Panoramic Volumes for Robot Localization[*]

Matej Artač, Matjaž Jogan, Aleš Leonardis
University of Ljubljana,
Faculty of Computer and Information Science
Tržaška 25, SI-1000 Ljubljana, Slovenia
{*matej.artac, matjaz.jogan, ales.leonardis*}*@fri.uni-lj.si*

Hynek Bakstein
Center for Machine Perception,
Dept. of Cybernetics,
FEL, Czech Technical University in Prague
Karlovo nám. 13, 121 35 Prague, Czech Republic
bakstein@cmp.felk.cvut.cz

***Abstract*— We propose a method for visual robot localization using a panoramic image volume as the representation from which we can generate views from virtual viewpoints and match them to the current view. We use a geometric image-based rendering formalism in combination with a subspace representation of images, which allows us to synthesize views at arbitrary virtual viewpoints from a compact low-dimensional representation.**

***Index Terms*— visual localization, panoramic images, subspace representation, mobile robots**

I. Introduction

Visual information is widely used for representing the environment and for estimating the spatial orientation of mobile robots [1]. The internal representation of the environment typically consists of a 3D model of the world, where spatial coordinates of points in space are usually obtained using geometric computations based on local features [2]. This approach makes localization less dependent on the number of training locations, however, such a representation is rather sparse and prone to numerical errors. Alternatively, appearance-based models which use local or global appearance [3], [4] require no geometry computation and can work even for environments where feature detection or feature matching fail. However, these methods generally require training images taken at a large number of locations to densely cover the environment.

In this paper we propose a method that combines the simplicity and power of global appearance representations with a geometric model for image synthesis in order to extend the range of position estimation to a larger area around the explored trajectory. This method uses a subspace representation of a panoramic image volume, which, combined with the geometric *image-based rendering* (IBR) formalism, allows to synthesize and compare views at a large number of virtual viewpoints.

In the training phase we store subspace representations of complete images taken along a straight trajectory with a camera pointed perpendicularly to the direction of robot's motion. We stack these representations into a spatio-temporal volume. At the localization phase, we employ an image-based rendering approach called X-slits rendering [5], which allows us to create virtual views from camera locations not included in the training set. For example, we can render visually faithful views from the camera viewpoints that are further away or closer to the scene than the training trajectory. We then estimate the location of the robot directly from the IBR parameters of the virtual view which is most similar to the query image. Fig. 1 depicts this process. As our results show, this method significantly extends the localization from the reference viewpoint locations to virtually any location which contains at least some of the elements visible from the reference views. By applying IBR we can achieve this without using a depth map and without any correspondences of local features.

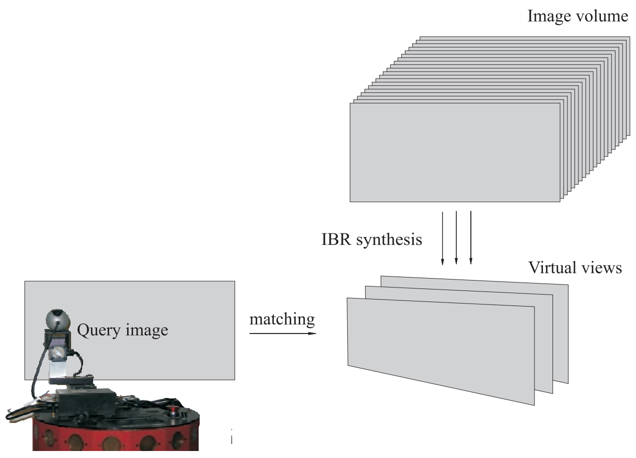


Fig. 1. The basic operations of the image-based rendering for localization.

For a review of image based rendering methods see [6]. X-slit rendering for visualization of novel views was described in [5] and was recently applied for photorealistic virtual reality applications [7].

The most related work in localization is the one by Yagi et al. [8], where the authors extract the horizon line of the omnidirectional image at each location and stack the lines into a spatio-temporal representation. They infer the location by retrieving horizon lines for viewpoints not contained in the exploration trajectory. However, the horizon line captures only a portion of the available information on the environment

[*] This research has been supported in part by the following funds: Research program Computer Vision P2-0214 (RS), EU FP6-004250-IP project CoSy, EU FP6-511051-2 project MOBVIS, CONEX project, SI-A project, BeNoGo IST-2001-39184, Kontakt ME 678, STINT Dur IG2003-2 062 and project MSM 6840770013.

and is therefore prone to ambiguities.

In contrast to this method, we use a full appearance-based representation. The major drawback of using whole images is the storage requirement needed to obtain and use a volume of images. However, since the visual information stored in the volume is highly redundant, one can represent the images in a much lower dimensional subspace representation using supspace methods such as Principal Component Analysis (PCA) [9]. PCA offers a way to achieve a compact representation while maintaining the highest possible level of visual reconstruction of images in the least squared error sense. Further, its ability to retrieve an arbitrary element of the input data, as demonstrated in e.g. [10], enables the synthesis of virtual views in a quick and intuitive manner, performing reconstruction and integration of only those fractions of each reference view that are needed according to IBR parameters.

When dealing with robot localization, we typically encounter two scenarios: the recovery after kidnapping and localization during continuous navigation. After being kidnapped, the robot is at an unknown location, presumably somewhere within a range of the previously mapped space. In this case, we have to perform a search in an extended portion of the search space and then find the viewpoint parameters which generate the view that is most similar to the current observation. Once the location has been estimated, we predict the next location according to the heading direction to generate only a few hypotheses in the area where the robot is most likely positioned. In this paper we focus only on the kidnapped robot problem.

In the next section we provide the geometrical background of virtual view synthesis, explain how to represent the volume in a PCA subspace, and how to perform localization with image synthesis from local portions of reference views. In Section III, we give experimental results of our method, and, in Section IV, we give a conclusion and an outlook of future work.

II. Virtual views and localization

A. *Virtual view synthesis using the X-slits rendering*

In this section we overview an IBR method called X-slits rendering [5]. We have chosen this method for the novel view generation because it has a geometric formalism which allows us to easily connect the position of a virtual camera to the position of the robot with respect to the image-based representation of the scene. Moreover, it belongs to a group of image-based rendering methods [6] which do not require information about the scene depth. The method we explain here applies to an ordinary perspective camera. With a few modifications, it could apply to any other type of camera, e.g., a camera equipped with a fish-eye lens or a panoramic camera [7].

The input to the X-slits method is a sequence of images captured on a straight trajectory. The trajectory does not have to be an exact straight line and the images do not have to be acquired with constant displacement, as we can use a method from [11] to regularize the spatio-temporal volume. The X-slits method creates virtual views composed from columns of the input images. This composition is illustrated in Fig. 2 (a). We sample a ray β from an image at position x_i into a column β in the image from the virtual camera V, which is at x_v along the trajectory and d_r away from the trajectory. Note that the virtual camera cannot have the field of view any larger than that of the input cameras.

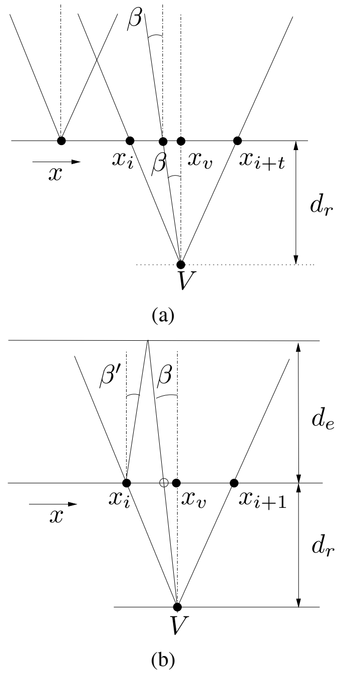


Fig. 2. (a) X-slits rendering and (b) normalization at depth d_e.

The input sequence is discrete, therefore it is likely that we want to sample a ray β from an image at a position which is not in the image sequence. The only solution is to approximate this ray by another ray β' from the nearest image in the sequence, as it is depicted in Fig. 2 (b) by the image at the position x_i,

$$i = \arg\min_{j} \left| x_j - (x_v + d_r \tan(\beta)) \right| . \tag{1}$$

However, this approximation holds only at a certain depth. We denote this depth as d_e and we call it the *normalization depth*. We can now compute the approximating ray β' given the approximated ray β, the normalization depth d_e, and the location of the virtual view:

$$\beta' = \arctan\left(\frac{(d_e + d_r)\tan(\beta) - (x_v - x_i)}{d_e} \right) . \tag{2}$$

When moving closer to or further from the scene, objects in the scene should get bigger or smaller. Equation (2) makes sure that the scene scales properly in the horizontal direction. However, we also have to set the resampling function in the vertical direction. This rescaling again depends on the normalization depth and the distance from the trajectory, as it is depicted in Fig. 3:

$$\varphi' = \arctan\left(\frac{d_e \tan(\varphi)}{d_e + d_r}\right). \quad (3)$$

Note that scaling will be correct only at depth d_e.

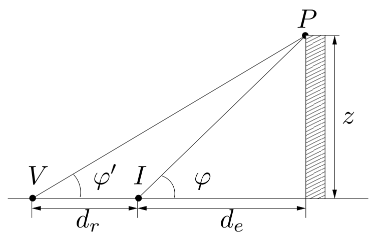


Fig. 3. Vertical scaling at normalization depth d_e.

The parameter d_e therefore influences the level of scaling performed on an image and the smoothness of the rendered scene. Underestimation of d_e results in spatial repetition of some scene portions on the virtual view, while by overestimation some portions might not appear on the virtual view. The method does not require any depth map of the scene, but it assumes a single depth of scene for the whole virtual view. We set d_e either to the value of an average scene depth, or the depth of the most dominant object visible from the virtual viewpoint. A constant d_e for a volume is a good approximation, but in principle we can have different values of d_e for different virtual viewpoints.

The virtual camera viewpoint V is determined by its distance from the straight trajectory and by a position of the nearest input image x_v. We can think about x_v as a position on a line where the optical axis of the virtual camera intersects the straight input trajectory. This expression of the position of V directly determines the position of the robot with respect to the input trajectory.

B. Image-based rendering with PCA

In the previous subsection we introduced a formal basis for image synthesis from a set of reference views. In practice, images in the volume contain a discrete set of rays, each pixel holding an information of one ray. We denote the pixel coordinates by their row r and column c. We assume that we know the focal length f, and that the center of the image is at row r_0 and column c_0. We synthesize images at virtual viewpoints by sampling image columns from the reference views and arranging them in order to compose virtual view.

Let $\mathcal{A} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n]$ be an image volume containing image matrices $\mathbf{A}_i$, $i \in [1..n]$. We transform images $\mathbf{A}_i$ into column vectors $\mathbf{a}_i$ and assemble them into a column matrix $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots \mathbf{a}_n]$. We denote the value at column c and row r of images $\mathbf{A}_i$ or virtual image $\mathbf{V}$ as $a_{(c,r)i}$ and $v_{(c,r)}$, respectively.

Let the virtual viewpoint be at offset x_v along the trajectory, the distance from the viewpoint to the trajectory d_r and the distance from the trajectory to the scene d_e. To obtain the

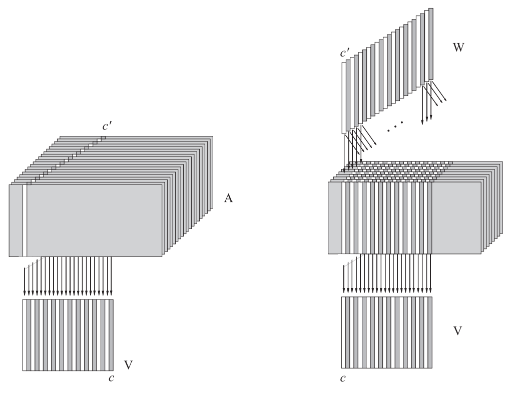


Fig. 4. (a) Image synthesis from the original images and (b) reconstructed from the PCA subspace.

image column c of the virtual view $\mathbf{V}$ using IBR, we compute $\beta = \tan\left(\frac{c-c_0}{f}\right)$. We obtain the index of the reference view using (1). From the reference image $\mathbf{A}_i$ we select the column c' computed as

$$c' = f \tan(\beta') + c_0 \,, \quad (4)$$

where β' is the ray approximation we compute using (2). The process is illustrated in Fig. 4 (a). To compensate for the depth change, we vertically resample all columns according to (3). We compute $\varphi = \tan\left(\frac{r-r_0}{f}\right)$ and $r' = f\tan(\varphi') + r_0$, where r is the row index in the virtual view, and r' is the row in the reference view $\mathbf{A}_i$.

In our approach, the images are not stored directly in an image volume, but rather as a low dimensional representation in a subspace computed using PCA. Because of a large number of images in the image volume, a batch computation of the PCA representation is often not possible. It is therefore necessary to use an incremental approach to the computation of the principal components, namely an incremental eigenspace approach as described in [4]. The eigenspace is therefore built incrementally, by gradually inserting one image after another during exploration. Furthermore, using incremental algorithm results in an open-ended model, since new images representing novel positions can always be added to the representation, as shown in a slightly different context in [4].

The incremental training using PCA produces a column matrix of eigenimages $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k]$ and, for each reference view $\mathbf{A}_i$, represented as a column matrix $\mathbf{a}_i$, a coefficient vector $\mathbf{w}_i = [w_{1i}, w_{2i}, \ldots, w_{ki}]^\top$, $i \in [1..n]$, where n is a number of reference views, and $k < n$, so that

$$\mathbf{a}_i \approx \sum_{j=1}^{k} \mathbf{U}_j \mathbf{w}_j + \bar{\mathbf{a}} \quad (5)$$

is an optimal reconstruction of the image $\mathbf{a}_i$ from a k-dimensional representation. $\bar{\mathbf{a}}$ is the mean image vector.

PCA has the essential property that the reconstruction (5) can be performed separately for each pixel [10]. This means that we do not have to reconstruct the whole images but only the columns that actually take part in the assembly of the synthesized view. In practice, this means that we can compute

$$v_{(c,\,r)} \approx \sum_{j=1}^{k} w_{ij} u_{(c',\,r)j} + \bar{a}_{(c',\,r)}\,. \qquad (6)$$

Fig. 4 (b) illustrates the image synthesis from the PCA representation. We select the columns c' in the eigenimages $\mathbf{U}$ and map them into the columns c in $\mathbf{V}$. The value of c also defines which coefficient vector $\mathbf{w}_i$ to use for the image part reconstruction according to (1).

C. *Image matching*

Once we generate the virtual views from the hypothetical locations, we have to compare them to the actual query view. As a similarity measure, we use the sum of the squared differences (SSD) of the image data vectors.

In practice, some parts of at least one of the images we compare are missing. When we move away from the trajectory, we have no information at the top and bottom rows of the image as not all the light rays are available (see Fig. 3). Further, there might be cases when we detect occlusions which would alter the similarity measure value. We therefore mask out the regions with missing data and hence not include them for the computation. However, we need to compensate for the change in the data size with a normalization factor.

Formally, we introduce a mask vector $\mathbf{m}$ with elements $m_j = 1$ if both elements x_j and y_j of some image vectors $\mathbf{x}$ and $\mathbf{y}$ are present, and $m_j = 0$ otherwise. Our modified similarity measure between the two vectors is hence

$$d_{\text{nssd}}(\mathbf{x},\,\mathbf{y}) = \frac{N}{\sum_{j=1}^{N} m_j} \sum_{\{j\,|\,m_j=1\}} (x_j - y_j)^2\,. \qquad (7)$$

d_{nssd} makes it possible for us to compare the similarities of pairs of images that have a different number of missing pixels. The square-rooted value of (7) gives a weighted Euclidean distance between the two vectors.

We have chosen this measure for its simplicity and fast computation, and because it was sufficient for experimental verification of our method. It would be straightforward to replace it with a more complex or robust measure without any modification of our framework.

III. EXPERIMENTAL RESULTS

So far we have provided the theoretical background of our method. In this section, we perform an experimental evaluation of the proposed approach.

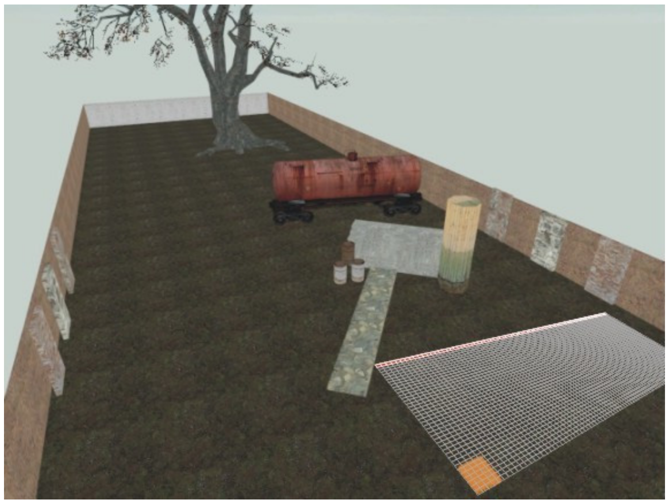
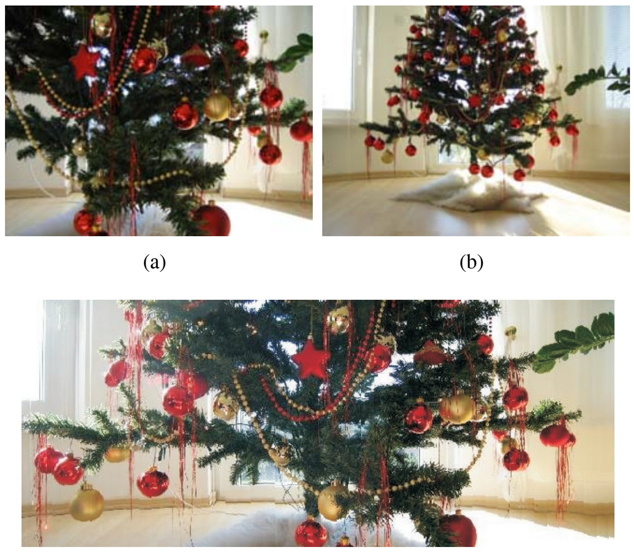


Fig. 5. A view showing the layout of the artificial scene. The superimposed grid represents the viewpoint locations, the highlighted cells showing the reference viewpoints.



(a)

(b)

(c)

Fig. 6. A sample image from (a) the real scene sequence close to the scene and (b) away from the scene (b); (c) a synthesized view.

A. *Input images*

We tested the proposed methods on two image sequences. Here we explain how we acquired them.

Artificial scene sequence. For the quantitative evaluation purposes and in order to achieve a high level of control, we captured an artificial scene from a set of viewpoints[1]. The viewpoints were organized in a uniformly spaced grid of 81×40 locations with an interval of 10 cm and with all the views pointed in the direction parallel to the shorter side of the grid. Our spatio-temporal volume consisted of 81 images from the row placed closest to the elements of the scene.

[1]We used the Source engine copyrighted by Valve Corporation, `http://www.half-life.com/`.

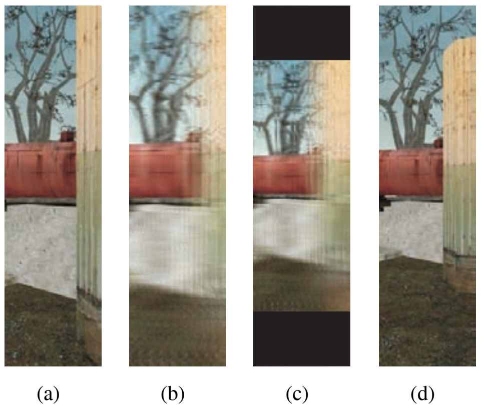


Fig. 7. (a) One of the reference images from the artificial scene sequence; (b) the reconstruction of the reference image from PCA; (c) a synthesized view; (d) the actual view at the virtual point.

Fig. 5 shows a wide view of our scene along with the grid of the image locations.

Real scene sequence. For the real environment tests we selected a scene containing a decorated tree. This scene contains a dense set of elements at different depths for the viewpoints that are close to the scene (Fig. 6 (a). We acquired images on several parallel paths, spaced at 50 mm. Each path consisted of 62 viewpoints at an interval of 10 mm. The last path was set 700 mm away from the first path. Fig. 6 (b) shows how the appearance changes substantially at this distance.

We resampled all the images to the resolution of 640×480 pixels and calibrated the camera to obtain its focal length f and principal point r_0, c_0. We performed incremental PCA on the volume images, and retained 20 eigenimages. This compression corresponds to capturing approximately 80% of the data variance in the case of the artificial scene and 84% in the case of the real scene sequence.

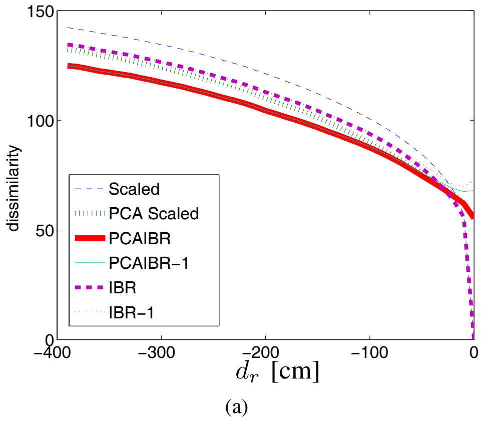
B. Image synthesis

Given an image volume and the virtual viewpoint parameters, our method produces an image representing the view at the virtual viewpoint. Fig. 6 (c) shows an example of a virtual view assembled from the Real scene sequence image volume.

Fig. 7 (a) shows a strip from a sample reference image, and Fig. 7 (b) shows a reconstruction from the PCA representation of the reference image set. We see that PCA visually alters the images and introduces a small amount of spatial aliasing. In general, we can afford a significant level of compression for the recognition task. Figs. 7 (c) and (d), show a virtual view generated with our method, and the actual view (ground truth) at that point, respectively.

C. Virtual image quality

We would like to evaluate the quality of the virtual images. To do this, we compose virtual views from a PCA image volume at a known set of parameters. The viewpoint set presented here simulated the motion away from the scene. To provide a comparison to simpler, yet less powerful methods for obtaining virtual viewpoints, we took the reference view at the same offset x as the virtual viewpoint, and rescaled it according to the virtual viewpoint parameters. Note that this is only possible if the viewpoint parameter value x equals that of one of the reference views.



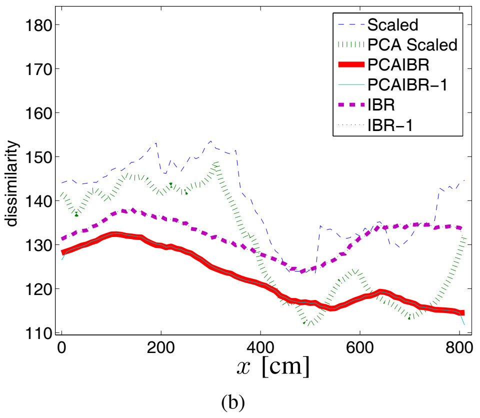


Fig. 8. Comparison of the performance of different methods: (a) average values depending on d_r and (b) $d_r = -350$ cm.

Fig. 8 shows the results of the tests. The curve labeled *Scaled* represents the method which takes the closest reference view and rescales it according to parameters d_e and d_r. *PCA Scaled* scales the PCA reconstruction of the closest reference view. *PCAIBR* represents the virtual view synthesis as described in this paper. *PCAIBR-1* uses the same method as *PCAIBR*, except that we omit the reference view at x for the image synthesis. Finally, *IBR* and *IBR-1* represent image-based rendering from original images.

We can see that the best results are achieved by synthesizing the images from the PCA representation of the image volume. This method outperforms the other approaches by far and its advantage is particularly evident when the query image is far from the training path, i.e., in viewpoints where views are expected to differ significantly from the training views.

As it can be seen from the chart, simple rescaling of the image performs worst, followed by the IBR synthesis from the original panoramic volume, and the rescaling of the image

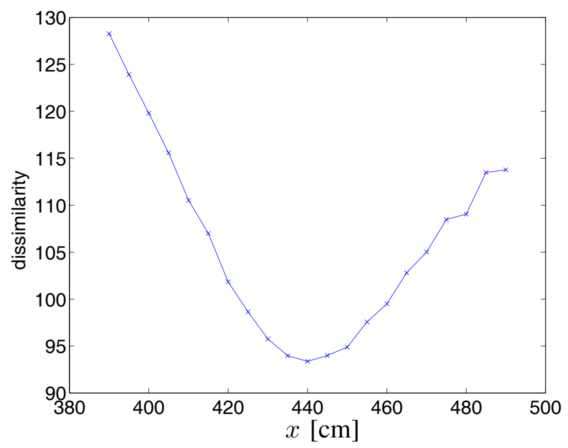


Fig. 9. Image distances depending on the location coordinate x.

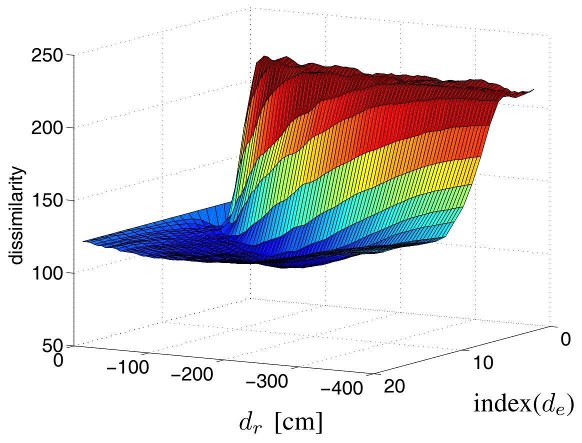


Fig. 10. Image distance for d_e and d_r where $x = 450$ cm.

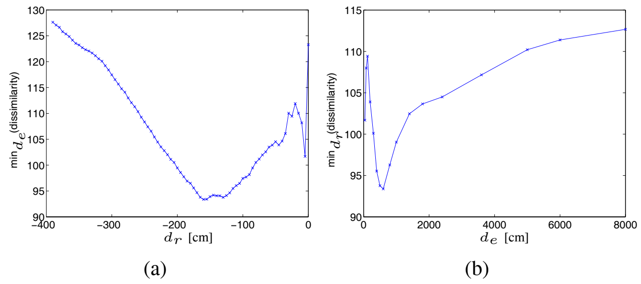


Fig. 11. The minimal image distances depending on d_r (a) and d_e (b).

from the PCA representation of the panoramic volume. If we omit some reference images, the results deteriorate only when $|d_r|$ is smaller than the reference viewpoint resolution.

We can conclude that the proposed method efficiently generates faithful novel views even between reference viewpoints and at a significant distance from training positions. This allows for efficient matching with the virtual viewpoint hypotheses, providing a good basis for localization.

D. *Image matching for localization*

To estimate a location of a query view, we generate virtual views at different viewpoints and select the most similar virtual view to the query view. Therefore, the localization becomes an optimization problem. The appearance of a virtual view depends on parameters x, d_r and d_e, two of which define a viewpoint, while the third one is scene-dependent. Even though we could estimate d_e by using a range scanner or a parallax matching of image features, we want to rely solely on the techniques presented in this paper. Therefore, we assume none of the parameters is known in advance, and we estimate all three of them in each query viewpoint location.

For this experiment, we selected an image from the artificial scene sequence at $x = 440$ cm along the panorama trajectory, and $y = 160$ cm in the perpendicular direction. We generated virtual views in a larger area around this location, thus simulating a thorough search for the minimum. We obtained a 3-dimensional tensor of the image distances.

Fig. 9 depicts the minimal values of the image distances along d_e and d_r. From this chart we also see that the global minimum is at $x = 440$ cm, which is a precise estimation. Fig. 10 shows the image distance depending on the other two parameters for the x where the global minimum occurs.

Fig. 11 shows the shape of the valley from Fig. 10 depending on d_e and d_r. We see the function shows an erratic behavior at the values of d_e that are too small, resulting in images of 10% their original size or less. For the rest of the values, the curve is nicely shaped, with a global minimum at $d_r = -160$ cm, and d_e at the distance to the tank wagon in Fig. 5. This demonstrates that the optimization selects the value for d_e as the distance to the most dominant visible object. The selection depends on the similarity measure, in our case the larger object that is in the more contrast with the rest of the scene, the more dominant it will be. In practice, we might select a different d_e for the visualisation purposes, but the minimization could prefer a parameter which produces more visual artifacts, but whose overall influence to the matching is lower.

From this example we can see that the proposed method generates views that are suitable for the minimization task which results in a location estimation.

E. *Localization*

Finally, we evaluate the proposed method's ability to perform the task of localization. The focus of the localization is the estimation of parameter x. However, we are also interested in the precision of estimation of parameter d_r, which provides the distance to the query view from the reference trajectory. To perform the experiments, we used both the artificial scene sequence and a real scene sequence. Here, we evaluate the results in terms of the distance between the estimated location and the actual location (i.e., ground truth).

In each sequence, we used only half a viewpoint resolution for composing the reference image volume. The test set consisted of all the images from each sequence. We considered each reference image to be at a fully unknown location. To get a location estimation, we used the proposed method to

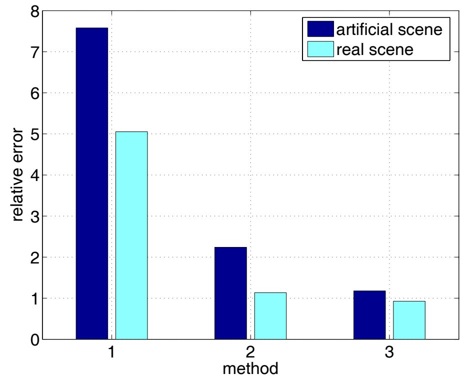


Fig. 12. Relative localization errors; 1 - direct matching, 2 - query image scaling, 3 - proposed method.

generate a set of hypotheses and selected the most similar one to each query view.

To narrow down the search space, we computed a set of up to 5 candidate values for x. We achieved this using the proposed method on a query view, as it represents a single-view image volume. In this case, only the $\frac{d_r}{d_e}$ ratio influences the resulting virtual view for simulating the forward/backward motion. We matched each virtual view to the reference views in the PCA volume [4]. The 5 closest matches in the volume thus provided the candidate values of x.

Fig. 12 gives the results for estimating x in terms of the relative localization error. The values are relative to the average distance between the query views and their closest reference view. Apart from the proposed method, the results also show the performance of direct matching of the query view to the reference views [4], and the best candidate for x obtained as explained above. High errors obtained by the direct matching demonstrate that the appearance changes considerably when the viewpoints move away from the reference viewpoints. Finding a virtual view obtained from the reference view that matches best to the reference views can provide good estimations of parameter x. The method, however is liable to aliasing, while it also cannot predict any changes in the occlusions. This lowers the overall performance of the method as shown by the artificial scene results, requiring additional estimates. Further, the location estimation is limited to the resolution of the reference viewpoints (unless we use an interpolation to increase the spatial distribution of the match candidates). Our proposed method performs best, even though, as shown by the real scene experiment, only slightly.

From these results it appears that the proposed method has only a slight edge over the competitive methods used in our experiments. However, only our method is capable of estimating the distance from the trajectory to the query viewpoint. In addition to estimating x as presented above, we also obtained the estimate of d_r for each test view. By using a resolution of 50 mm for the virtual views (which was also the resolution of the test images), the average localization error for d_r was 16.9 mm.

IV. Conclusion

In this paper we proposed a method for mobile robot localization using a panoramic image volume as the representation which allows for generation of virtual views from arbitrary viewpoints in the vicinity of training locations using image-based rendering. By searching for the viewpoint with the view that is most similar to the query view we can estimate the momentary location of the robot. The major advantage of our method is that we do not need any depth map or correspondences on local features while still being able to get precise estimates of the robot's location. By using an incremental PCA for a low-dimensional representation of the image volume, we managed to create an open-ended and compact representation, which enables fast reconstruction of virtual views. We obtain fast synthesis largely due to the inherent ability of the PCA representation to allow for pixelwise reconstruction of images.

We have demonstrated the performance of our method first on an artificial sequence and then on real data from a camera on a mobile robot. Our experiments show that the range of reliable localization is substantially larger when using IBR. When acquiring the reference images on a linear trajectory, we can reliably estimate novel viewpoint locations on a plane.

Our future work is focused on the merging of representations obtained on different paths. Further, we are exploring a combined local and global representation. We also plan experiments on navigation in large open environments.

References

[1] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 237–267, February 2002.

[2] S. Se, D. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *International Journal of Robotic Research*, vol. 8, no. 21, pp. 735–760, August 2002.

[3] M. Jogan and A. Leonardis, "Robust localization using an omnidirectional appearance-based subspace model of environment," *Robotics and Autonomous Systems, Elsevier Science*, vol. 45, no. 1, pp. 51–72, 2003.

[4] M. Artač, M. Jogan, and A. Leonardis, "Mobile robot localization using an incremental eigenspace model," in *IEEE International Conference on Robotics and Automation, May 11–15, 2002, Washington, D. C.*, vol. 1. IEEE, 2002, pp. 1025–1030.

[5] A. Zomet, D. Feldman, S. Peleg, and D. Weinshall, "Mosaicing new views: The crossed-slits projection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 6, pp. 741–754, June 2003.

[6] H.-Y. Shum and S. B. Kang, "A review of image-based rendering techniques," in *IEEE/SPIE Visual Communications and Image Processing (VCIP) 2000*, June 2000, pp. 2–13.

[7] H. Bakstein and T. Pajdla, "Rendering novel views from a set of omnidirectional mosaic images," in *Proceedings of Omnivis 2003: Workshop on Omnidirectional Vision and Camera Networks*. Los Alamitos, USA: IEEE Computer Society Press, June 2003.

[8] Y. Yagi, K. Imai, and M. Yachida, "Iconic memory-based omnidirectional route panorama navigation," in *IEEE International Conference on Robotics and Automation*, vol. 1, Taipei, Taiwan, September 2003, pp. 564–570.

[9] H. Murase and S. K. Nayar, "Visual learning and recognition of 3-D objects from appearance," *IJCV*, vol. 14, no. 1, pp. 5–24, January 1995.

[10] A. Leonardis and H. Bischof, "Robust recognition using eigenimages," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 99–118, 2000.

[11] A. Rav-Acha, Y. Shor, and S. Peleg, "Mosaicing with parallax using time warping," in *Second IEEE Workshop on Image and Video Registration (IVR'04)*, 2004.