. The point cloud acquired by the station i is noted $\mathbf{PC_i} = \{\mathbf{P_{i,j}}(\mathbf{X},\mathbf{I})/\forall \mathbf{j} \in [\![\mathbf{0},\mathbf{N_i}-\mathbf{1}]\!]\}$ with N_i the number of points acquired by the i^{th} station. The complete model **PCm** contains the N registered point clouds: $\mathbf{PCm} = \{\mathbf{PC_i}/\forall \mathbf{i} \in [\![\mathbf{0},\mathbf{N}-\mathbf{1}]\!]\}$.

An organized point cloud **OPC** dataset is the name given to point clouds that resemble an organized image (or matrix) like structure, where the data is split into rows and columns. [1] We can see an **OPC** as two matrix, one which is an intensity image (Fig. 2) and another which contains the coordinates of the projected and visible 3D points in the image. In the following, we note $\mathbf{I_{OPC}}$ the intensity image of an **OPC**.

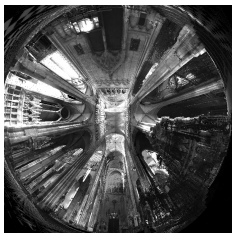


Fig. 2 Example of virtual fisheye image obtained inside a 3D point cloud environment using the unified spherical projection camera model

We expressed the generation of a virtual fisheye organized point cloud in a scene by $\mathbf{OPC}(pr_\gamma(\mathbf{PCm}),{}^{\mathbf{c_v}}\mathbf{M_o})$ where pr_γ represents the unified spherical projection previously presented and ${}^{c_v}\mathbf{M_o}$ is the virtual camera pose in the scene. A matrix ${}^{c}\mathbf{M_o}$ represents the transformation from the **PCm** (object (o)) frame to a camera (c) frame :

$${}^{c}\mathbf{M_o} = \begin{pmatrix} {}^{c}\mathbf{R_o}_{(3\times3)} & {}^{c}\mathbf{t_o}_{(3\times1)} \\ \mathbf{0}_{(\mathbf{1}\times\mathbf{3})} & 1 \end{pmatrix}_{(\mathbf{4}\times\mathbf{4})} \quad (3)$$

This matrix ${}^{c}\mathbf{M_o}_{(4\times4)}$ is formed by a rotation matrix ${}^{c}\mathbf{R_o}_{(3\times3)}$ and a translation vector ${}^{c}\mathbf{t_o}_{(3\times1)}$.

2.3 Database generation process

In our experimentations we use a model of a cathedral interior (Fig. 3(a)) and a urban model of four streets (Fig. 3(b)). These models contain tens of millions of 3D points. It is unthinkable to use these complete models as robot environment representation. To deal with this problem, offline, we create a database of organized point clouds. Then, during the online pose estimation process of an input digital image, the robot only uses one organized point cloud of the database.

(a) (b)

Fig. 3 Examples of two point cloud models: a cathedral interior visible from the outside (a) and a urban environment of four streets (b)

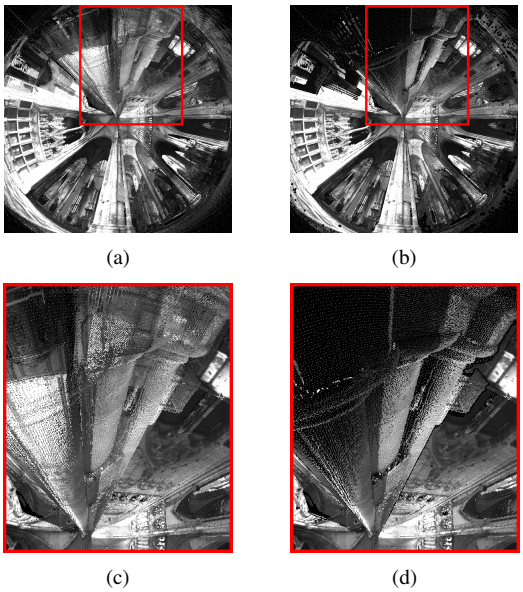
For creating the database, wide field of view virtual cameras pointing up are placed every a regular step all over the ground surface of the 3D model **PCm**. For the x camera positions ${}^{vc_x}\mathbf{M_o}$, we generate an organized point cloud $\mathbf{OPC_x}(pr_\gamma(\mathbf{PCm}),{}^{\mathbf{vc_x}}\mathbf{M_o})$. We apply the Hidden Point Removal (HPR) [13] on the $\mathbf{OPC_x}$ to remove the 3D points which should be invisible as viewed from the pose ${}^{vc_x}\mathbf{M_o}$. Figure 4 shows an $\mathbf{I_{OPC}}$ before (a) and after (b) using the Hidden Point Removal on its associated **OPC**.

Finally, the $\mathbf{OPC_x}$ treated by the HPR operator and their associated ${}^{vc_x}\mathbf{M_o}$ are stored into the database. From now, if we want to generate a virtual image from a pose ${}^{vc^*}\mathbf{M_o}$ we do not need to use the complete model **PCm** but only the organized point cloud $\mathbf{OPC_x}$ of the database whose the pose ${}^{vc_x}\mathbf{M_o}$ is the closest to ${}^{vc^*}\mathbf{M_o}$. To simplify notations in the following, the nearest $\mathbf{OPC_x}$ of the database is written $\widehat{\mathbf{OPC}}$ and $\mathbf{I}_{\widehat{\mathbf{OPC}}}({}^{\mathbf{c}}\mathbf{M_o})$ expresses the image of the $\widehat{\mathbf{OPC}}$ generated from the pose ${}^{c}\mathbf{M_o}$. A pose can also be expressed as a vector $\mathbf{r} = [\mathbf{t_X},\mathbf{t_Y},\mathbf{t_Z},\theta\mathbf{_X},\theta\mathbf{_Y},\theta\mathbf{_Z}]$. $\mathbf{r}$ is a vector representation of a ${}^{c}\mathbf{M_o}$ including three translation and three rotation degrees of freedom.

3. Localization

The aim of the work is to compute the pose of the real camera using its digital images and an organized point cloud of the environment as a reference. The image-based virtual visual servoing (VVS) is an interesting framework in a tracking process to deal with this problem. VVS is a full scale non-linear optimization technique which can be used for pose optimization based on a cost function defined in the image [9].

[1]Point Cloud Library - http://pointclouds.org



(a) (b) (c) (d)

Fig. 4 Comparison between an $\mathbf{I_{OPC}}$ before (a) and after (b) applying the HPR operator on its associated **OPC**. The zoomed part (c) show an area of (a) which contains a lot of points which should not be visible. The zoomed part (d) show the same area after the HPR operator.

3.1 Pose optimization

Our visual servoing goal is to regulate the difference between a desired digital image and virtual images obtained using the camera model projection presented in section 2.1 and the $\widehat{\mathbf{OPC}}$ of the database which has been generated from the closest pose than the previously estimated robot pose :

$$\mathbf{e} = \mathbf{I}_{\widehat{\mathbf{OPC}}}(\mathbf{r}) - \mathbf{I}^*, \tag{4}$$

The time variation of a pixel $\mathbf{x} = (\mathbf{x},\mathbf{y})$ in the image plane $\mathbf{I}_{\widehat{\mathbf{OPC}}}(\mathbf{r})$ is related by a geometric interaction matrix [7] to the pose velocity $\mathbf{v} = \dot{\mathbf{r}}$ as:

$$\dot{\mathbf{x}} = \mathbf{L_x}\mathbf{v}. \tag{5}$$

where $\mathbf{v} = (\upsilon,\omega)$ is respectively the linear and angular camera velocities. The interaction matrix $\mathbf{L_x}$ depends on the projection model defined for the unified projection model [4]. $\mathbf{L_x}$ contains the point depth $\rho = \sqrt{X^2+Y^2+Z^2}$ as defined in section 2.1. The 3D point coordinates (X,Y,Z) are easily and rapidly found because they are in the organized point cloud $\widehat{\mathbf{OPC}}$ associated to the $\mathbf{I}_{\widehat{\mathbf{OPC}}}$.
Note $I_{\widehat{\mathbf{OPC}}}(\mathbf{x},\mathbf{t})$ the intensity of the pixel $\mathbf{x}$ at time t. Under temporal luminance consistency [8] and assuming a small pixel displacement $\mathbf{dx}$, we can write :

$$I_{\widehat{\mathbf{OPC}}}(\mathbf{x}+\mathbf{dx},\mathbf{t}+\mathrm{dt}) = \mathbf{I}_{\widehat{\mathbf{OPC}}}(\mathbf{x},\mathbf{t}) \tag{6}$$

If $\mathbf{dx}$ is small enough, and considering a Lambertian scene, the optical flow constraint equation is valid, then

$$\nabla I_{\widehat{\mathbf{OPC}}}\dot{\mathbf{x}} + \dot{I}_{\widehat{\mathbf{OPC}}} = 0 \tag{7}$$

with $\nabla I_{\widehat{\mathbf{OPC}}}$ the spatial gradient of $I_{\widehat{\mathbf{OPC}}}(\mathbf{x},\mathbf{t})$ and $\dot{I}_{\widehat{\mathbf{OPC}}} = \frac{\partial I_{\widehat{\mathbf{OPC}}}(\mathbf{x},\mathbf{t})}{\partial t}$, is the temporal gradient. The computation of the spatial gradients $\nabla I_{\widehat{\mathbf{OPC}}}$ is conventionally done in the image with two derivative filters along $\vec{x}$ (8) and $\vec{y}$ (9) image axes :

$$\mathbf{F_x} = \frac{\mathbf{1}}{\mathbf{8418}}(-\mathbf{112},-\mathbf{913},-\mathbf{2047},\mathbf{0},\mathbf{2047},\mathbf{913},\mathbf{112}) \quad (8)$$

$$\mathbf{F_y} = \mathbf{F_x}^{\mathbf{T}} \quad (9)$$

Coefficients are obtained from a derivative Gaussian filter. However, in our case, the image $\mathbf{I}_{\widehat{\mathbf{OPC}}}$ is generated by projecting 3D points of a cloud. Consequently, the virtual image can have "holes" because of pixels which are not filled by a 3D point (like on the top of the image in Figure 4(d)). Using these pixels would affect the computation of the image gradients. That is why we only use the pixels which contain a projected 3D point and whose the six vertical and six horizontal pixels neighbors contain projected 3D points.
Substituting eq.(5) in eq.(7), we get the relationship between the intensity variation at an image point and the camera velocity:

$$\dot{I}_{\widehat{\mathbf{OPC}}} = -\nabla I^{\top}_{\widehat{\mathbf{OPC}}}\mathbf{L_x}\mathbf{v} = \mathbf{L}_{\mathbf{I}_{\widehat{\mathbf{OPC}}}}\mathbf{v}. \quad (10)$$

An image $\mathbf{I}_{\widehat{\mathbf{OPC}}}$ of size $N \times M$ can be seen as a vector of intensities : $\mathbf{I}_{\widehat{\mathbf{OPC}}} = (\mathbf{I}_{\widehat{\mathbf{OPC0}}},\mathbf{I}_{\widehat{\mathbf{OPC1}}},...,\mathbf{I}_{\widehat{\mathbf{OPCN\times M}}})$. Then, eq.(10) can be writen for the entire image as :

$$\dot{\mathbf{I}}_{\widehat{\mathbf{OPC}}} = \begin{bmatrix} \mathbf{L}_{\mathbf{I}_{\widehat{\mathbf{OPC0}}}} \\ \mathbf{L}_{\mathbf{I}_{\widehat{\mathbf{OPC1}}}} \\ \vdots \\ \mathbf{L}_{\mathbf{I}_{\widehat{\mathbf{OPCN\times M}}}} \end{bmatrix} \mathbf{v} = \mathbf{L}_{\mathbf{I}_{\widehat{\mathbf{OPC}}}}\mathbf{v} \quad (11)$$

Considering visual servoing as an optimization problem [16], [8] formulate photometric servoing control law using a Levenberg-Marquardt like optimization technique. It has been shown that it ensures better convergence than other kind of control laws for the photometric feature. According to eq.(4), the control law of the virtual camera in our case is given by:

$$\mathbf{v} = -\lambda\left(\mathbf{H}+\mu\,\mathrm{diag}(\mathbf{H})\right)^{-1}\mathbf{L}^{\top}_{\mathbf{I}_{\widehat{\mathbf{OPC}}}}\left(\mathbf{I}_{\widehat{\mathbf{OPC}}}(\mathbf{r})-\mathbf{I}^{*}\right) \quad (12)$$

where $\mathbf{H} = \mathbf{L}^{\top}_{\mathbf{I}_{\widehat{\mathbf{OPC}}}}\mathbf{L}_{\mathbf{I}_{\widehat{\mathbf{OPC}}}}$ with $\mathbf{L}_{\mathbf{I}_{\widehat{\mathbf{OPC}}}}$ the interaction matrix linking the luminance of image $\mathbf{I}_{\widehat{\mathbf{OPC}}}$ and the camera displacements.
At each iteration of the visual servoing, the pose increment $\mathbf{v}$ is used to update the camera pose using the exponential map $e^{[.]}$ of $SE(3)$:

$${}^{c}\mathbf{M}_{o}^{t+1} = {}^{c}\mathbf{M}_{o}^{t}e^{[\mathbf{v}]}. \quad (13)$$

The process is repeated until the error $\mathbf{e}$ (eq.(4)) is stable. Here, time t, represents optimization loop iteration number.

4. RESULTS

The correlation criteria that we use is the Zero-mean Normalized Sum of Square Differences (ZNSSD) between intensities of the virtual current and the desired digital images. In a previous work [19], we compared different cost function shapes obtained with several correlation criteria (ZNSSD, Zero mean Normalized Cross-Correlation (ZNCC), Mutual Information (MI)). We observed that the ZNSSD cost function has a more convex shape than the others and a more pronounced minimum. Moreover, the ZNSSD is faster to compute.

We first tackle the use of our method on a sequence of virtual images where the virtual fisheye camera is driven in a simple 3D model (Fig. 5) made of colored meshes. The goal of this section is to highlight the ability of our proposed method to succeed.

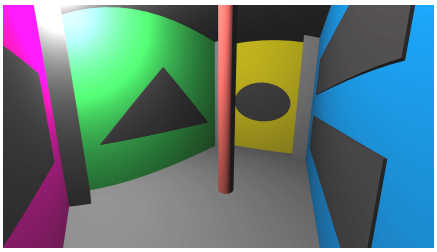
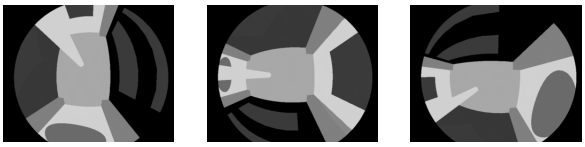


Fig. 5 The virtual scene used in our simulation experiments

4.1 Simulation results

The virtual room is 14 meters long to 9 meters wide to 7 meters height. To try our complete approach we generate a database containing 12 organized point clouds which cover the room.

A synthetic sequence is made by moving a virtual camera along a 23 meters path in the 3D room (Fig. 6). We extract 390 rendered images from this sequence. In a mean, there are around 10 cm between each image or a rotation of 2 degrees. Poses of these images are the ground truth of the virtual experiment.

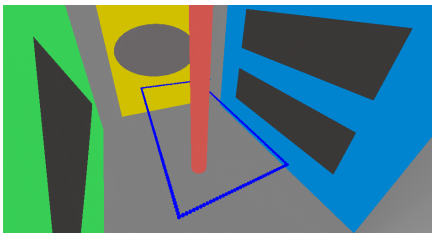


(a) (b) (c)

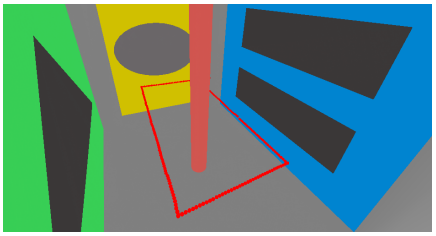
Fig. 6 Three virtual images extracted from the beginning (a), middle (b) and end (c) of the synthetic sequence

Our method succeeds to track the 3D room in virtual fisheye images all along the sequence using the organized point clouds of the database. Figure 7 shows qualitatively the proximity between true and estimated trajectories.

Indeed, the mean distance between true and estimated positions is about 4.0 cm. Specifically, the mean errors for the three axes X, Y and Z are respectively 0.8 cm, 1.3 cm and 1.2 cm. The mean errors for the three rotations around the axes X, Y and Z are respectively 0.148°,

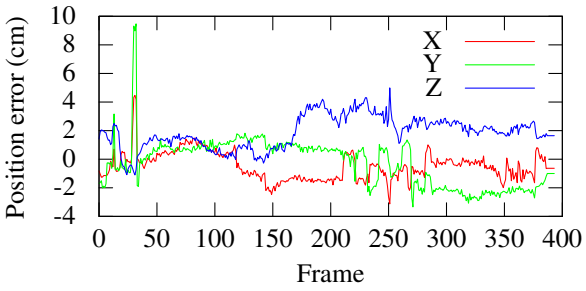


(a)



(b)

Fig. 7 Comparison between the true trajectory (a) and the estimated one (b)



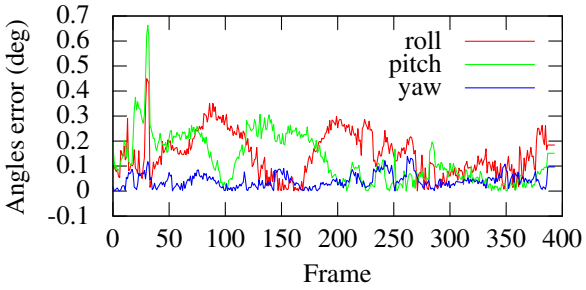


Fig. 8 Errors on the virtual camera six degrees of freedom for the 390 images of the sequence at convergence. X-axis: red, Y-axis: green, Z-axis: blue

0.128° and 0.041°. Figure 8 shows the errors on the virtual camera six degrees-of freedom for the 390 images of the sequence.

These results show our method is not only able to converge to a minimal error in the image plane, starting from image differences, but also very close to the true pose. The remaining error is due to the discretization induced by considering point clouds on which the images are registered.

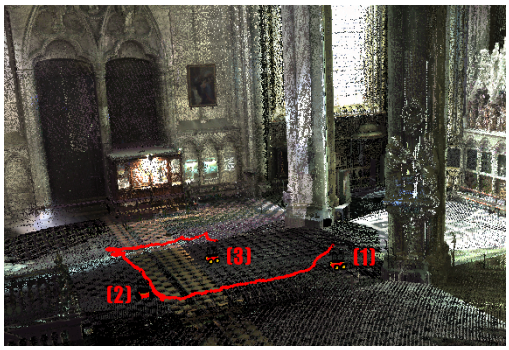
4.2 Real experiments

The real localization experiments aim to estimate the fisheye camera 3D pose for each image of sequences. The initial pose of the virtual camera is manually determined at the beginning of the image sequence but then, the complete process is automatic. We realized our experiments using two point clouds, a model of a cathedral interior (Fig. 3(a)) and a urban model of four streets (Fig. 3(b)). These points clouds have been acquired by a Leica Geosystems TPS400 laser scanner for the former, and a Faro Focus 3D, for the latter.

A. Cathedral interior

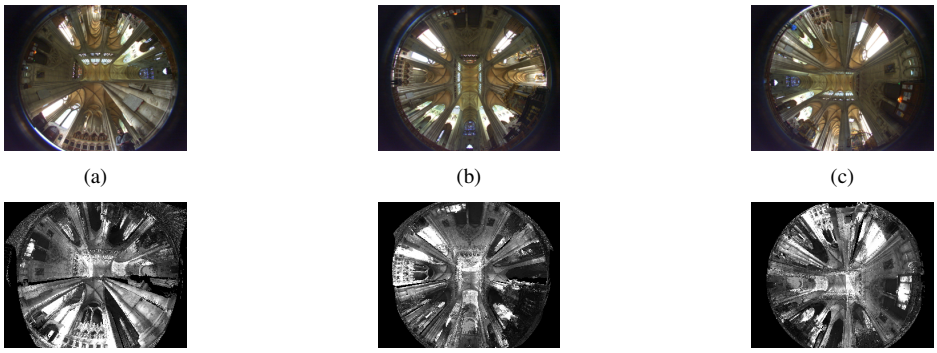
This environment has been scanned by placing the laser scanner at more than 50 positions. The model composed by the acquired point clouds contains more than 30 million of 3D points for a volume of around 140 meters long to 50 meters wide to 42 meters height.

In a first time, we produce the organized point clouds database. We experimentally chose a one meter step between each camera positions for creating the database. When the database is loaded by the robot each $\mathbf{OPC_x}$ pose ${}^{vc_x}\mathbf{M_o}$ is added to a k-dimensional tree. We aim to estimate the wide-angle camera 3D pose for each image of a sequence of 600 images acquired by the robot which navigate inside the cathedral along a path of more than 20 m (Fig. 10 (a-c)). The pose of the virtual camera is optimized minimizing the difference between the virtual view of the environment and the desired digital image. At each iteration of the pose optimization, the virtual image is generated using the organized point cloud of the database which has been generated from the closest pose than the previously estimated robot pose. The closest **OPC** pose is rapidly found using a nearest neighbor search method in the k-dimensional tree. Figure 10 (d-f) show virtual images rendered at optimal poses corresponding to real images of figure 10 (a-c). In order to qualitatively evaluate pose estimations, figure 9 shows the estimated trajectory rendered inside the complete 3D cathedral model. The robots labelled (1), (2) and (3) are rendered respectively at the estimated poses of the digital images of figure 10 (a-c), in other words at the same camera poses which generated the virtual images of figure 10 (d-f).



(a)

Fig. 9 The estimated trajectory rendered inside the complete 3D cathedral model



(a) (b) (c)

(d) (e) (f)

Fig. 10 Some digital images at the (a) beginning, (b) middle and (c) end of the sequence acquired by the robot. (d-f) show virtual images obtained at optimal poses corresponding to real images (a-c). The estimated trajectory rendered inside the complete 3D cathedral model (g)

B. Urban streets

This environment has been scanned by placing the laser scanner at 13 geographical positions. The model composed by the acquired point clouds contains more than 10 million of 3D points for about 1000 meters of streets length.

This time, we produce the organized point clouds database with a three meters step between each camera positions because there are fewer things that can cause occlusions unlike inside the cathedral. We aim to estimate the wide-angle camera 3D pose for each image of a sequence of 620 images acquired by the robot which navigate in a street along a path of more than 110 meters (Fig. 11 (a-c)). In order to qualitatively evaluate pose estimations, figure 12 shows the estimated trajectory superimposed to an aerial view of the street in which the experiment took place. One can note that between the start and end positions of the camera, it moved progressively going closer and closer to the cathedral portal. This is true and it can be seen too in fisheye images of figure 11. An attached video shows other results for a longer path.

Fig. 12 The complete trajectory of the robot (more than 110 meters) successfully estimated

We obtained the same kind of results from other experiments realized using a car with the camera on top of it or moving manually the camera on top of head, walking in the street.

5. CONCLUSIONS

The visual tracking of dense point clouds models for wide-angle cameras, exploiting their acquired images have been successfully led on a robot, when the camera is hand-led and even on the top of a car. To reach this goal, we proposed a method to only considering the useful 3D points of the model depending on the robot pose. The virtual images generated with the useful points are taken as input of a Levenberg-Marquardt optimization process to compute the optimal pose of the camera. The pose of the camera is computed based on the photometric feature, i.e. the whole image, leading to consistent and precise pose estimations. To tackle the fact that we are dealing with point clouds, adaptations to the photometric camera servoing have been proposed. In future work we would like to do a GPU implementation of our process and adapt our organized point cloud database generation to be able to localize a UAV.

References

[1] P. Alcantarilla, A. Bartoli, and A. Davison, *Kaze features*, Conf. on Computer Vision - Vol.Part VI (Berlin, Germany), ECCV'12, 2012, pp. 214–227.

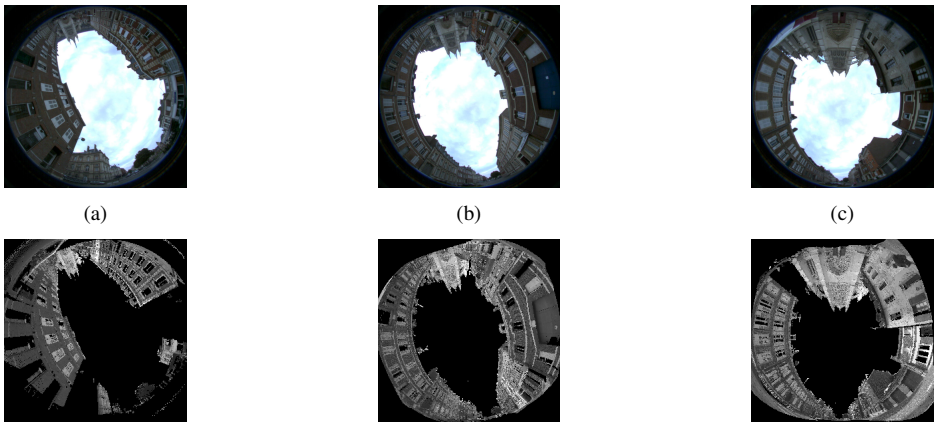
[2] A. Ansar and K. Daniilidis, *Linear pose estimation from points or lines*, European Conf. on Computer Vision (Copenhagen, Denmark), vol. 25, May 2002, pp. 578–589.

[3] J. P. Barreto and H. Araujo, *Issues on the geometry of central catadioptric imaging*, IEEE Int. Conf. on Computer Vision and Pattern Recognition (Hawaii, USA), December 2001, pp. 422–427.

[4] J.P. Barreto, F. Martin, and R. Horaud, *Visual servoing/tracking using central catadioptric images*, Experimental Robotics VIII. (2003).

[5] S. Benhimane and E. Malis, *Homography-based 2d visual tracking and servoing*, Int. Journal of Robotics Research **26** (2007), no. 7, 661–676.

[6] G. Caron, A. Dame, and E. Marchand, *Direct model based visual tracking and pose estimation using mutual information*, Image and Vision Computing, IMAVIS **32** (2014), no. 1, 54–63.



(a) (b) (c)

(d) (e) (f)

Fig. 11 Some digital images at the (a) beginning, (b) middle and (c) end of the sequence acquired by the robot. (d-f) show virtual images obtained at optimal poses corresponding to real images (a-c)

[7] F. Chaumette and Hutchinson S., *Visual servo control, part i: Basic approaches*, IEEE Rob. and Autom. Mag. **13** (2006), no. 4, 82–90.

[8] C. Collewet and E. Marchand, *Photometric visual servoing*, IEEE Trans. on Robotics **27** (2011), no. 4, 828–834.

[9] A.I. Comport, E. Marchand, M. Pressigout, and F. Chaumette, *Real-time markerless tracking for augmented reality: the virtual visual servoing framework*, IEEE Trans. on Visualization and Computer Graphics **12** (2006), no. 4, 615–628.

[10] D.F. DeMenthon and L. S. Davis, *Model-based object pose in 25 lines of code*, Int. Journal of Computer Vision **15** (1995), 123–141.

[11] R. M. Haralick, C. Lee, K. Ottenberg, and M. Nolle, *Analysis and solutions of the three point perspective pose estimation problem*, Tech. report, Hamburg, Germany, 1991.

[12] Carlos Jaramillo, Ivan Dryanovski, Roberto G. Valenti, and Jizhong Xiao, *6-dof pose localization in 3d point-cloud dense maps using a monocular camera*, ROBIO'13, 2013, pp. 1747–1752.

[13] S. Katz, A. Tal, and R. Basri, *Direct visibility of point sets*, ACM Trans. Graph. **26** (2007), no. 3, 24.

[14] S. Leutenegger, M. Chli, and R. Y. Siegwart, *Brisk: Binary robust invariant scalable keypoints*, IEEE Int. Conf. on Computer Vision (2011), 2548–2555.

[15] D. Lowe, *Fitting parameterized three-dimensional models to images*, IEEE Trans. on Pattern Analysis and Mach. Intell. **13** (1991), 441–450.

[16] E. Malis, *Improving vision-based control using efficient second-order minimization techniques*, IEEE Int. Conf. on Robotics and Automation, ICRA'04 (New Orleans), vol. 2, April 2004, pp. 1843–1848.

[17] C. Mei, S. Benhimane, E. Malis, and P. Rives, *Efficient homography-based tracking and 3-d reconstruction for single-viewpoint sensors*, IEEE Transactions on Robotics **24** (2008), no. 6, 1352–1364.

[18] J.M Morel and G. Yu, *Asift: A new framework for fully affine invariant image comparison*, SIAM J. Img. Sci. **2** (2009), no. 2, 438–469.

[19] Crombrez N., Caron G.e, and Mouaddib E., *Colorisation de nuages de points 3d par recalage dense d'images numriques*, TS. Traitement du signal **31** (2014), no. 1-2, 81–106, fre.

[20] A. Paulino and H. Araujo, *Pose estimation for central catadioptric systems: An analytical approach*, Int. Conf. on Pattern Recognition (Washington, DC, USA), IEEE Computer Society, 2002.

[21] T.Q. Phong, R. Horaud, A. Yassine, and P.D. Tao, *Object pose from 2-d to 3-d point and line correspondences*, 1995.

[22] L. Quan and Z. Lan, *Linear n-point camera pose determination*, IEEE Trans. Pattern Anal. Mach. Intell. **21** (1999), no. 8, 774–780.

[23] Ryan W. Wolcott and Ryan M. Eustice, *Visual localization within LIDAR maps for automated urban driving*, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (Chicago, IL, USA), September 2014, pp. 176–183.