Image-Based Environment Perception for Cognitive Technical Systems

Elmar Mair, Werner Maier, Darius Burschka and Eckehard Steinbach

Abstract— **The fundamental basis of any cognitive system is perception - all reasoning is based on it. In the past, a lot of research has been done to achieve a sophisticated environment model. However, most of the approaches are based on geometrical reconstruction, which is not sufficient for several scenarios. In this paper, we investigate image-based localization and environment modeling to provide a robust and accurate perception from a single camera. Experiments furnish proof for high pose accuracy and realistic modeling. Further, some application scenarios and preliminary results within these applications are presented.**

I. INTRODUCTION

Cognitive systems need an appropriate model of their environment in order to act and execute tasks. For path and motion planning, purely geometric models are usually used which bear information about the distances of objects of interest or obstacles and thus help to avoid collisions. For some tasks, it is not only the geometry of the environment but also its appearance which plays an important role. An image based mismatch detection, e.g., is less complex in 2D as a full search in cartesian space. A common approach for environment modeling in computer graphics is to map textures onto triangles of a mesh-based geometry model of the scene. The texture might be chosen view-dependent which enhances the authenticity of such models. However, it requires sophisticated and computationally expensive methods like raytracing to model translucent or reflective objects in a realistic way. Image-based rendering techniques ([1]) turned out to be very suitable for the photorealistic modeling of such objects because computational complexity does not depend on the properties of the environment. A densely acquired set of images is stored and used together with approximate geometry information in order to predict virtual images in a given viewpoint space. Taking this set of images as a reference model, future observations of one or multiple cognitive systems can be stored if changes in the environment happen so that the model can be updated at any time.

To allow image-based model generation out of a set of images, the exact pose of the camera for each image has to be known. In the past, many different approaches have been developed to estimate a robot's pose. Most of them are based on laser range sensors, because of its high accuracy and the fact that the required data is directly obtained without cumbersome preprocessing. However, there are several disadvantages using lasers. First of all, you are using an active sensor emitting laser rays, which makes the use within a human environment questionable. The robustness of the laser depends on the environment: the rays can be reflected by specular surfaces or go through objects made out of glass. Concerning a kitchen or also a factory environment, you usually have to deal with such surfaces and, at least in the kitchen, also with humans. Further, there are two other advantages using images for localization: we can use the same sensor as we use for the image based modeling and the error of the localization correlates with the error of the modeling algorithm from image data.

To this end, we present in this work a robust localization algorithm, which is applied to a novel probabilistic approach for image-based view synthesis. Unlike other methods in literature, our localization scheme does not require explicit markers in the environment. Experimental results show that this combination allows for photorealistic prediction of the appearance of a cognitive robot's environment which can serve as a reference for the current observation. By this, cognitive processes like the generation of visual surprise triggers can be supported.

Hence, we are dealing with several fields of research. We reference related work in the respective sections.

The remainder of this paper is structured as follows. In Section II, we present our algorithm for visual camera localization. Section III describes a novel probabilistic approach for image-based view synthesis. Before we conclude this work, we present in Section IV some experimental results and outline the integration of our module into the demonstration scenarios envisioned in the cluster of excellence CoTeSys.

II. VISUAL LOCALIZATION

A lot of research has been done within the field of visual localization to speed up and to increase the accuracy of image-based pose estimation. In this chapter, we describe an accurate and real-time capable algorithm to estimate the position of a camera within a monocular image sequence. Therefore, we use only the calibrated images without any external references as for example artificial markers or the dimensions of a known object in the world. We show how we enhance and combine state of the art algorithms to solve the visual localization problem.

This work is supported in part within the DFG excellence initiative research cluster *Cognition for Technical Systems – CoTeSys*, see also www.cotesys.org

E. Mair and D. Burschka are with the Department of Informatics, Technische Universität München, 85748 Garching, Germany elmar.mair@cs.tum.edu, burschka@cs.tum.edu

W. Maier and E. Steinbach are with the Department of Electrical Engineering, Technische Universität München, 80333 München, Germany werner.maier@tum.de, eckehard.steinbach@tum.de

A. Feature tracking using an extended KLT tracker

To localize a robot in the world from an image sequence, you first have to recognize the motion of the world in that video stream. In the last years, several different kinds of trackers have been developed. Blob-, line-, edge- and point-tracker and several combinations of these, but also more uncommon methods, as for example trackers of local energy (by the phase discrepancy in the spectral range of an image [13]) or support vector tracker ([14]).

Due to its speed and its robustness, the Kanade-Lucas-Tomasi (KLT) tracker ([9], [10], [11], [12]) is supposed to fit our requirements best. However, if you apply the convolution kernels to the whole image, you are not able to process VGA images (640x480 pixels) in real-time (25 Hz). Therefore, we enhanced this tracker by several features. The most important and processing-time saving improvements are:

1) Limitation of the image processing routines to small patches around the features to track. Therefore, we compared the run time of three different implementations:
   - the default way, where the entire image is processed,
   - a definition of regions of interests (ROIs) around the features within the image and processing only these parts of the image and
   - a patch-based alternative, where instead of processing within the entire image, new subimages around the ROIs are used as the new input to the orignal KLT. As relation to the prior image, only the position of the patch within that image is stored.

   Tests have shown, that if you track just a few features in large images, then the patch alternative is the more time efficient one. The ROI implementation is always slower than either the standard or the patch method. The processing time of the various KLT variants are compared in Table I (Section IV).
2) Linear feature propagation: The first derivative of the feature motion is saved and in the next step it is added to the propagated search window. This allows for fewer tracking iterations and larger feature displacements between the images. Further, the size of the ROIs (patches) can be reduced.
3) We also introduced the Intel®'s Integrated Performance Library[1] (IPP) routines and achieved a speed up of about 200 times for the convolution functions.

B. Finding stereo correspondences for KLT features

Due to the fact that we use a monocular camera, but work without any markers and we also do not know the exact size of one or more objects in the image, we need a stereo camera system to get the scale of the translation. This is the only reason why we use a second camera - the whole localization algorithm and all the other algorithms described in the next sections are based on a monocular video stream. If the exact scale is not needed the second camera is not needed.

W.l.o.g. we assume that the left camera is our main camera, where the features are tracked. The right camera has only supporting purpose, namely to calculate the distance of the points from the camera, as it is necessary to provide the exact scale for the localization algorithm (see Section II-C). To find the stereo correspondences, we use again the KLT tracker to select good features to track in both images. Then we apply the tracker to search for matches of the left features in the right camera. Therefore, the tracker does not check every pixel along the epipolar line like it is proposed in [16] but only at the locations found by the prior applied select-good-feature algorithm ([11]) in a small band around the epipolar line. This select-good-feature method, proposed for KLT tracking, is not designed to be a good detector in the sense of re-detect the same features in an other image. Mostly not the same points are found, but usually the points are quite close to the real correspondences. Increasing the number of interesting features in the right image and using them as starting points for the KLT tracker leads to a subpixel accurate and fast stereo matching method. Of course this method works only as long as there is a not too large affine transformation between the feature windows. This is not the case in a stereo image pair, where the cameras are mounted parallel on a rig.

C. RVGPS: robustified motion estimation between two images

Knowing the optical flow for some features, you can estimate the motion of the camera with respect to the world coordinate system. In the past, several algorithms have been developed to calculate the extrinsic parameters of an image sequence. First of all, various point algorithms, especially the 8- ([16]) and 5-point ([15]) algorithm have to be mentioned as maybe currently the most popular ones. Other basic approaches are the iterative methods as the 3-point algorithm ([17]), vision-based GPS (VGPS - [18], [19]) or bundle adjustment algorithms ([16]).

We use VGPS for our processing, due to its low complexity and its speed. VGPS estimates the transformation between the current and a reference image. Thus, no bias is accumulating as it would be the case in stepwise approaches. Someone may now think that there will be uncertainties, if we use a tracker, which tracks the features between images and if we estimate the transformation refering to a "far" distant reference. Tests have shown, that the KLT tracker drifts only little over time. Even after a large movement of the camera. Figure 1 shows some patches, as described in Subsection II-A, during a camera motion from different viewpoints. Even if we apply large camera movements, the affine distortion can be recognized very well in the whole patches. The center of the feature is drifting only up to 2 pixels from the original point. Our algorithm can deal with large affine distortions, keeping the error small enough for a robust pose estimation, while the underlying tracker is still fast enough to track in real time.

[1] http://www.intel.com/cd/software/products/asmo-na/eng/302910.htm

(a) Patch 17 in the images 4, 635, 718 and 827.

(b) Patch 30 in the images 4, 635, 718 and 827.

Fig. 1. These two patch sequences are taken from the same image stream and illustrate the affine transformation due to varying point of views. After the whole movement the center of the KLT patch drifts about one pixel, which is small enough for a robust pose estimation.

Nevertheless, some enhancements have been added to the VGPS algorithm to react to bad features, like occlusions, disocclusions and virtual features, which every tracker has to deal with. We use a least-squares M-estimator to minimize the absolute error, which is the difference between the new feature position in the image and the reprojection of the old position by the estimated transformation. This outlier detection is not only used to weigh them, but also to mark them as invalid and to exclude them from tracking. This prevents these features to have a too large influence on the pose estimation because the position is always calculated with respect to the first frame. The error caused by these features becomes irrelevant after the camera has performed a significant translation (see 4(b)). This yields to a bias free pose estimation.

Obviously, the initial feature set gets lost after some time, because the features leave the field of view of the camera. Every time the amount of trackable features drops under a defined threshold, a new feature set becomes initialized using the stereo images. Such a re-initialization can not be done in real-time and takes a few seconds depending on the number of features and the size of the image. Therefore, it is swapped to a parallel thread. After the initialization thread has terminated, the features are projected to the current image by the transformation accumulated since the start of the thread.

Due to the fact that the motion estimation is ill conditioned, if all features lie within a small area of the image, the centroid of the feature point cloud and its projection on the image plane are calculated. In case that the centroid leaves a specified inner area of the image, the initialization of a new feature set is triggered even if enough features are still trackable.

The old feature sets and their centroids are saved in a history and the reprojection of their centroids on the current image is used to determine which set allows the best conditioned motion estimation. If it is an other set than the currently active one, its features are projected on the image plane and are tried to track. Thus, the bias accumulated with each new feature set becomes removed moving backward and in the case the camera is returning to its origin, even on a different trajectory, the bias becomes zero again.

D. Summary of the visual navigation algorithm

Below you find a summary of the steps executed by the algorithm:

1) Selection of good features to track.
2) Estimation of the real distance of the features using a stereo camera.
3) Tracking the features from image to image.
4) Localization using robustified VGPS (RVGPS) and removing bad features (mismatches and wrong correspondences) as suggested by RVGPS.
5) If features are moving out of the image or not enough features could be tracked, the current feature set is saved and step 1 and 2 are executed again.
6) In each step it is checked if the centroid of any feature set in the history is closer to the center than the centroid of the current set. If yes, then the current set is saved and the old feature set is used. Jump to step 3.

III. IMAGE-BASED VIEW PREDICTION

One type of image-based scene representation that recently has become very popular uses view-dependent geometry and texture. Instead of computing a global geometry model which is valid for any view point and viewing direction, the geometry of the scene is estimated locally and holds only for a small region in the viewpoint space. It has been shown that this approach is suitable especially when the scene contains specular and translucent objects. Hence, view-dependent geometry is the basis for our algorithm which is described in this section. In the following, the term "reference image" denotes the left image of a captured stereo pair.

A. Per-pixel depth maps and view selection

In order to predict novel virtual views from captured image data, correspondences between the pixels in the reference images have to be established. Hence, for each reference image, a depth map is calculated, which for each pixel stores the distance of the scene with respect to the camera (z-coordinate in local camera coordinate system). In order to handle occlusions, we select multiple left images from other stereo pairs for intensity matching. A matching cost volume is calculated like in [2] with multiple left images from other stereo pairs and their pose information from Section II. Loopy belief propagation [3] then minimizes the matching cost globally and yields the most probable depth value for each pixel, assuming that the scene is smooth between depth discontinuities. Fig. 2 shows a reference image (left) together with the computed depth map (right). The depth map is illustrated by a grayscale image where high intensity values represent near scene points and low intensities far scene points. A triangulated mesh is reconstructed from each depth map and simplified with the algorithm in [4] for fast view synthesis.

Each time a virtual image is synthesized only a small subset of all reference images contributes to view interpolation. In real-world environments most surfaces are non-Lambertian which means that the reflected intensity depends

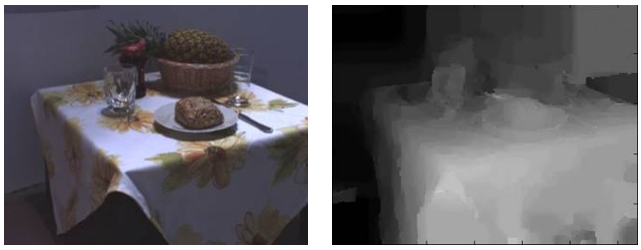


Fig. 2. A reference image (left) together with its computed depth map (right). High intensity values represent scene points which are near to the camera, low intensities correspond to far scene points.

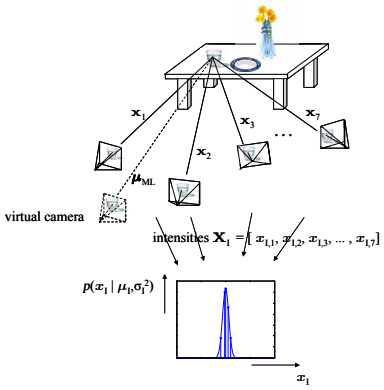


Fig. 3. View synthesis: The pixels of the virtual image are predicted from a set of reference images. The color values from the reference images are assumed to be samples from a Gaussian distribution whose mean is the true color value.

on the position of the viewer. Hence, in our approach, the reference cameras are ranked in terms of their orthogonal distance with respect to predefined rays within the viewing frustum of the virtual camera. Before a new frame is rendered, the seven closest ones are selected.

B. View synthesis

Novel virtual views are synthesized in a two-pass procedure. In the first rendering pass, the color data of the selected reference images which is associated with the triangle vertices of the view-dependent meshes is warped into the virtual view. Pixels that lie inside the projected triangles are interpolated from the color values at the corners. The second rendering pass then determines the final color of each pixel in the virtual view. Common approaches like in [5] and [6] use pose information and geometry cues like normal vectors in order to calculate deterministic interpolation weights so that the final color is a weighted sum of the reference color data. However, since it is still a challenging task to recover the correct scene geometry with state-of-the-art computer vision methods, a view synthesis scheme has to deal with erroneous correspondences between the images and with the uncertainty about the true color value at a given pixel in the virtual image.

We present in this work a novel approach for view synthesis which infers probabilistic models for the single pixel colors in the virtual image. It is assumed that the warped color values of the reference images are data samples $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_7]$ which are independently drawn from a Gaussian distribution whose mean value μ is identical to the true color value at the respective pixel in the virtual view. Fig. 3 illustrates this in case of four selected reference cameras. Each sample is a RGB-tripel $\mathbf{x}_k = [x_{\mathrm{R},k}, x_{\mathrm{G},k}, x_{\mathrm{B},k}]^{\mathrm{T}}, k = 1, \ldots, 7$ which results in a likelihood function for a multivariate Gaussian distribution

$$p(\mathbf{X} \mid \mu, \mathbf{\Sigma}) = \tag{1}$$
$$= \prod_{k=1}^{7} \frac{1}{(2\pi)^{\frac{3}{2}}} \frac{1}{\left|\mathbf{\Sigma}\right|^{\frac{1}{2}}} \exp\left\{-\tfrac{1}{2}\left(\mathbf{x}_k - \mu\right)^{\mathrm{T}} \mathbf{\Sigma}^{-1} \left(\mathbf{x}_k - \mu\right)\right\}.$$

$\mathbf{\Sigma} = \mathcal{E}\left[\left(\mathbf{x}_k - \mu\right)\left(\mathbf{x}_k - \mu\right)^{\mathrm{T}}\right]$ is a 3×3 covariance matrix and $\left|\mathbf{\Sigma}\right|$ denotes its determinant. Here, $\mathcal{E}\left[\cdot\right]$ is the expectation of a random variable.

The task of the second rendering pass is to find an acceptable estimate for the true color value μ. A common method to determine the parameters of a probability distribution from sample data is maximum-likelihood (ML) estimation. The goal is to calculate the values μ_{ML} and $\mathbf{\Sigma}_{\mathrm{ML}}$ from the sample data which maximize the log-likelihood function $\ln p\left(\mathbf{X} \mid \mu, \mathbf{\Sigma}\right)$. It can be shown that the ML estimate of the mean is obtained by

$$\mu_{\mathrm{ML}} = \frac{1}{7} \sum_{k=1}^{7} \mathbf{x}_k \tag{2}$$

Thus, the estimated color value at a given pixel position in the virtual image is the arithmetic mean of the pixels colors from the seven reference images.

C. Relevance for cognitive processes

For pure view synthesis, the covariance of the sample data is of minor importance and thus is not computed. However, it bears information about the uncertainty about the predicted color and therefore is important for cognitive processes like surprise detection. In [8] we show that if the set of samples $\mathbf{X}$ is augmented by a sample provided by the currently captured image, which is the observation of a cognitive system, surprise can be quantified by the difference between the prior and posterior distribution over the covariance. As shown in [7], surprise is a crucial cue for the direction of human attention to unexpected events. In cognitive systems surprise influences belief states about the environment and action plans in unforeseen situations. Image-based modeling provides a way to rapidly generate pixel-wise surprise triggers which can be further processed by other cognitive instances of a technical system.

IV. RESULTS

In principle, our module for visual camera localization and image-based environment modeling can be integrated into any demonstration scenario in CoTeSys. In this section, we show some test results of our visual navigation algorithm and

the visual output obtained from our image-based modeling technique applied to a household scene.

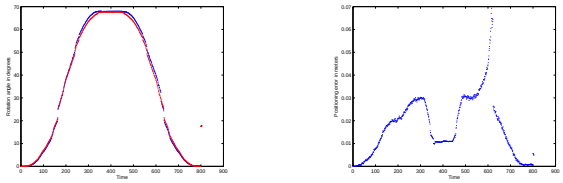
In our studies, we use a KLT-package provided by Stan Birchfield at the Clemson University[2]. Table I compares the processing time of the various KLT implementations (see section II-A). For this test, an AMD® Athlon(tm) 64 X2 Dual Core Processor 3800+ has been used. The result is the mean of 100 tracking runs using 7x7 pixel features. You can see that the processing time of the patch variant does not depend on the image size, while the overhead to extract the patches increases proportionally to the number of features. If there are too many features to track, the processing of the whole image is preferred to prevent that the same image regions of adjacent patches are processed multiple times.

image size	320x240				640x480			
search range	3		10		3		10	
# features	50	150	50	150	50	150	50	150
whole image	22	38	34	59	75	100	110	140
ROIs	17	41	44	99	45	69	91	160
patches	15	46	55	164	15	47	56	170

TABLE I

PROCESSING TIMES (IN MS) OF THE VARIOUS KLT-VARIANTS.

Figure 4 illustrates the results of the visual navigation algorithm (see Section II-C) using images from a 25 Hz VGA video stream. In Figure 4(a) the estimated orientation becomes compared to ground truth, which are the poses of the robot. In Figure 4(b) the translation error is displayed. In the right figure the algorithm detects and removes wrong initialized features as soon as their error becomes large enough. Thus, a robust pose estimation is provided.



(a) Orientation (b) Translation error

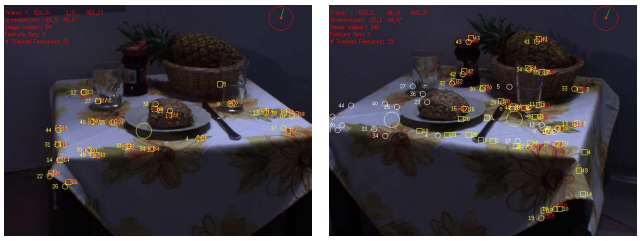
Fig. 4. These figures show the results of a ground truth test, where a camera has been mounted on a high precision KUKA robot. Figure 4(a) shows the orientation measured by the robot (red) and the rotation estimated by the presented algorithm (blue).

For the assistive household scenario in CoTeSys, we envision the acquisition of an image-based model of a typical household environment, which is the basis for cognitive processes like visual surprise detection. Figure 5 (left) shows the acquisition of a set of images with a stereo camera head (640x480 pixels) mounted on a Pioneer 3-DX robot during AUTOMATICA 2008. The robot went along an approximately circular trajectory around a table set with household objects like glasses, plates etc. with the stereo camera looking towards the objects and capturing 213 pairs of images. The set of images was subsampled by a factor of two and processed as described in Sections II and III.

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

Fig. 5. (Left): Acquisition of a set of images with a stereo camera head mounted on a Pioneer 3-DX. (Right): A virtual image rendered from selected reference images. The virtual camera is placed at a position where there is no real camera.

Figure 6 shows two screenshots of the visual navigation algorithm where the tracking and pose estimation results are visualized. The yellow squares are the propagated feature positions, while the small yellow circles are the reprojections of features according to the estimated position and their 3D structure. The red circles stand for the currently tracked features. If there are yellow squares without any circles it means, that these are bad features which could not be tracked. The large circle illustrates the projection of the centroid of a feature set. Somewhere between Figure 6(a) and Figure 6(b) the feature set has changed, because the features moved out of the image. The white circles at the left border of Figure 6(b) stand for the reprojection of the old feature set.



(a) Screenshot image 89 (b) Screenshot image 146

Fig. 6. These are screenshots of the visual navigation algorithm applied to the image sequence, which is also used for the image based environment modeling (see Section III).

The visual localization results for that image sequence are illustrated in Figure 7. The time slots where the camera images could not be stored on the hard disk due to the swapping mechanism, can be easily detected as white gaps in the trajectory. However, swapping does not affect the accuracy of the used algorithm.

A virtual image rendered from several reference cameras with the view synthesis algorithm described Section III is depicted in Figure 5 (right). Note that the virtual camera was placed at a position where no real image had been captured during acquisition. The objects of interest on the table, including the two glasses, are predicted in a photorealistic way.

The left image in Figure 8 illustrates the poses of some reference cameras which were estimated with our visual

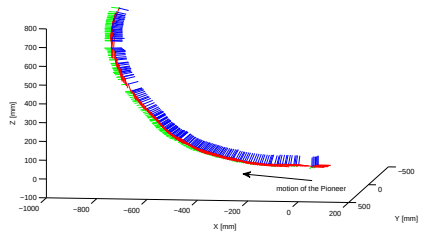


Fig. 7. 3D-plot of the localization results, which where used for the model generation in Section III. The pioneer was programmed to drive a quarter circle around the scene.

localization method in Section II. The reference cameras are represented by a part of their viewing frusta. The seven colored cameras are the ones which were selected with our view selection method in Section III whereas the white ones are not involved in the prediction of the virtual image shown in this figure. The photorealistic virtual view is largely free of visual artifacts which emphasizes that the accuracy of the localization data is acceptable for image-based modeling. The right image in Figure 8 shows a close-up view onto the objects of interest. Thus, even if the position of the virtual camera is chosen far off the trajectory of the reference cameras, acceptable predictions can be achieved with reliable geometry estimates.

Fig. 8. (Left): A virtual image which shows apart from the scene some of the reference cameras which are depicted by a part of their viewing frusta. The colored reference cameras are chosen by the view selection algorithm and contribute to rendering this virtual image. (Right): A close-up virtual view onto the objects of interest.

V. CONCLUSION AND FUTURE WORK

In this work, we presented an approach for image-based environment modeling for cognitive technical systems. A novel visual localization technique working on stereo images is applied to a probabilistic approach for image-based view synthesis. Experimental results show that this method provides photorealistic predicted images, even in challenging real-world household environments with glasses. Furthermore, we outlined the possible integration of our work into the demonstration scenarios in CoTeSys.

Even though the visual localization is quite accurate and robust, we still have many ideas how to improve its modules. There are also some plans to integrate the image-based view prediction in the popular player/stage/gazebo-project[3]. Further, we want to build a better geometric model to provide an object segmentation in the 3D space. Combined with the visual representation of the world, this is supposed to lead to a human like, domain independent basis for every high level module as for example a surprise trigger or object classification.

[3]http://playerstage.sourceforge.net/

VI. ACKNOWLEDGEMENT

We thank the German Aerospace Center (DLR) for making available a Pioneer 3-DX for our experiments.

REFERENCES

[1] H.-Y. Shum, S.B. Kang and S.-C. Chan, Survey of Image-Based Representations and Compression Techniques, *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, no. 11, 2003, pp. 1020-1037.

[2] R.T. Collins, "A Space-Sweep Approach To True Multi-Image Matching", *in Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, San Francisco, USA, 1996, pp. 358-363.

[3] P. Felsenszwalb and D. Huttenlocher, "Efficient Belief Propagation for Early Vision", *in Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, Washington, DC, USA, 2004, pp. I261-I268.

[4] M. Garland and P.S. Heckbert, "Simplifying Surfaces with Color and Texture Using Quadric Error Metrics", *in Proc. IEEE Conf. on Visualization*, Durham, USA, 1998, pp. 263-269.

[5] C. Buehler et al., "Unstructured Lumigraph Rendering", *in Proc. Int. Conf. on Computer Graphics and Interactive Systems*, Los Angeles, USA, 2001, pp. 425-432.

[6] D. Burschka, G. Hirzinger and E. Steinbach, "Perception-Driven Geometry- and Image-Based Environment Modeling for Cognitive Aerial Vehicles", *Interim report on CoTeSys project 208*, 2007.

[7] L. Itti and P. Baldi, Bayesian Surprise Attracts Human Attention, *in Adv. in Neural Information Processing Systems*, vol. 19, 2006, pp. 547-554.

[8] W. Maier, E. Mair, D. Burschka and E. Steinbach, "Surprise Detection and Visual Homing in Cognitive Technical Systems", submitted to *1st International Workshop on Cognition for Technical Systems*, Munich, Germany, 2008.

[9] Bruce D. Lucas and Takeo Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. *in International Joint Conference on Artificial Intelligence*, pages 674-679, 1981.

[10] Carlo Tomasi and Takeo Kanade. Detection and Tracking of Point Features. *in Carnegie Mellon University Technical Report CMU-CS-91-132*, April 1991.

[11] Jianbo Shi and Carlo Tomasi. Good Features to Track. *in IEEE Conference on Computer Vision and Pattern Recognition*, pages 593-600, 1994.

[12] Stan Birchfield. Derivation of Kanade-Lucas-Tomasi Tracking Equation. *Unpublished*, May 1996.

[13] P Kovesi. Image features from phase congruency. *in Videre : Journal of Computer Vision Research*, 1(3):1 27, 1999.

[14] S. Avidan. Support vector tracking. *in IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2001.

[15] D. Nistr, An efficient solution to the five-point relative pose problem, *in IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(6):756-770, June 2004.

[16] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, *in Cambridge University Press*, 2000.

[17] D. Nistr, A Minimal solution to the generalised 3-point pose problem, *in Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004)*, Volume 1, pages 560-567, 2004.

[18] Darius Burschka and Gregory D. Hager. V-GPS - Image-Based Control for 3D Guidance Systems. *in Proc. of IROS, pages 1789-1795*, October 2003.

[19] Darius Burschka and Gregory D. Hager. V-GPS(SLAM): - Vision-Based Inertial System for Mobile Robots. *in Proc. of ICRA*, pages 409-415, April 2004.

[20] Herbert Bay, Tinne Tuytelaars, Luc Van Gool, "SURF: Speeded Up Robust Features", *in Proc. of the ninth European Conference on Computer Vision*, May 2006.