Acquisition of a Dense 3D Model Database for Robotic Vision

Muhammad Zeeshan Zia, Ulrich Klank, and Michael Beetz

***Abstract*— Service Robots in real world environments need to have computer vision capability for detecting a large class of objects. We discuss how freely available 3D model databases can be used to enable robots to know the appearance of a wide variety of objects in human environments with special application to our Assistive Kitchen. However, the open and free nature of such databases pose problems for example the presence of incorrectly annotated 3D models, or objects for which very few models exist online. We have previously proposed techniques to automatically select the useful models from the search result, and utilizing such models to perform simple manipulation tasks. Here, we build upon that work, to describe a technique based on Morphing to form new 3D models if we only have a few models corresponding to a label. However, morphing in computer graphics requires a human operator and is computationally burdensome, due to which we present our own automatic morphing technique. We also present a simple technique to speed the matching process of 3D models against real scenes using Visibility culling. This technique can potentially speed-up the matching process by 2-3 times while using less memory, if we have some prior information model and world pose.**

I. INTRODUCTION

For intelligent service robots to make their way into widespread use, computer vision capabilities are needed to detect many different classes of objects. These object classes, for example cups, pots, spoons themselves have members that vary widely in their appearance. Thus, it is very hard to train robots to recognize even a small subset of objects present in a human environment. One possible solution for this is proposed in our previous work [1], where we present how public 3D model databases available on the internet can be used to obtain various annotated 3D models given abstract instructions. Here we extend those techniques to allow new models to be generated (using morphing) when very few models of a given class are available and utilize a technique called visibility culling from 3D game design to speed up the matching process against a real scene.

Moreover, usually a service robot's task will not end at just detecting the presence of an object but also involve manipulating the object. For this task, 3D models provide an opportunity for the robot to have complete information about the hidden geometry of the object, which can result in efficient manipulation. Thus such models allow the robot to not only localize the objects in the real world but also guess their weight, center of mass and good grasping points.

This work was supported by MVTec Software GmbH, München and by the cluster of excellence Cognitive Technical Systems (CoTeSys).

M. Z. Zia, U. Klank and M. Beetz are with the Intelligent Autonomous Systems (IAS) group, Technische Universität München, Boltzmannstr. 3, 85748 Garching bei München {zia, klank, beetz}@in.tum.de

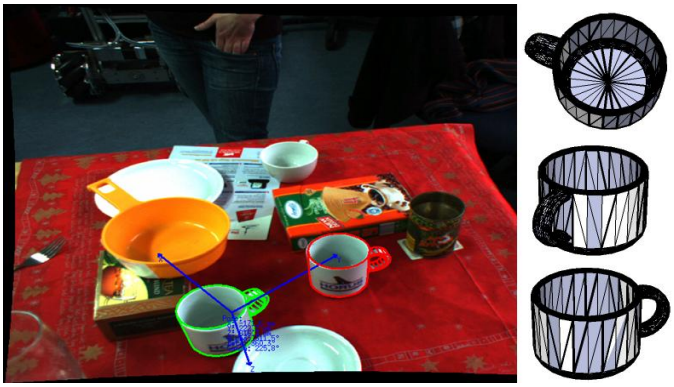


Fig. 1. Multiple views of a 3D mug model matched against a scene. A few views of this model are shown for illustrative purposes.

We have developed this system keeping in particular view the needs of our Assistive Kitchen environment [2]. Fig. 1 shows a scene where two cups were localized using a 3D mug model obtained from the internet.

We propose an automatic integrated robot object recognition system whose operation takes place as follows:

1) The robot is given abstract instructions, out of which it extracts names of objects.
2) It searches for geometric 3D object models annotated with these names from an open online database (in our case, Google's 3D Warehouse) and downloads them.
3) Due to the open nature of such databases, every search usually returns some garbage models also. Thus we apply a classification algorithm to select the best models automatically.
4) If only few models are available, new 3D models are created by scaling/registering the models and then performing automatic morphing. It must be emphasized that this whole process takes place completely automatically, whereas usually morphing algorithms require significant input from a human operator.
5) For making the later matching step efficient the system reduces the model's complexity based on context information
6) Given these models which do not contain accurate size information, the robot looks for objects in the environment matching these to a sufficient extent.
7) The robot presents the objects it has found together with their class labels and the specialized 3D models computed in the last step in order to get them approved by the robot programmer.
8) Upon approval of the object labels and models the object recognition system computes appearance descriptions using environment tailored feature that enable

visual popout mechanisms in cluttered scene as well as accurate localization using model knowledge.

II. Related Work

Our objective is to localize an object whose name is known but appearance is unknown. We are further interested in manipulating the object. One way to learn relevant information for grasping is to automatically estimate grasping points like suggested in [3] or simply approximating an object with primitives before grasping it [4]. These approaches are limited due to automatic segmentation which is still a problem, especially in range images, see [5].

A database of 3D objects enables many new possibilities such as high level motion planning [6], [7]. Using internet databases automatically was already done [8], but only using 2D information i.e. using images. We on the other hand, extract 3D models from internet databases. The selection of relevant objects is performed using clustering techniques based on the shape distribution function proposed by [9] and the distance measures tested in [10] to find out the major cluster of the resulting models.

To be able to perform completely automatic morphing, we need to register the models. One technique for doing this is introduced in [11].

Almost all existing algorithms for morphing work in two phases, the first one being that of establishing correspondences and the second is of interpolating between corresponding points[12]. For the first stage, these algorithms usually project both 3D models to the unit sphere or simplify the models considerably and then project the original models to these simplified models. Also this stage requires input from the human operator as to some basic corresponding points/edges; and most algorithms available in the literature only work for models with genus-0 geometry (i.e. models which are closed). Thus, we need a simpler and faster algorithm that can handle unclosed geometry, and exploit the fact that we only need to morph between objects of similar basic structure.

The last step of our problem is to match many 3D objects in 2D images. This problem was solved before and there are many different state of the art methods like [13] which is based on many views of a 3D object. We are using a state of the art 3D shape model matching technique, which is described in [14]. This method matches 3D-CAD-models in an image by simulating the 2D appearance in a shape model generation phase. A range of position relative to the camera must be specified, which can only be done if information about the scene is available. In our kitchen scenario this is given, since we assume all static parts to be approximately known by earlier 3D-Laser scans or similar techniques and are semantically labeled via methods like [15]. However the technique described in [14], first requires computationally expensive pre-processing on the 3D model which limits the applicability of this approach to small models. Thus, we utilize the technique of Visibility Culling used by 3D rendering engines to speed up this preprocessing. Some very efficient techniques for culling are discussed in [16].

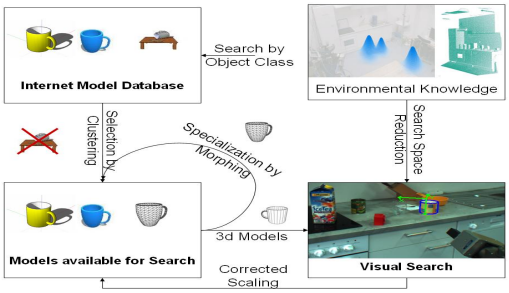


Fig. 2. Overview of the system

III. Method

The key contributions of our work are novel preprocessing mechanisms that are embedded in a technique that allows robots to localize objects only knowing their name and some restrictions about their position in a partially known environment. This is particularly interesting for a learning phase once the service robot is deployed in a new environment. We introduce a new morphing algorithm that operates automatically without the need for any manual human input, and takes very little time to produce useful models for matching tasks. Finally we use a very basic Visibility Culling technique called Backface Culling to demonstrate that the matching process can be speeded up by such techniques.

An overview of our method is shown in Fig. 2. These steps are explained in detail below.

A. Object Selection

We pass a search string for example, “mug”, “pot”, or “kettle” to a 3D Model database (Google’s 3D Warehouse in our case), and automatically download the resulting models. However, these 3D Models are in binary format which are converted to an easily understandable format (Collada format). In public databases contain incorrectly annotated models, for example, there are racing cars labelled as spoon, or a stove on which some pots are present labelled as just pot. Thus, before resulting models can be utilized for matching, we need to discard the irrelevant models. To find similarities between 3D models we use the shape distribution function proposed by [9]. We further use k-means introduced by [17] for clustering with $k = 4$ clusters, since we expect maximally 3 types of outliers[1]. We justify this procedure on the assumption (verified on a number of search strings) that most of the models from the internet search are relevant models, and thus we choose the largest cluster as inliers.

B. Forming new models

Often we do not get enough models to perform a successful matching in a scene. Thus we create new models from the ones we already possess, for which we utilize morphing between models, and choose in-between models.

1) Alignment of Models: Morphing process usually requires the presence of a human operator to perform an appropriate scaling, and alignment to make both models of same size and aligned[12]. We use the technique presented by [9] to form a histogram of distances between randomly selected points on the model-surface, and then pick the

TABLE I
MORPHING ALGORITHM

Step 1: Translate along z-axis and scale making the extent on both side equal.
Step 2: For every vertex in the source model, find the nearest point on the surface of the destination model.
Step 3: Introduce this nearest point as a new vertex in the triangulated model, and divide the triangle containing the vertex into three triangles.
Step 4: Store this as a mapping from the vertex in source model to the new vertex in destination model.
Step 5: Repeat step 2-4 reversing the two models.
Step 6: All vertices in both models now have a one-to-one mapping between them, and they are equal in number.
Step 7: Interpolate between the corresponding pairs of points using linear interpolation based on the parameter t.

most commonly occurring distance (actually the middle point of the most frequent bin) and scale that distance to a constant value and the whole model is scaled isotropically. A similar technique is used for finding a statistical centroid and translating both the models so that the centroid lies at their origin.

Next we need to register the models against each other before we can start the morphing process. For this, one technique of interest is the Iterative Closest Point Algorithm (ICP) [11]. However, ICP works for only dense point-clouds; while the 3D models have only vertices (of the triangulated faces) which can be directly used for ICP. These vertices are densely packed at places where a high curvature is present in the model; and very few of these are present for comparatively flat surfaces. Thus, even if the curved surface has a small area, its alignment is given more weight by the ordinary ICP, as compared to a more planar surface with large area - which should not be the case. Thus, we used the technique presented by [9] to form such a dense point-cloud which has a distribution of points proportional to the surface area of the faces. This enables us to run a ICP with equivalent weight to all parts of the object. We called this variant a "Volumetric ICP" [1].

2) Morphing: Morphing is a technique that finds common use in computer animation. It involves transforming from one image or 3D model to another controlled by a parameter t going from 0 to 1. At $t = 0$, the resulting model is the same as the initial source model, while at $t = 1$ the resulting model is the same as the original destination model; whereas at values of $0 < t < 1$, we have a resulting model that is visually "in-between" the source and the destination model. Thus it is different from the original models, while maintain similar basic geometry in case the morphing is performed between objects of similar shape (which is the case of interest to us). We exploit this technique to generate new models when we fail to obtain models that fit the objects in the scene well enough.

Our approach yields particularly good results when the models have rotational symmetry around a line (detecting symmetry in 3D models is a well-studied problem [18]). Many of the models that we find useful for our kitchen environment have such a geometry that we may assume the z-axis to be the principle axis of the models. In the following therefore, we assume that the axis of symmetry is the z-axis. The extent of both models is made equal on both sides of the x-y plane by translating and scaling slightly. In the following, we refer to one of the two models (between whom we perform the morphing) as the source model (corresponding to the morphing parameter $t = 0$) and the other model as the destination model (corresponding to $t = 1$). For each vertex in the source model, we take the x-y plane on which it lies, and intersect the triangles in the destination model with this plane, to obtain the nearest point on the surface of the destination model to the current vertex of the source model (we choose the nearest point out of the nearest points contributed by each plane). This is done by finding the intersection points of the x-y plane on the edges using Plücker lines [19]. The reason behind this major step is that the source and destination models will usually have different levels of detail. If we have two vertices A and B which form an edge, the edge is represented by L; and p represents in homogenous coordinates the x-y plane on which the source vertex s lies, then

$$L = \begin{bmatrix} 0 & A_xB_y - B_xA_y & A_xB_z - B_xA_z & A_x - B_x \\ A_yB_x - B_yA_x & 0 & A_yB_z - B_yA_z & A_y - B_y \\ A_zB_x - B_zA_x & A_zB_y - B_zA_y & 0 & A_z - B_z \\ B_x - A_x & B_y - A_y & B_z - A_z & 0 \end{bmatrix}$$

$$p = \begin{bmatrix} 0 & 0 & 1 & -s_z \end{bmatrix}^T.$$

Then the point of intersection u is given by $u = Lp$, and with another intersection point v on another edge of this face, we can find the nearest point on this line segment from the source vertex using parameter θ (obtained by taking equating the derivative of the distance to zero). If the points u, v, and p are represented in non-homogenous coordinates, then c gives the desired closest point.

$$\theta = \frac{(u-p)(u-v)}{|\, uv \,|}$$

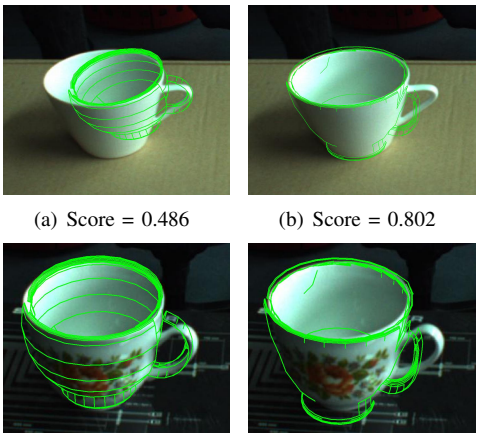
$$c = u + \theta v$$

This nearest point is introduced into the mesh of the destination model as a vertex by dividing the triangle that contains it into three (one new vertex in destination model for each original vertex of source model). We repeat this procedure, reversing the roles of the source and destination model; and store this one-to-one mapping. The introduction of new points into the models can be thought of as increasing the degrees of freedom of the model to take the shape of the other model and still looking pleasant. The last step is to perform a linear interpolation of the vertices from their position in source model to a their final position in the destination model. The complete algorithm is summarized in Table I.

C. Final Detection and speed up using Visibility Culling

The 3D shape matching approach [14] uses 3D models to calculate all necessary shape appearances and searches them in an image. This makes it necessary to restrict the search

Fig. 3. A Morphing sequence - the first and the last models are obtained from Google 3D Warehouse.



(a) Score = 0.486 (b) Score = 0.802

(c) Score = 0.683 (d) Score = 0.750

Fig. 4. Two real cups in our kitchen which better match one of the in-between models ($t = 0.25$) in Fig. 3, as compared to the original model ($t = 0$).

space. Since we estimate the final size of found objects we need a calibrated stereo setup. We also need an approximated position of the object, to calculate beforehand all relevant possible views. Simply calculating all views is not efficient and will probably use more than the available memory of any machine.

1) Detection: Any model we obtain from the internet usually already appears in a very appropriate position for finding it in the world. For most kitchen objects we found on the internet we see that they are somehow lying on supporting plane which is the xy-plane in the 3D model coordinates. This is of course an assumption and we do not rely completely on it, but we assume that the majority of the models are aligned in such a way. Since we align all models which we select as inliers to each other, we get a major upright direction. In our kitchen domain we assume that we have a box model of the major components of our world. And in case we are looking for an object we can expect an object present at one of several locations, like on the table, on the counter, in the fridge or in the cupboard. For all positions we can approximate a mean-position of the object relative to the robot inspecting this area. This will be the reference pose, of an object combined with the zero pose given by the object itself. Around this pose an uncertainty has to be specified that increases the search space such that all relative positions are in it. The expected distance of the object has to be adapted additionally, since the scaling of the model does not correspond to the real object. We increase the distance search space by 20 percent [20].

However, this shape matching approach requires significant preprocessing of the given 3D model, whose complexity

Fig. 5. Morphed and culled model matched in a cluttered scene

is polynomial increasing with the number of faces in the model. We utilize a very simple technique that is common in the 3D rendering community, called backface culling[16]. The complexity of this operation is linear with the number of faces, and reduces the number of faces to approximately half for the parameters we use (discussed shortly). The technique of backface culling simply requires calculating the angle between the normal of a face and the viewing vector (a vector from camera position to any point on the face). If this angle is greater than a threshold θ we do not include the face in our resulting model as it will not be visible from the camera anyway. This threshold can be estimated by using the range in which we have to search approximately. This is known, since we assume one plane to be the most probable to find the object standing on. So, we only have to consider a part of the viewing field specified by an anlge $\alpha_{lon} < 2\pi$ and $\alpha_{lat} < 2\pi$. A good prior for this values is given in [20]: the mean longitude and latitude search space for a dining table is about 34° in both directions. Thus, if we use a reasonable value for $\theta = 20°$ we can express such a world imposed view reductions with 2 culled models. An value of $\theta = 20$ can lead to a reduction of up to half of the faces of the original model (depending on its convexity). Thus, instead of giving one large pose and size range for the original model while matching, we break this range parts and pre-generate models which contain only those faces that will be visible in these smaller ranges. The number of parts can be by $n = \lceil \frac{\alpha_{lon}}{\theta} \rceil \lceil \frac{\alpha_{lat}}{\theta} \rceil$ Then, we feed the matching algorithm only these simplified models that are matched over the smaller ranges. The results in Table III indicate that for the same search space (in which the culled model is valid), the model generation time using the complete model takes more time by a factor of 2-3. Thus, even if we have a full pose range (complete sphere) still we may break it up into parts and obtain this much speedup.

2) Localization: During localization the position vector which is result of the detection has to be scaled correctly. This can be done either monocular, as soon as the size of the real object correspond to the size of the model or with a stereo setup. In a stereo setup we use the disparity that we can calculate using either two matches of the model in a left and a right image, or by finding the model once and calculating for the region of interest the disparity using template matching like normalized cross correlation. This method can only applied to well textured image regions. The distance z_1 of any point from the first camera of a stereo setup

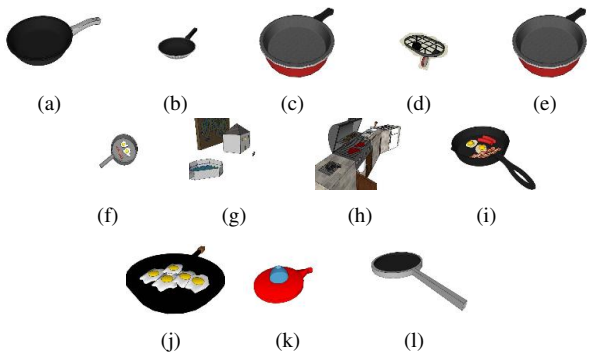


Fig. 6. The resulting four clusters for the query for "pan", a - e are inliers, all the others are classified as outliers (clusters are f-i, j-k and l).

with focal lengths f_1 and f_2 and relative distance of the two cameras after rectification d_{12} can be calculated using the disparity d_{pix} by $z_1 = \frac{d_{12} f_1}{d_{pix}}$. Using this information we can scale the pose which we got from the matching algorithm relative to the camera. The z component of the result of the algorithm z_{result} is the distance from the camera plane, which means that the scaling factor f_{scale} is $f_{scale} = \frac{Z_1}{z_{result}}$. This factor can be used to adapt the model and connect it with the real object that was found. That includes dividing all length measurements by this factor. Afterwards, the detection of this object can be done also monocular.

IV. Results

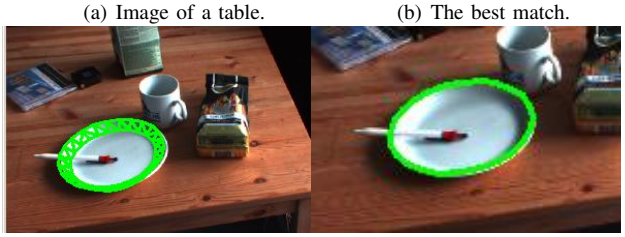
For the sake of completeness, we briefly review here some of the previous results we obtained in [1] and present in greater detail new ones.

A. Summary of Previous Results

Clustering and selection results on the models obtained for the search string "pan" is shown in Fig. 6. We manually counted the erroneous classification results for several queries, and it was seen that the quote of inliers is improved by selection in our test from 60% to 79%. We demonstrated and discussed how localization can be performed once the object is detected in the scene in detail; where we had partial knowledge of robot and the scene's location which allowed us to restrict the distance between camera and object to a certain range. We further used a stereo setup to derive the correct position of the detected mug. It was shown that our approach is feasible in an autonomous system by testing it in the following scenario: the robot gets the command to find a pot on the counter, in the cupboard or on the table. The current position of the robot is known and it can look at all three places by simple camera rotation. It was observed that the robot successfully localized the position of the pot.

The results of similar experiments searching plates can be found in Table II. Here we can see the potential of the method to deal with uncertainties: First, several models confirm the same correct object position, which allows detection of wrong matches. Also models are probably outlier themselves, if they do not support the most probable hypothesis. 7(a) shows the image of the table that was processed. Fig. 7(b) to 7(d) show several models matched at the same position.

(a) Image of a table. (b) The best match.



(c) The second match. (d) The third match.

Fig. 7. Results with plates.

Model(Faces)	Time Model	Time Search	Matches/Score	Remarks
Plate 0(188)	201 s	0.65s	1/0.98	Fig. 7(b)
Plate 10(22)	76 s	0.20s	1/0.97	Fig 7(d)
Plate 9(258)	296 s	1.89s	1/0.86	Fig 7(c)
Plate 4(332)	112 s	0.67s	1/0.83	mismatch
Plate 7(120)	223 s	1.09s	1/0.78	
Plate 3(118)	80 s	0.32s	1/0.76	
Plate 8(626)	326 s	1.24s	0/0.70	outlier, no match

TABLE II

Results for plates on a cluttered table.

B. Morphing

Since we want to detect all objects in our kitchen even those which are not described well with the direct search results, we want to show that certain objects can be fit much better with morphed models. Fig. 3 shows a morphing sequence ($t = 0, 0.25, 0.5, 0.75, 1.0$) between two cup models downloaded from Google 3D Warehouse. Fig. 4 shows that for real cups in our kitchen, the original model from the database is a poor match, whereas an in-between model ($t = 0.25$) is a good match. We reject matches with scores less than 0.7, thus these cups would not have been detected with the original model. Fig. 5 shows the same in-between culled model matched in a cluttered scene with high score (0.82). It should thus be understandable even with this simple example that it is possible to generate new models, when we do not find enough models for a particular search from the database. Unfortunately, it is difficult to quantitatively compare our approach with many others already existing in literature; because most of the existing ones require constant input from a human operator at different stages and do not work with models with open geometry (non genus-0 objects). Our technique is both automatic and works with models with open geometry which is usually the case with most objects present in a kitchen environment. On the other hand, our technique does not yield as visually attractive results as some others. However, our algorithm is extremely fast, giving reasonable morphing results for computer vision in tens of seconds for even very complex models.

C. Face reduction by Visibility culling

Since we have now more models, we have to speed up the shape model generation and the matching of them. Fig. 8 shows a cup and a pot that have been culled using the

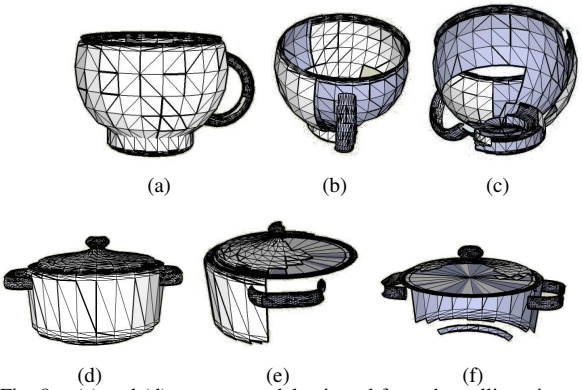


Fig. 8. (a) and (d) are two models viewed from the culling viewpoint, rest are the same models viewed from arbitrary viewpoints. Shaded faces are the invisible sides of visible faces.

Model	Original Model	Model after Culling
Mug	904 faces,113.7s	447 faces,29.5s
Cup	1892 faces,87.48s	844 faces,25.5s
Pot	344 faces,33.74s	179 faces,10.14s
Pot	1534 faces,60.393s	791 faces,38.924s
Cup Morphed	3552 faces,794.5s	1786 faces,188.2s

TABLE III

REDUCTION IN THE GENERATION TIME FOR SOME MODELS

technique of backface culling. The view from the culling viewpoint is valid (here) for rotations of 20 degrees in any direction. The original cup and pot models had 3164 and 6272 faces respectively, whereas the culled models have 1415 faces and 3135 faces. Table III shows the reduction in the model generation time when we apply backface culling and some extra reduction of far away faces (from the camera). The same pose and size ranges for both the original and the culled models were used. The speedup obtained is around 2 to 3 on the average. This indicates that significant speedups are possible, which will be particularly useful for the larger models. However, this is only possible if we have some prior knowledge of the pose. If that is not the case, the culling may still be useful for large models caused by the high memory usage of the generation process.

V. CONCLUSIONS

We present a system that enables a robot to automatically model the 3d shape of previously unseen objects. The system not only acquires already available models from an internet database, but also creates its new models when only few models are available online. This in comparable to the ability of humans of generalizing the shape of objects say cups or pots. We further applied the technique of visibility culling to speed-up the matching process, by making its most time-consuming stage faster. Our approach needs some time to calculate, so the search for a new object does not compete with human performance now. But given we can save the model generation phase or calculate it offline, the online matching would be fast enough for interaction with the world. We also see this only as a starting point for further specialization process that learns specific shape or visual descriptor models from the images with the already located objects that will allow realtime interaction.

REFERENCES

[1] U. Klank, M. Zia, and M. Beetz, "3d model selection from an internet database for robotic vision," *Proceedings of IEEE International Conference on Robotics and Automation 2009*, May 2009.

[2] M. Beetz, F. Stulp, B. Radig, J. Bandouch, N. Blodow, M. Dolha, A. Fedrizzi, D. Jain, U. Klank, I. Kresse, A. Maldonado, Z. Marton, L. Mösenlechner, F. Ruiz, R. B. Rusu, and M. Tenorth, "The assistive kitchen — a demonstration scenario for cognitive technical systems," in *IEEE 17th International Symposium on Robot and Human Interactive Communication (RO-MAN), Munich, Germany*, 2008, invited paper.

[3] A. Saxena, J. Driemeyer, and A. Ng, "Robotic Grasping of Novel Objects using Vision," *The International Journal of Robotics Research*, vol. 27, no. 2, p. 157, 2008.

[4] A. Miller, S. Knoop, H. Christensen, and P. Allen, "Automatic grasp planning using shape primitives," *IEEE International Conference on Robotics and Automation (ICRA), 2003*, vol. 2, pp. 1824–1829 vol.2, Sept. 2003.

[5] R. B. Rusu, A. Sundaresan, B. Morisset, M. Agrawal, and M. Beetz, "Leaving Flatland: Realtime 3D Stereo Semantic Reconstruction," in *Proceedings of the International Conference on Intelligent Robotics and Applications (ICIRA) 2008, October 15-17, Wuhan, China*, 2008.

[6] K. Okada, M. Kojima, S. Tokutsu, T. Maki, Y. Mori, and M. Inaba, "Multi-cue 3D object recognition in knowledge-based vision-guided humanoid robot system," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2007.*, pp. 3217–3222, 2007.

[7] A. Morales, T. Asfour, P. Azad, S. Knoop, and R. Dillmann, "Integrated grasp planning and visual object localization for a humanoid robot with five-fingered hands," in *International Conference on Intelligent Robots and Systems (IROS), 2006*, Beijing, China, 2006.

[8] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman, "Learning Object Categories from Googles Image Search," *Tenth IEEE International Conference on Computer Vision (ICCV), 2005.*, vol. 2, 2005.

[9] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, "Shape distributions," *ACM Transactions on Graphics (TOG)*, vol. 21, no. 4, pp. 807–832, 2002.

[10] E. Wahl, G. Hillenbrand, and G. Hirzinger, "Surflet-pair-relation histograms: a statistical 3D-shape representation for rapid classification," *Proceedings. Fourth International Conference on 3-D Digital Imaging and Modeling (3DIM), 2003.*, pp. 474–481, 2003.

[11] P. Besl and H. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, Feb 1992.

[12] F. Lazarus and A. Verroust, "Three-dimensional metamorphosis: a survey," *The Visual Computer*, vol. 14, no. 8, pp. 373–389, 1998.

[13] R. Hoover, A. Maciejewski, and R. Roberts, "Pose detection of 3-d objects using s2-correlated images and discrete spherical harmonic transforms," *IEEE International Conference on Robotics and Automation (ICRA), 2008*, pp. 993–998, May 2008.

[14] C. Wiedemann, M. Ulrich, and C. Steger, "Recognition and tracking of 3d objects," in *Pattern Recognition*, ser. Lecture Notes in Computer Science, G. Rigoll, Ed., vol. 5096. Berlin: Springer-Verlag, 2008, pp. 132–141.

[15] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3D Point Cloud Based Object Maps for Household Environments," *Robotics and Autonomous Systems Journal (Special Issue on Semantic Knowledge)*, 2008.

[16] D. Cohen-Or, Y. Chrysanthou, C. Silva, and F. Durand, "A survey of visibility for walkthrough applications," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 3, pp. 412–431, July 2003.

[17] E. Forgy, "Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications," 1965.

[18] C. Sun and J. Sherrah, "3d symmetry detection using the extended gaussian image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 164–169, February 1997.

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

[20] R. Tavcar, "Connecting High-Level Planning, Reasoning, and Model-driven Vision into a Robotic System that Enables Everyday Manipulation Tasks," Master's thesis, Intelligent Autonomous Systems Group, Technische Universitt Mnchen, Germany, March 2009.