Accurate Shape-based 6-DoF Pose Estimation of Single-colored Objects

Pedram Azad, Tamim Asfour, Rüdiger Dillmann

Institute for Anthropomatics, University of Karlsruhe, Germany

azad@ira.uka.de, *asfour@ira.uka.de*, *dillmann@ira.uka.de*

***Abstract*— The problem of accurate 6-DoF pose estimation of 3D objects based on their shape has so far been solved only for specific object geometries. Edge-based recognition and tracking methods rely on the extraction of straight line segments or other primitives. Straight-forward extensions of 2D approaches are potentially more general, but assume a limited range of possible view angles. The general problem is that a 3D object can potentially produce completely different 2D projections depending on the view angle. One way to tackle this problem is to use canonical views. However, accurate shape-based 6-DoF pose estimation requires more information than matching of canonical views can provide. In this paper, we present a novel approach to 6-DoF pose estimation of single-colored objects based on their shape. Our approach combines stereo triangulation with matching against a high-resolution view set of the object, each view having associated orientation information. The errors that arise from separating the position and orientation computation in first place are corrected by a subsequent correction procedure based on online 3D model projection. The proposed approach can estimate the pose of a single object within 20 ms using conventional hardware.**

I. Introduction

Shape-based pose estimation of 3D objects has so far been addressed mostly for specific object geometries. Edge-based recognition and tracking methods rely on the extraction of straight line segments or other primitives [1], [2], [3], [4], [5], [6]. Extensions of 2D approaches such as the generalized Hough transform [7] or geometric hashing [8], the latter for practical application also relying on the extraction of primitives or interest points, in addition assume a limited range of possible view angles, as it is also the case in [9].

It is an accepted fact that 3D shapes can in general not be represented by a single 2D representation [10]. The reason is that a 3D object can potentially produce completely different 2D projections depending on the view angle, as illustrated in Fig. 1. One way to tackle this problem is to use *canonical views*, which were introduced by the biological vision community [11] and later became of interest in the computer vision community [12], [13]. The general idea of canonical views is to represent an object by a reduced number of views that are sufficient to cover all possible appearances of the object. A suitable data structure for storing and arranging such views is an aspect graph; a survey is given in [14]. However, such representations are mainly used for recognition and a rather coarse localization of the object; an accurate shape-based 6-DoF pose estimation requires more information than matching of canonical views can provide and must be regarded as a separate problem.

Other approaches model the appearance of an object by one or several 2D contours. Similarly to canonical views, the full pose of the object, i.e. rotation and translation in 3D space, cannot be derived accurately on the basis of deformable 2D contours directly.

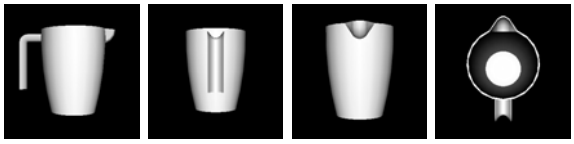

Fig. 1. Different views of a measuring cup.

Traditional model-based recognition and pose estimation methods rely on relatively simple object geometries. Straight line segments can be mapped very efficiently to the image and correspondences between 2D points or lines and 3D model points can be established easily. These correspondences build the input to an optimization approach. However, when dealing with more complex shapes, such approaches become inapplicable. In particular, curved surfaces can be modeled accurately only by a multiplicity of polygons, leading to a substantially higher computational effort for 2D projection. Furthermore, the contour of the object cannot be expressed by straight line segments of the model, making both contour and correspondence computation a complicated and computationally expensive task.

In the recent past, *global* appearance-based recognition and pose estimation methods have become less popular; the trend goes toward *local* appearance-based methods using point features (e.g. [15], [16]) or region-based features (e.g. [17]). However, such approaches are only applicable for objects that exhibit such local features. For single-colored objects (e.g. Fig. 1) this is not the case.

In the following, a novel approach to the problem of shape-based 6-DoF pose estimation of globally segmentable objects will be presented. As the focus is on accurate pose estimation, the segmentation routine used for pre-processing is not of particular interest. Throughout the experiments, a simple color segmentation method operating in HSV color space was utilized, which achieves very good segmentation results using fixed color models.

We will first briefly summarize our previous work on this problem in Section II, and then present our novel pose correction method in Section III. Results of an experimental evaluation with simulated image data as well as real image data from the humanoid robot ARMAR-III [18] operating in a kitchen environment are given in Section IV.

II. Basic Approach

Our basic approach to recognition and pose estimation of single-colored objects has been presented in [19] and is summarized in this section, as it builds the starting point for the subsequent improvements.

Color blobs are segmented and matched between the left and right view of a stereo camera system. Region candidates that result from this step are normalized to a fixed size and matched against a view database. To achieve a higher robustness, the matching is performed on the basis of gradient images of the normalized views. The matching is speeded up by reducing the normalized views of size 64×64 to 64 dimensions using PCA (see [20] for an analysis of the number of dimensions); the best matching view is computed as the nearest neighbor in the eigenspace. In the following, the basic 6-DoF pose estimation method is explained, which builds the basis for the correction procedure explained in Section III.

Ideally, for appearance-based 6-DoF pose estimation with respect to a rigid object model, for each object, training views would have to be acquired in the complete six-dimensional space i.e. varying orientation *and* position. The reason for this is that not only the orientation influences the projected appearance of an object, but also the translation does. In other words, an object with the same orientation appears in a different way if its position changes, as illustrated in Fig. 2.

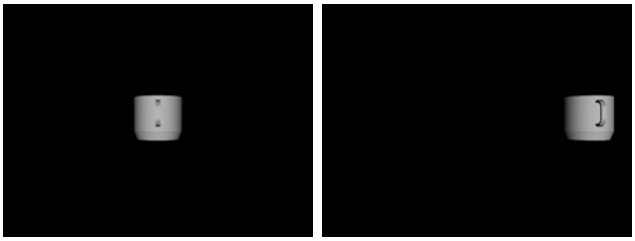

Fig. 2. Illustration of the influence of the position on the appearance. The views were generated with the same orientation of the object, only the position was modified.

However, in practice it is impossible to store views covering the full six-dimensional space of possible object poses. Therefore, we solve the problem by calculating position and orientation independently in first place.

The basis for the calculation of the position is the result of stereo triangulation between the centroids of the matched regions in the left and the right image. However, the result varies with the view of the object. As shown in [19], a 3D correction vector $\mathbf{c}_t$ can be defined, which is added to the triangulation result during the pose estimation process.

As already discussed, the projected appearance of an object is influenced by both the orientation and the position of the object. Therefore, the introduced position correction vector is only accurate if the object is located at the same position it was trained with when computing and storing the position correction vector. In all other cases, the correction can only be an approximation. An accurate solution to this problem is presented in Section III-B.

The orientation of an object is calculated on the basis of the rotational information that was stored with each view during the acquisition process. However, assuming that the translation for each stored view was zero in the x- and y-component – the training views are generated in the center of the left camera image – causes an error if the object is not located in the center of the image.

As a conclusion, in the basic approach presented in this section, for the case of the object being located in the center of the left camera image, a relatively accurate pose estimate is calculated as follows:

$$\mathbf{t} = f(\mathbf{p}_l, \mathbf{p}_r) + \mathbf{c}_t \quad (1)$$

$$R = R_0 \quad (2)$$

where $\mathbf{p}_l$, $\mathbf{p}_r$ denote the centroids of the matching regions in the left and right image, $f : \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow \mathbb{R}^3$ is the transformation performing the stereo triangulation for two matching centroids, $\mathbf{c}_t$ is the position correction vector, and R_0 is the stored rotation for the matched view, given as a rotation matrix. Precise correction methods for the general case are introduced in the Sections III-A and III-B. Note that another way for achieving a high precision without further corrective calculation is to foveate the object of interest in the left camera image. By doing this, the pose of only one object at a time could be estimated accurately, which, however, would be sufficient for grasping a single object with a robot that has an *active* head.

III. Pose Correction

The approach that has been proposed in Section II is purely appearance-based i.e. no model is needed for the acquisition of the views of the object. However, the pose estimate is error-prone since the position and orientation are treated separately. As already mentioned, an accurate pose estimate is computed only when the object is located at the same position it was trained at.

For computing an accurate pose estimate in the general case, knowledge about the 3D shape of the object is necessary. In the following, we will present a pose correction method that utilizes a 3D model of the object in order to compute and compensate the pose error online. Furthermore, using a 3D model offers the benefit that the view set can be generated in simulation, not requiring an expensive hardware setup for view acquisition.

A. Orientation Correction

As already discussed, the core idea of the pose estimation procedure is to decouple the determination of the position and the orientation in first place, in order to make the problem tractable. However, this decoupling has the effect that the rotation information stored with each view is only valid if the object is recognized at the same position in the image it was learned with. The influence of the object position on its appearance is illustrated in Fig. 2.

The error that is caused by a deviating position can be corrected analytically. Let the position of the object in the training view be $\boldsymbol{t}_l$ and the position in the current view be

$\boldsymbol{t}$, both being specified in the camera coordinate system. The idea is to assume a spherical image sensor and to calculate the rotation R_c that is necessary to rotate the projection of $\boldsymbol{t}_l$ to $\boldsymbol{t}$, with the origin of the rotation being the projection center as the center of the image sensor sphere.

An object at the position $\boldsymbol{t}_l$ with its orientation being described by the rotation matrix R_0 produces a specific appearance when projected on the spherical image sensor. If rotating the object with the rotation matrix R_c, i.e. rotating its position *and* orientation, then it produces the exact same appearance at a different position of the image sphere.

Thus, when retrieving the stored rotation information R_0, assuming that the object producing the current view is located at the position $\boldsymbol{t}_l$, the object must be rotated around the projection center to its true position $\boldsymbol{t}$ to produce the same appearance at the actual position of the image sensor. Therefore, the only thing that has to be done for correcting the rotation is to apply the corrective rotation R_c to the retrieved rotation R_0:

$$R = R_c\, R_0 \tag{3}$$

with R_c being defined by:

$$R_c\, \boldsymbol{t}_l = \boldsymbol{t} \tag{4}$$

In order to be able to compute the corrective rotation R_c, the vectors $\boldsymbol{t}_l$ and $\boldsymbol{t}$ must have been normalized to the same length beforehand. This normalization is legitimate, since a differing distance of the object to the projection center essentially causes a different size of its appearance, when being moved along the principal axis. Note that many rotation matrices R_c satisfy the condition of Eq. (4), but that rotation matrix R_c is searched that directly transfers $\boldsymbol{t}_l$ to $\boldsymbol{t}$, without using the undetermined degree of freedom. The matrix R_c is thus computed by RotationMatrixAxisAngle($\boldsymbol{t_l} \times \boldsymbol{t}$, Angle($\boldsymbol{t_l}$, $\boldsymbol{t}$, $\boldsymbol{t_l} \times \boldsymbol{t}$)) (see [20], Appendix A.4). The effect of the orientation correction is shown in Fig. 3 for an example scene.

In our experiments, we produced the training views so that the object was located at the center of the left camera image. The world coordinate system was the camera coordinate system of the left camera. Therefore, the position of the object was $\boldsymbol{t} = (0,\ 0,\ z)^T$ throughout the acquisition of the training views, with z being constant. The origin of the object coordinate system was set to its center of mass.

Fig. 3. Effect of the orientation correction on an example scene. Left: no correction. Right: with orientation correction.

B. Position Correction

As already explained, the position of the object is computed on the basis of the triangulated centroids of the segmented regions using a calibrated stereo system. However, the triangulation result is error-prone because of three reasons:

1) The position of the triangulated 3D point in the object coordinate system differs depending on the view.
2) The centroids of the 2D regions in general do not originate from projection of the same object point.
3) The projection on a planar image sensor causes a deformed image of the object.

The reasons (1) and (2) effectively lead to the same problem: The relationship between the triangulated point and the object is unknown. Intuitively one might first think that triangulating the centroids of the 2D regions results in the computation of a point on the surface of the object. While this is approximately true for planar objects – it is not true for 3D objects. Even for the optimal case of a sphere, the triangulated point does not lie on the surface and varies depending on the position of the sphere, as shown in [20].

Already in the basic approach, a rudimentary position correction is applied, as explained in Section II. However, because of the three explained reasons, it is clear that an accurate position correction for the general case depends on the geometry of the object, the stereo camera setup and the views in both images, and therefore on the position and orientation of the object. Since the exact position $\boldsymbol{t}$ of the object is not known, the task is to find a function f that calculates $\boldsymbol{t}$, given the position estimate $\boldsymbol{t}'$ calculated by stereo triangulation and the corrected orientation R (see Section III-A):

$$f(\boldsymbol{t}', R) = \boldsymbol{t} \tag{5}$$

To achieve maximum accuracy, the correction procedure must be performed iteratively, since the position affects the orientation correction, and the orientation affects the position correction. However, in practice the effect of the position correction on the orientation correction is so small that at most two iterations are necessary (see Fig. 6).

The question now is how to find the function f. Our experiments have shown that the attempt to learn f completely fails, even on perfect simulation data. The reason is that f depends on the object geometry. Therefore, a learning procedure would have to implicitly learn the full object geometry based on a set of pairs $\{(\boldsymbol{t}'^{(i)}, R^{(i)}), \boldsymbol{t}^{(i)}\}$, which is practically impossible.

Our approach is to simulate the present situation at runtime, including the stereo camera setup, the object geometry, and the estimated object pose in simulation. By doing this, the position correction vector can be computed for the actually present conditions. The stereo camera system is simulated by using the intrinsic and extrinsic camera parameters from the calibrated stereo camera system that is actually used. Since neither software nor hardware rendering take into account lens distortions, the input images are undistorted after being captured from the camera i.e. all calculations are

performed on the undistorted images. The 3D model of the object that was used for producing the training views is used for simulating the object at run-time.

The effects of reason (3) are negligible in practice, since the deformations due to the projection on a planar image sensor are relatively small. Nevertheless, the resulting errors are handled by our approach as well, since rendering produces the same deformations.

Finally, the full pose estimation procedure for a segmented region pair and a given object representation is summarized in Algorithm 1 in pseudo code. Here, $\boldsymbol{c}_l, \boldsymbol{c}_r$ denote the centroids of the left and right region. The rotation matrix that was stored with the best matching view is denoted by R_0, and *model* denotes the 3D model of the object of interest.

Algorithm 1 CalculatePoseSegmentable($\boldsymbol{c}_l$, $\boldsymbol{c}_r$, R_0, *model*) $\rightarrow R, \boldsymbol{t}$

1) Calculate the position estimate by stereo triangulation: $\boldsymbol{t}_0 \leftarrow$ Calculate3DPoint($\boldsymbol{c}_l$, $\boldsymbol{c}_r$)
2) Set $\boldsymbol{t} := \boldsymbol{t}_0$.
3) Perform the steps 4–7 k times:
4) Calculate the corrected orientation R on the basis of R_0 and $\boldsymbol{t}$ using the Eqs. (3) and (4).
5) Simulate the current situation by applying R and $\boldsymbol{t}$ to the 3D *model* of the object, yielding the simulated camera images I'_l, I'_r.
6) Perform stereo triangulation with the centroids of the object regions in the simulated views in I'_l, I'_r, yielding $\boldsymbol{t}'$.
7) Compute the position correction by the update $\boldsymbol{t} := \boldsymbol{t}_0 + \boldsymbol{t} - \boldsymbol{t}'$.

The simulated images I'_l and I'_r are understood as binary images, which can be directly computed by conventional graphics hardware by turning off shading. The formula $\boldsymbol{t}_0 + \boldsymbol{t} - \boldsymbol{t}'$ from the last step can be explained as follows. In the simulation procedure, the ground-truth position of the object is $\boldsymbol{t}$. The computed triangulation result on the simulated images is $\boldsymbol{t}'$, thus the position correction vector is $\boldsymbol{t}_c = \boldsymbol{t} - \boldsymbol{t}'$. Finally, the corrected position for the real setup reads $\boldsymbol{t}_0 + \boldsymbol{t}_c = \boldsymbol{t}_0 + \boldsymbol{t} - \boldsymbol{t}'$.

As already mentioned, the accuracy of the proposed pose estimation procedure can be increased by performing $k > 1$ iterations. In practice, the second iteration leads to an observable improvement with subsequent immediate convergence, as is shown in Fig. 6. The effect of the position correction is illustrated in Fig. 4 for an example scene. Only one iteration, i.e. $k = 1$, was used.

IV. Experimental Results

A. Accuracy

In the following, the accuracy of the proposed pose estimation method was measured on simulated data so that ground truth information was available. The results for variations of the z-coordinate within a range of $[500, 1000]$ (mm) are illustrated in Fig. 5. Only the critical degrees of freedom are shown. The main problem is the rotation angle θ_y around the main axis of the measuring cup. A correlation between this θ_y-error and the z-error can be observed, which results from the fact that the position correction formula depends on the orientation information. Since from a side-view, small variations of the angle θ_y result in very similar views[1], the likelihood that a slightly wrong view is matched is relatively high, which explains the errors for the angle θ_y of up to $10°$. Note that this problem cannot be solved by using more iterations of the pose correction procedure. The cup does not suffer from this problem, as the y-axis is its rotational symmetry axis.

Fig. 4. Effect of the position correction on an example scene. Left: with orientation correction only. Right: with orientation and position correction.

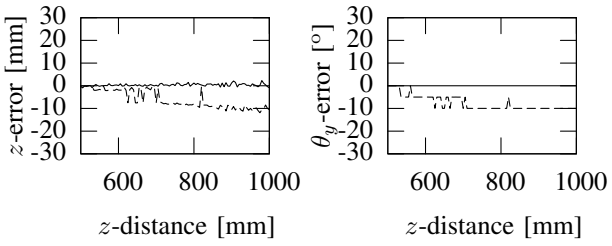

Fig. 5. Accuracy of 6-DoF pose estimation depending on the z-coordinate. The solid line indicates the result for the cup, the dashed line the result for the measuring cup.

For the subsequent experiments, a single scalar value was computed for the overall 3D error rather than presenting the errors of all six degrees of freedom independently. For this purpose the surface of the object was sampled uniformly with a point distance of 2 mm, resulting in a 3D point cloud. In order to compute an overall 3D error, the ground truth pose was applied to the point cloud as well as the computed pose, and the average Euclidean distance between corresponding points was computed.

The effect of the pose correction formula (see Sections III-A and III-B) is illustrated in Fig. 6. As can be seen, the first iteration yields a significant improvement of the accuracy and a further improvement can be observed for the second iteration, after which the error converges.

The effect of different rotational resolutions of the training views was evaluated by the example of a cup with a rotational symmetry axis. The high resolution view set was acquired

[1]Note that it is a natural, object-specific problem that rotations around certain degrees of freedom produce only small variations of the appearance.

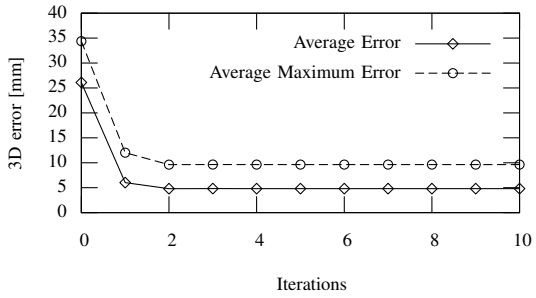

Fig. 6. Accuracy of 6-DoF pose estimation depending on the number of iterations of the correction procedure.

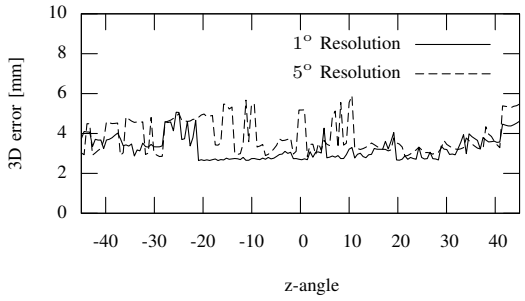

Fig. 7. Accuracy of 6-DoF pose estimation depending on the rotational resolution of the learned views for a cup.

with a resolution of 1° (6,461 views). The second view set was acquired with a resolution of 5° (285 views). As can be seen, the higher resolution leads to an improvement of approx. 2–3 mm.

In the Fig. 8, the errors for 1,000 random trials with a cup and a measuring cup were evaluated, within a range of $[-100, 100] \times [-100, 100] \times [500, 1000]$ for the position, and $[0°, 70°] \times [45°, 135°] \times [-45°, 45°]$ for the angles around the x-, y-, and z-axes. The average error refers to the average 3D Euclidean distance of the sampled points, after application of the ground-truth pose and the computed pose, respectively. The maximum error is the maximum distance that occurred for the set of all sampled points. A deviation between the average error and the maximum error indicates errors of the orientation estimation.

As can be seen, the measuring cup produces greater errors, with the average error being below 10 mm and the maximum error below 20 mm for over 95% of the trials. For the cup, the average error is below 5 mm and the maximum error is below 10 mm for over 95% of the trials, i.e. is more accurate by a factor of approx. 2.

In Table I, the standard deviations for the estimated poses for static objects using real image date are given. The standard deviations have been calculated for 100 frames. The units are [mm] and [°], respectively.

B. Runtime

In Table II, the runtimes for the different processing stages are given for the recognition and pose estimation

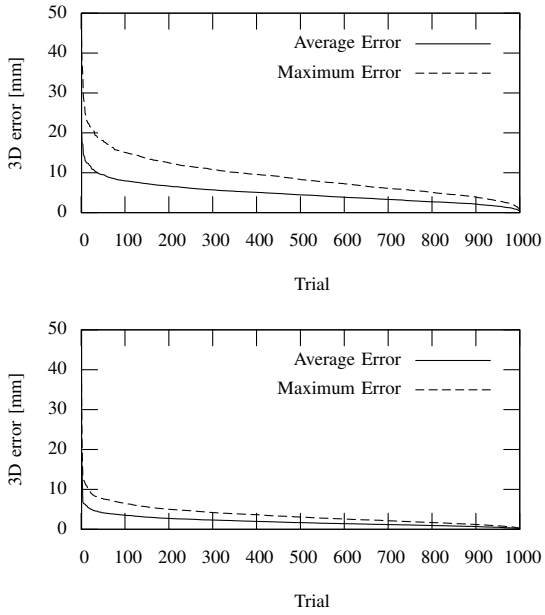

Fig. 8. Accuracy of 6-DoF pose estimation for 1,000 random trials with a measuring cup (top) and a cup (bottom). The errors were sorted in decreasing order in order to illustrate the error distribution.

	x	y	z	θ_x	θ_y	θ_z
Measuring cup	0.028	0.049	0.41	0.0034	0.0023	0.0026
Cup with handle	0.033	0.053	0.0017	0.0034	0.0022	0.0027
Cup	0.045	0.029	0.14	0.0018	-	0.0017
Plate	0.049	0.071	0.37	0.0021	-	0.0017

TABLE I

STANDARD DEVIATIONS FOR STATIC OBJECTS.

of one object. The runtime is proportional to the number of training views of the object. For a full scene analysis without temporal information, so that a region of interest is not known, the runtime is also proportional to the number of potential regions extracted from the image. The processing times are given for the example of the measuring cup, which was trained with 15,675 views. Matching 10,000 views, which were compressed to 64 dimensions with PCA, takes approx. 1.3 ms on a 3 GHz single core CPU.

The computation time of the pose correction procedure depends on the graphics card and the graphics driver, since the stereo setup is simulated online with the aid of OpenGL for this purpose. The time-critical part is not the rendering itself, but the transfer from the rendered data to the main memory. With the use of pixel buffers, rendering and transfer for a 640×480 grayscale image was achieved within approx. 7 ms on a Windows PC. The pose correction including simulation of the stereo pair thus takes 14 ms and 28 ms for $k = 2$ iterations of the pose correction. Grasping experiments with the humanoid robot ARMAR-III have proved that a single iteration is fully sufficient for grasp execution [21].

The system was implemented using the Integrating Vision Toolkit (IVT, `http://ivt.sourceforge.net`).

Finally, Fig. 9 shows visualizations of the results for sev-

	Time [ms]
Color segmentation	4
Matching	2
Pose correction	14
Total	**20**

TABLE II

PROCESSING TIMES FOR THE PROPOSED RECOGNITION AND POSE ESTIMATION SYSTEM. ONE ITERATION OF THE POSE CORRECTION PROCEDURE WAS USED. THE TESTS WERE PERFORMED ON A PENTIUM IV, 3 GHZ.

eral example scenes seen by the humanoid robot ARMAR-III operating in a kitchen environment.

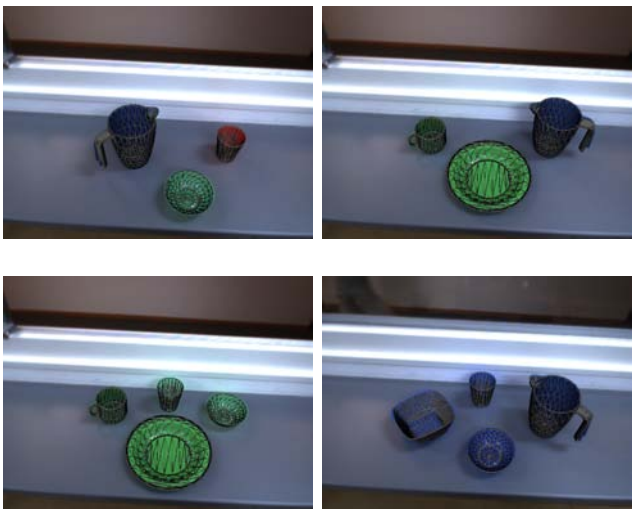

Fig. 9. Exemplary results with the proposed object recognition and pose estimation system. The computed object pose has been applied to the 3D object model and the wireframe model has been overlaid.

V. DISCUSSION AND OUTLOOK

We have presented an accurate 6-DoF pose estimation method for single-colored objects. For these kinds of objects, state-of-the-art approaches are not applicable, since the only feature suitable for pose estimation is their shape. It was shown that by combining stereo triangulation and appearance-based matching of globally segmented views, the 6-DoF pose can be estimated. For this purpose we have introduced a pose correction method that utilizes online projection of a 3D model of the object. With the proposed method, a single object can be recognized, including pose estimation, within 20 ms using conventional hardware.

Our future work will focus on incorporating a more general segmentation approach, which also allows to deal with occlusions. Furthermore, the accuracy of the estimated pose could be increased further by subsequent online generation of a small view set at full resolution in the vicinity of the estimated pose.

ACKNOWLEDGMENT

The work described in this paper was partially conducted within the EU Cognitive Systems projects PACO-PLUS (IST-FP6-IP-027657) and GRASP (IST-FP7-IP-215821) funded by the European Commission, and the German Humanoid Research project SFB588 funded by the German Research Foundation (DFG: Deutsche Forschungsgemeinschaft).

REFERENCES

[1] D. G. Lowe, "Three-Dimensional Object Recognition from Single Two-Dimensional Images," *Artificial Intelligence*, vol. 31, no. 3, pp. 355–395, 1987.

[2] ——, "Robust Model-based Motion Tracking through the Integration of Search and Estimation," *International Journal of Computer Vision (IJCV)*, vol. 8, no. 2, pp. 113–122, 1992.

[3] C. G. Harris and C. Stennett, "3D object tracking at video rate – RAPiD," in *British Machine Vision Conference (BMVC)*, Oxford, UK, 1990, pp. 73–78.

[4] M. Armstrong and A. Zisserman, "Robust Object Tracking," in *Asian Conference on Computer Vision (ACCV)*, vol. 1, Singapore, 1995, pp. 58–61.

[5] E. Marchand, P. Bouthemy, F. Chaumette, and V. Moreau, "Robust Real-Time Visual Tracking using a 2D-3D Model-based Approach," in *IEEE International Conference on Computer Vision (ICCV)*, Kerkyra, Greece, 1999, pp. 262–268.

[6] G. Klein and D. Murray, "Full-3D Edge Tracking with a Particle Filter," in *British Machine Vision Conference (BMVC)*, vol. 3, Edinburgh, UK, 2006, pp. 1119–1128.

[7] D. H. Ballard, "Generalizing the Hough Transform to Detect Arbitrary Shapes," *Pattern Recognition*, vol. 13, no. 2, pp. 111–122, 1981.

[8] Y. Lamdan and J. H. Wolfson, "Geometric Hashing: A General and Efficient Model-based Recognition Scheme," in *IEEE International Conference on Computer Vision (ICCV)*, Tampa, USA, 1988, pp. 238–249.

[9] C. Wiedemann, M. Ulrich, and C. Steger, "Recognition and Tracking of 3D Objects," *Lecture Notes in Computer Science*, vol. 5096, pp. 132–141, 2008.

[10] D. G. Lowe, *Perceptual Organization and Visual Recogntion*. Kluwer Academic Publishers, 1985.

[11] S. E. Palmer, E. Rosch, and P. Chase, "Canonical Perspective and the Perception of Objects," *In Attention and Performance IX*, pp. 135–151, 1981.

[12] D. Weinshall and M. Werman, "On View Likelihood and Stability," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 19, no. 2, pp. 97–108, 1997.

[13] T. Denton, M. F. Demirci, J. Abrahamson, A. Shokoufandeh, and S. Dickinson, "Selecting Canonical Views for View-based 3-D Object Recognition," in *International Conference on Pattern Recognition (ICPR)*, Cambridge, UK, 2004, pp. 273–276.

[14] R. D. Schiffenbauer, "A Survey of Aspect Graphs," Polytechnic University, New York City, USA, Tech. Rep. TR-CIS-2001-01, 2001.

[15] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision (IJCV)*, vol. 60, no. 2, pp. 91–110, 2004.

[16] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," in *European Conference on Computer Vision (ECCV)*, Graz, Austria, 2006, pp. 404–417.

[17] S. Obdrzalek and J. Matas, "Object Recognition using Local Affine Frames on Distinguished Regions," in *British Machine Vision Conference (BMVC)*, vol. 1, Cardiff, UK, 2002, pp. 113–122.

[18] T. Asfour, K. Regenstein, P. Azad, J. Schröder, N. Vahrenkamp, and R. Dillmann, "ARMAR-III: An Integrated Humanoid Platform for Sensory-Motor Control," in *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, Genova, Italy, 2006, pp. 169–175.

[19] P. Azad, T. Asfour, and R. Dillmann, "Combining Apperance-based and Model-based Methods for Real-Time Object Recognition and 6D Localization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, 2006, pp. 5339–5344.

[20] P. Azad, "Visual Perception for Manipulation and Imitation in Humanoid Robots," Ph.D. dissertation, Universität Karlsruhe (TH), Karlsruhe, Germany, 2008. [Online]. Available: http://digbib.ubka.uni-karlsruhe.de/volltexte/1000011294

[21] N. Vahrenkamp, S. Wieland, P. Azad, D. Gonzalez, T. Asfour, and R. Dillmann, "Visual Servoing for Humanoid Grasping and Manipulation Tasks," in *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, Daejeon, Korea, 2008, pp. 406–412.