Synthetic Learning Set for Object Pose Estimation: Initial Experiments

Joo-Haeng Lee[1,2], Woo-Han Yun[1], Jaeyeon Lee[1], Jaehong Kim[1]
[1]Human-Machine Interaction Group, ETRI, Daejeon, 34129, Korea
(Tel: +82-42-860-1338; E-mail: joohaeng@etri.re.kr)
[2]Computer Software Dept., University of Science and Technology, Daejeon, 34113, Korea

***Abstract*—We summarize a method to generate a synthetic learning set for object pose estimation in robotic manipulation tasks. Exploiting modern computer graphics techniques, our synthetic learning set satisfies the requirements both in quantitative diversity and qualitative precision. We report the partial results of initial experiments and discuss some future research directions.**

***Keywords*—Synthetic learning set, pose estimation, machine learning, robot manipulation.**

1. Introduction

Object pose estimation is a critical subtask for robotic manipulation. Recently, there is a strong tendency to apply a machine learning technique with a data-centric viewpoint. To handle real-world problems in pose estimation, however, the quantity and quality of a learning set seriously matter at once. In terms of quantity, for example, a myriad of different cases with viewing angles, lighting conditions, and occlusions are required as input images. In the qualitative side, each instance of training image should be accompanied by a set of precise and comprehensive annotation describing a specific case. It is not easy for a dataset based on collection of the Internet images with manual annotation to satisfy this requirement. In this paper, we illustrate a method to generate a synthetic learning set for object pose estimation based on computer graphics technique to satisfy both the quantitative and qualitative requirements, and report initial experimental results and discuss some future research directions.

2. Synthetic Learning Set

2.1. Target task and objects

The robot task considered in this study is basically similar to the Amazon Picking Challenge (APC) [1]: a robotic manipulator recognizes a target object in the shelf and pick it out to place in the goal bin or vice versa. In our study, we handle 23 objects from three sources: (1) 9 objects from APC 2015 dataset, which are relatively simple in shape and material, (2) 5 box-shaped objects from a Korean grocery store, which can be characterized with different textures, and (3) 9 free-form objects from BigBIRD datasets [2], which were scanned from high-resolution laser scanners [3]. See Fig 1.

(a) RGB images

(b) Position (c) Normal (d) Depth

Fig. 1: Samples of synthetic learning set: 9 objects from APC 2015 with red labels; 5 objects from a Korean grocery store with green labels; and 9 objects from BigBIRD with blue labels.

2.2. Modeling and scene setup

In this study, the overall process to generate a synthetic learning set largely depends on computer graphics technology. We constructed a synthesis pipeline based on *Blender*, an open source tool for contents creation [4]. See Fig. 2. Among key steps, this section describes modeling and scene setup.

For box-shaped objects, modeling is straightforward. We could easily measure the geometric information either based on rectangle constraints or manually, and the cube map texture can be composed with pictures captured from a smart-phone

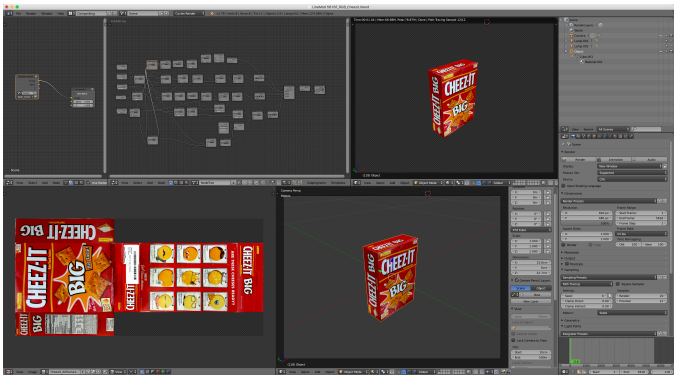


Fig. 2: An example of Blender usage.

camera. The lower left corner of Fig 2 shows an example of simple texture.

For objects with more general shapes, we used reconstructed models provided by BigBIRD, which were scanned using a quality scanner [3]. For example, a mesh with 512K faces is available with a fine texture. The corresponding physical objects are also available from Amazon store.

In the scene setup, the basic lighting is omni-directional white light, which can be replaced with HDRI environment maps for more realistic lighting. Each object is arranged so that its origin exists in the inertial center and placed at the world origin.

The camera can change its position and orientation according to control parameters. For example, 5,616 camera poses were synthesized with different parameters around each object: 24 pan angles, 13 tilt angles, 3 yaw and 6 depths. (There can be duplicates, but we do not consider them for now.) The principal axis of a camera always passes through the center of the object. The upper left corner of Fig. 2 shows an example of node tree to control camera motion around a target object to generate 5,616 frames. The number of frames roughly matches that of LINEMOD, which also utilizes synthetic images for pose estimation [5].

Generation of a synthetic learning set is different from the conventional computer graphics rendering in that various reference annotations should be appended to each image. In this study, annotations include position, normal and depth for each RGB pixel. (Masking information is included in the alpha channel.) Therefore, four types of images are generated at one camera viewpoint. See Fig. 1.

Each pixel of an RGB image contains color information of a corresponding object point under a specific illumination and viewing condition. Each type of annotation images contains non-RGB pixels representing position, normal and depth, respectively, of a corresponding object point. Annotation values are defined under an object-centric coordinate system.

2.3. Rendering

For image synthesis or rendering we resort to two methods. First, we implemented a simple ray tracer using *Mathematica* and generated various types of annotations for prototyping and testing additional algorithms. After a prototyping stage, the pipeline is transferred to Blender and its *Cycle* renderer. The Cycle renderer provides advanced rendering techniques such as path tracing, physical lighting, materials, camera models and composition to produce quality images similar to real photographs.

RGB images are generated using path tracing rendering. Currently, all the materials are assumed to be matte. Position and normal information are rendered using simple geometric shaders written in OSL (Open Shader Language). Depth images can be easily generated in a composition layer.

Render time varies depending on control parameters such as the size of the target image, input mesh complexity, lighting conditions and various sampling options for path tracing. In the current experiment, an image with 640×480 resolution is generated under a minimum sampling: i.e., 20 samples per pixel. In this setup, a typical render time per each camera pose is less than a second: *i.e.*, it takes about 0.42 sec to generate a set of RGB, position, normal and depth images for each object at a certain camera pose under 20 CPU cores. Although GPU rendering is supported in the Cycle renderer, IO overheads between main and GPU memories hinders to get any performance gain.

As a total, it takes about 15 hours to generate 516,672 frames for 5,616 camera poses, 23 objects and 4 types of images. Generated images might be further processed and organized for specific tasks (*i.e.*, pose estimation) as in the next section. Otherwise, they could be direct input to deep neural nets, which is our on-going work.

2.4. Dataset

Before directly feeding into deep neural nets, we applied our synthetic learning set to a more classical approach as follows. First, we extract feature points (*i.e.*, SURF) from synthetic RGB images. For each feature point, corresponding annotations such as position, normal and depth are indexed together. We apply appropriate filtering and spatial acceleration techniques to reduce data size and speed up search performance. After this process, each object has an independent index structure.

Among various feature descriptors, we are currently evaluating several methods to find a suitable one for pose estimation. See Fig 3. This experiments will be compared with end-to-end approach based on deep neural nets in the future.

3. Application in Pose Estimation

3.1. Experiment environments

A number of objects are placed on the shelf. The task environment is captured as an image using a camera, then we extract feature points from the image. A background modeling helps for better segmentation. Pose estimation is computed as in Section 3.2 based on feature points. See Fig. 4 for the overall process and experiment environment.

To provide better interaction modality, a projector-camera system is installed as well. See Fig. 4b The projector visually displays the recognition and estimation results directly on

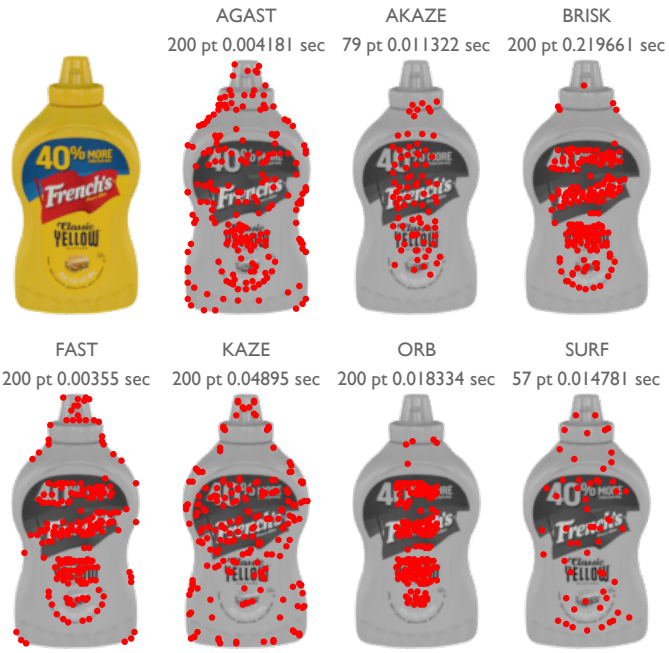
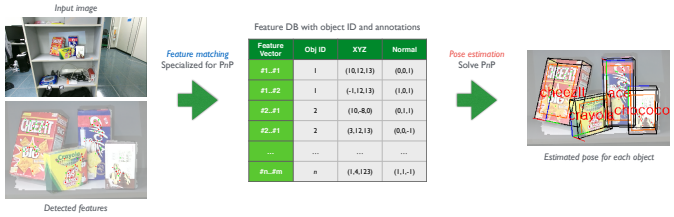
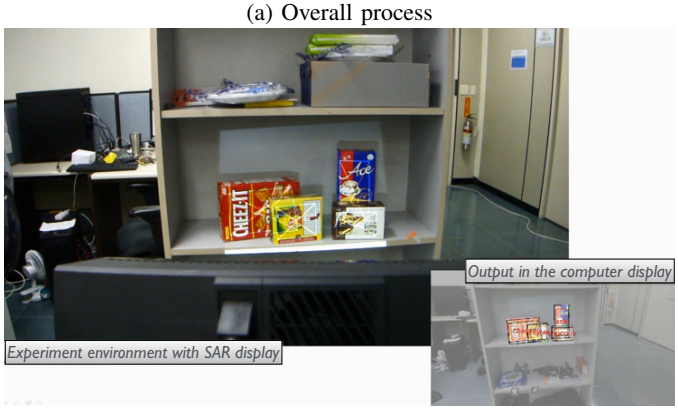


Fig. 3: Feature extraction methods from a synthetic image: AGAST, AKAZE, BRISK, FAST, KAZE, ORG, and SURF.



(a) Overall process



(b) Experiment environment with SAR display

Fig. 4: Experiment process and environment.

the object surface using a spatial augmented reality (SAR) technique.

3.2. *Pose estimation method*

The actual pose estimation is computed using PnP (Perspective n Point) method. See Fig. 4a. To apply this scheme, we utilize the index structure of Section 2.4 derived from the synthetic learning set. First, feature points are extracted from the real input images. Then, we search object-wise index structures for most similar features and corresponding annotations.

Finally, we apply PnP for the key points found in real images and corresponding positions of each object in the synthetic dataset. The object with the largest number of inliers in the PnP is chosen as the detected object with its estimated pose. Normal and depth images also help to constrain the search space.

3.3. *Experimental results*

According to our experiments based on the proposed method, pose estimation can be computed within a few second for a couple of objects, which shows a linear time performance to the number of objects in the task environment. The accuracy is visually plausible when the results are imposed over the physical objects using a projector.

Through our initial experiments, we could confirm solid possibility of the proposed method based on synthetic learning sets as well as several points for improvements. In particular, it is quite interesting that we did not use any real images to build an index structure in the actual experiment, which was a tremendous labor saver.

As an immediate further work, we are working on to feed the synthetic images directly into deep neural nets for detection, recognition and pose estimation without specifying explicit features. For precise estimation of pose, however, we still believe that an analytic solution such as PnP should be augmented to form a hybrid solution.

4. Summary

We outlined our on-going project on object pose estimation for robotic manipulator. Especially, our pose estimation heavily utilizes a synthetic learning set generated using computer graphics techniques. Initial experiments show the promising contribution of a synthetic learning set as well as future research directions. Our synthetic learning set will be shared in public if ready [6].

Acknowledgment

This research was supported by the MOTIE under the Robot industry fusion core technology development project (10048920) supervised by the KEIT.

References

[1] Correll, N., et al., 2016. “Lessons from the amazon picking challenge,” arXiv preprint arXiv:1601.05484.
[2] Calli, B., et al., 2015. “Benchmarking in manipulation research: The YCB object and model set and benchmarking protocols,” arXiv preprint arXiv:1502.03143.
[3] Singh, A., 2016. “Benchmark for Cloud Robotics,” Technical Report No. UCB/EECS-2016-142.
[4] Blender. https://www.blender.org
[5] Hinterstoisser, S., et al., 2012, November. “Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes,” In *ACCV 2012* (pp. 548-562).
[6] Synthetic Learning Set for Machine Learning, https://github.com/joohaeng/SLS4ML.