A Self-supervised Learning System for Object Detection using Physics Simulation and Multi-view Pose Estimation

Chaitanya Mitash, Kostas E. Bekris and Abdeslam Boularias

***Abstract*— Progress has been achieved recently in object detection given advancements in deep learning. Nevertheless, such tools typically require a large amount of training data and significant manual effort to label objects. This limits their applicability in robotics, where solutions must scale to a large number of objects and variety of conditions. This work proposes an autonomous process for training a Convolutional Neural Network (CNN) for object detection and pose estimation in robotic setups. The focus is on detecting objects placed in cluttered, tight environments, such as a shelf with multiple objects. In particular, given access to 3D object models, several aspects of the environment are physically simulated. The models are placed in physically realistic poses with respect to their environment to generate a labeled synthetic dataset. To further improve object detection, the network self-trains over real images that are labeled using a robust multi-view pose estimation process. The proposed training process is evaluated on several existing datasets and on a dataset collected for this paper with a Motoman robotic arm. Results show that the proposed approach outperforms popular training processes relying on synthetic - but not physically realistic - data and manual annotation. The key contributions are the incorporation of physical reasoning in the synthetic data generation process and the automation of the annotation process over real images.**

I. Introduction

Object detection and pose estimation is frequently the first step of robotic manipulation. Recently, deep learning methods, such as those employing Convolutional Neural Networks (CNNs), have become the standard tool for object detection, outperforming alternatives in object recognition benchmarks. These desirable results are typically obtained by training CNNs using datasets that involve a very large number of labeled images, as in the case of ImageNet [1]). Creating such large datasets requires intensive human labor. Furthermore, as these datasets are general-purpose, one needs to create new datasets for specific object categories and environmental setups that may be of importance to robotics, such as warehouse management and logistics.

The recent Amazon Picking Challenge (APC) [2] has reinforced this realization and has led into the development of datasets specifically for the detection of objects inside shelving units [3], [4], [5]. These datasets are created either with human annotation or by incrementally placing one object in the scene and using foreground masking.

The authors are with the Computer Science Department of Rutgers University in Piscataway, New Jersey, 08854, USA. Email: {cm1074,kb572,ab1544}@rutgers.edu.

This work is supported by NSF awards IIS-1734492, IIS-1723869 and IIS-1451737. Any opinions or findings expressed in this paper do not necessarily reflect the views of the sponsors. The authors would like to thank the anonymous IROS'17 reviewers for their constructive comments.

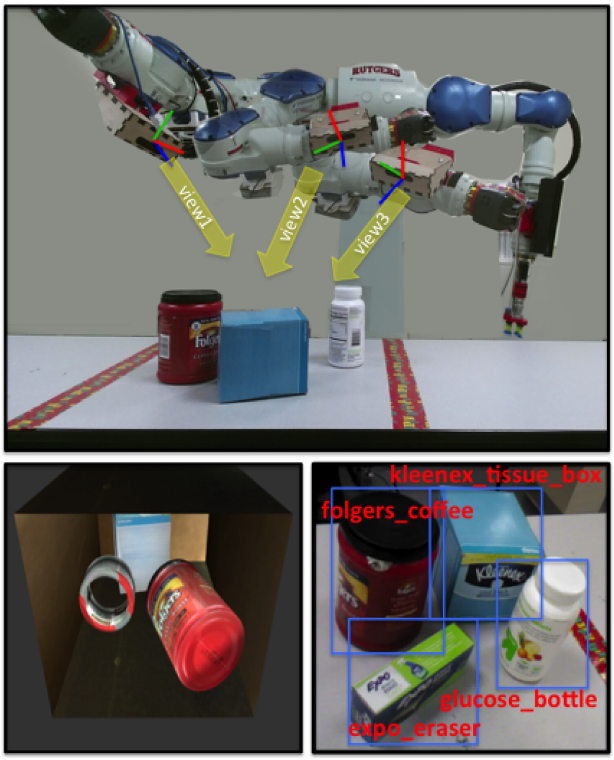

Fig. 1. (top) A robotic arm performs pose estimation from multiple view points using an object detector trained in physically-simulated scenes (bottom-left). The estimated poses are used to automatically label real images (bottom-right). They are added to the training dataset as part of a lifelong learning process. Initially, the multi-view pose estimation procedure bootstraps its accuracy by trusting the objects' labels as predicted from the detector given training over synthetic images. It then uses these labels to annotate images of the same scene taken from more complex viewpoints.

An increasingly popular approach to avoid manual labeling is to use synthetic datasets generated by rendering 3D CAD models of objects with different viewpoints. Synthetic datasets have been used to train CNNs for object detection [6] and viewpoint estimation [7]. One major challenge in using synthetic data is the inherent difference between virtual training examples and real testing data. For this reason, there is considerable interest in studying the impact of texture, lighting, and shape to address this disparity [8]. Another issue with synthetic images generated from rendering engines is that they display objects in poses that are not necessarily physically realistic. Moreover, occlusions are usually treated in a rather naive manner, i.e., by applying cropping, or pasting rectangular patches, which again results in unrealistic scenes [6], [7], [9].

This work proposes an automated system for generating and labeling datasets for training CNNs. The objective of the

proposed system is to reduce manual effort in generating data and to increase the accuracy of bounding-box-based object detection for robotic setups. In particular, the two main contributions of this work are:

- A physics-based simulation tool, which uses information from camera calibration, object models, shelf or table localization to setup an environment for generating training data. The tool performs physics simulation to place objects at realistic configurations and renders images of scenes to generate a synthetic dataset to train an object detector.
- A lifelong, self-learning process, which employs the object detector trained with the above physics-based simulation tool to perform a robust multi-view pose estimation with a robotic manipulator, and use the results to correctly label real images in all the different views. The key insight behind this system is the fact that the robot can often find a good viewing angle that allows the detector to accurately label the object and estimate its pose. The object's predicted pose is then used to label images of the same scene taken from more difficult views, as shown in Fig. 1. The transformations between different views are known because they are obtained by moving the robotic manipulator.

The software and data of the proposed system, in addition to all the experiments, are publicly available at `http://www.physimpose.com`

II. Related Work

The novelty of the proposed system lies on the training process for generating synthetic data as well as augmenting the synthetic data with real ones that are generated from an automated, self-learning process. This involves several modules, which have been studied in the related literature over the years.

Object Segmentation: The tasks of object detection and semantic segmentation of images have been studied extensively and evaluated on large scale image datasets. Recently, the `RCNN` approach combined region proposals with convolutional neural networks [11]. This opened the path to high accuracy object detection, which was followed up by deep network architectures [12], [13] and end-to-end training frameworks [14], [10]. There has also been a significant success in semantic labeling of images with the advent of Fully Convolutional networks (`FCN`) [15] and its extensions [16], [17], [18]. This work utilizes `FCN` and `Faster-RCNN` and proposes an automated way to collect data and incrementally train the structures for improved performance.

Pose Estimation: One way to approach this challenge is through matching local features, such as SIFT [19], or by extracting templates using color gradient and surface normals from 3D object models [20]. Synthesis-based approaches have also been gaining popularity [21], [22]. Nevertheless, in application domains, such as those studied by the Amazon Picking Challenge [2], which involve varying light conditions and cluttered scenes, it has been shown [5] that `CNN`-based segmentation [10], [15] followed by point cloud registration with 3D models [23], [24], [25] is an effective approach. This paper builds on top of these techniques for pose estimation and proposes a method to self-feed the output of such processes to improve accuracy.

Synthetic Datasets: Synthetic datasets generated from 3D models have been used for object detection [6], [26] and pose estimation [27], [7] with mixed success as indicated by an evaluation of the performance of detectors trained on synthetic images to those trained with natural images [9]. This work proposes the incorporation of a physics-based simulator to generate realistic images of scenes, which helps object detection success rate.

Self-supervised Learning: The idea of incrementally learning with minimal supervision has been exploited previously in many different ways. Curriculum learning [28] and self-paced learning [29] have been adapted to improve the performance of object detectors [30], [31]. The self-learning technique proposed here involves the robot acquiring real images of scenes from multiple views. Then the robot uses the knowledge acquired from confidently detected views and 3D model registration to improve object detection in a lifelong manner.

III. Physics-aware Synthetic Data Generation

The proposed system starts by physically simulating a scene as well as simulating the parameters of a known camera. The accompanying tool generates a synthetic dataset for training an object detector, given 3D CAD models of objects. This module has been implemented using the Blender API [32], which internally uses the Bullet physics engine [33]. The pipeline for this process is depicted in Fig. 2, while the corresponding pseudocode is provided in Alg. 1. The method receives as input:

- a set of predefined camera poses $\mathbf{P}_{cam}$,
- the pose of the resting surface $\mathbf{P}_s$,
- the intrinsic parameters of the camera $\mathbf{C}_{int}$,
- the set of 3D object models $\mathbf{M}$ and
- the number of simulated training images N to generate.

Algorithm 1: PHYSIM_CNN($\mathbf{P}_{cam}, \mathbf{P}_s, \mathbf{C}_{int}, \mathbf{M}, N$)

dataset $\leftarrow \emptyset$;
while $(|\mathbf{dataset}| < N)$ **do**
  $\mathbf{O} \leftarrow$ a random subset of objects from $\mathbf{M}$;
  $\mathbf{P}^O_{init} \leftarrow$ INITIAL_RANDOM_POSES($\mathbf{O}$);
  $\mathbf{P}^O_{final} \leftarrow$ PHYS_SIM($\mathbf{P}^O_{init}$, $\mathbf{P}_s$, $\mathbf{O}$);
  Light $\leftarrow$ PICK_LIGHTING_CONDITIONS();
  foreach $(\mathbf{view} \in \mathbf{P}_{cam})$ **do**
    image $\leftarrow$ RENDER($\mathbf{P}^O_{final}$, **view**, C_{int}, **Light**);
    { **labels, bboxs** } $\leftarrow$ PROJECT($\mathbf{P}^O_{final}$, **view**);
    dataset $\leftarrow$ **dataset** $\cup$ (**image, labels, bboxs**);
SIM_DETECT($\cdot$) $\leftarrow$ FASTER-RCNN(**dataset**);
return SIM_DETECT($\cdot$) AND **dataset**;

In a sensing system for robotic manipulation, a 6 degree-of-freedom (DoF) pose of the camera mounted on a robotic arm $(\mathbf{view} \in \mathbf{P}_{cam})$, can be exactly computed using forward

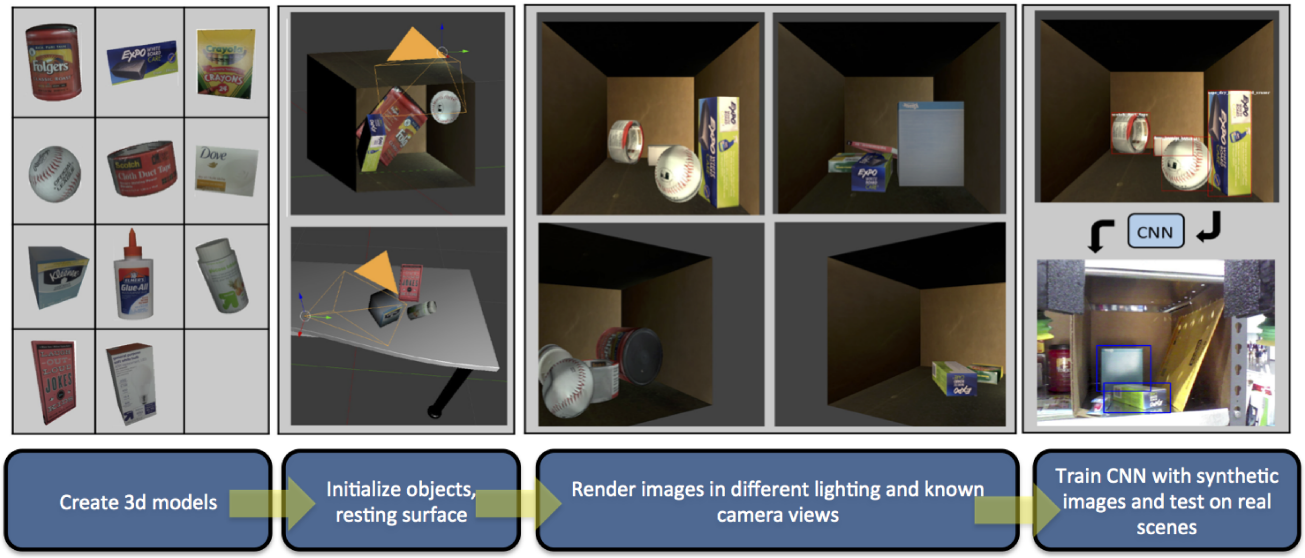

Fig. 2. Pipeline for physics aware simulation: The 3D CAD models are generated and loaded in a calibrated environment on the simulator. A subset of the objects is chosen for generating a scene. Objects are physically simulated until they settle on the resting surface under the effect of gravity. The scenes are rendered from known camera poses. Perspective projection is used to compute 2D bounding boxes for each object. The labeled scenes are used to train a Faster-RCNN object detector [10], which is tested on a real-world setup.

kinematics. Furthermore, the camera calibration provides the intrinsic parameters of the camera $\mathbf{C}_{int}$. To position the resting surface for the objects, a localization process is performed first in the real-world to compute the pose of the resting surface $\mathbf{P}_s$. The system has been evaluated on an APC shelf and a table-top environment. The shelf localization process uses RANSAC [34] to compute edges and planes on the shelf and the geometry of the shelf model is used to localize the bins. Given the above information as well as 3D object models $\mathbf{M}$, the method aims to render and automatically label N different images in simulation.

The algorithm simulates a scene by first choosing the objects $\mathbf{O}$ from the list of available object models $\mathbf{M}$ (line 3). The initial pose of an object is provided by function INITIAL_RANDOM_POSES (line 4), which samples uniformly at random along the x and y axis from the range $(\frac{-dim_i}{2}, \frac{dim_i}{2})$, where dim_i is the dimension of the resting surface along the i^{th} axis. The initial position along the z-axis is fixed and can be adjusted to either simulate dropping or placing. The initial orientation is sampled appropriately in SO(3). Then the function PHYS_SIM is called (line 5), which physically simulates the objects and allows them to fall due to gravity, bounce, and collide with each other as well as with the resting surface. Any inter-penetrations among objects or with the surface are treated by the physics engine. The final poses of the objects $\mathbf{P}^O_{final}$, when they stabilize, resemble real-world poses. Gravity, friction coefficients and mass parameters are set at similar values globally and damping parameters are set to the maximum to promote fast stabilization.

The environment lighting and point light sources are varied with respect to location, intensity and color for each rendering (line 6). Simulating different indoor lighting sources according to related work [35] helps to avoid over-fitting to a specific texture. This makes the training set more robust to different testing scenarios. Once lighting conditions **Light** are chosen, the simulated scene is rendered from multiple views using the pre-defined camera poses (line 6). The rendering function RENDER requires the set of stabilized object poses $\mathbf{P}^O_{final}$, the camera viewpoint **view** as well as the selected lighting conditions **Light** and intrinsic camera parameters $\mathbf{C}_{int}$ (line 7). Finally, perspective projection is applied to obtain 2D bounding box labels for each object in the scene with function PROJECT (line 8). The overlapping portion of the bounding boxes for the object that is further away from the camera is pruned.

The synthetic **dataset** generated is used to train an object detector SIM_DETECT($\cdot$) based on Faster-RCNN [10], which utilizes a deep VGG network architecture [13].

IV. Self-Learning via Multi-view Pose Estimation

Given access to an object detector trained with the physics-based simulator, the self-learning pipeline labels real-world images using a robust multi-view pose estimation. This is based on the idea that the detector performs well on some views, while might be imprecise or fail in other views. Aggregating 3D data over the confident detections and with access to the knowledge of the environment, a 3D segment can be extracted for each object instance in the scene. This process combined with the fact that 3D models of objects are available, makes it highly likely to estimate correct 6DoF pose of objects given enough views and search time. The results of pose estimation are then projected back to the multiple views, and used to label real images. These examples are very effective to reduce the confusion in the classifier for novel views. The process also autonomously reconfigures the scene using manipulation actions to apply the labeling process iteratively over time on different scenes, thus generating a labeled dataset which is used to re-train the object detector. The pipeline of the process is presented in Fig.3 and the pseudocode is provided in Alg. 2.

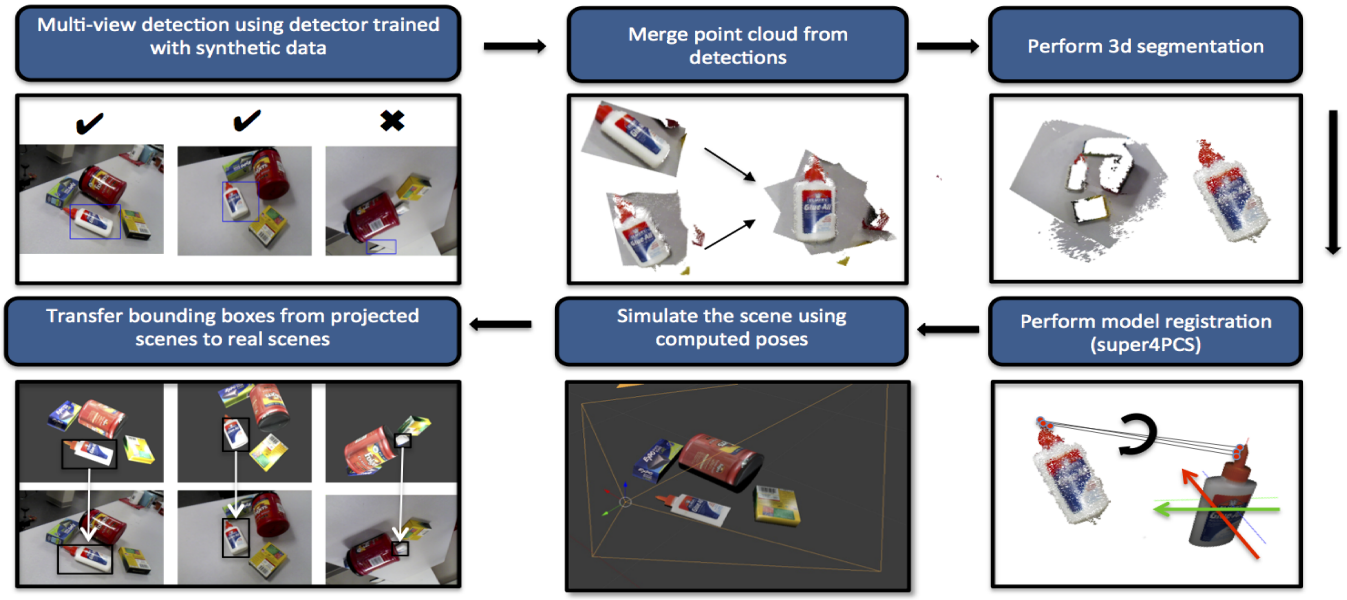

Fig. 3. Automatic self-labeling pipeline: The detector, which is trained with simulated data, is used to detect objects from multiple views. The point cloud aggregated from successful detections undergoes 3D segmentation. Then, Super4PCS [24] is used to estimate the 6D pose of the object in the world frame. The computed poses with high confidence are simulated and projected back to the multiple views to obtain precise labels over real images.

A robotic arm is used to move the sensor to different pre-defined camera configurations $\mathbf{P}_{cam}$ and capture color (**RGB**) and depth (**D**) images of the scene (lines 2-3). The PRACSYS motion planning library [36], [37] was used to control the robot in the accompanying implementation.

Algorithm 2: SELF-LEARN($\mathbf{dataset}, \mathbf{P}_{cam}, \mathbf{M}, N'$)

1 **while** $|dataset| < N'$ **do**
2 **foreach** **view** $\in \mathbf{P}_{cam}$ **do**
3 $\{\mathbf{RGB}_{view}, \mathbf{D}_{view}\} \leftarrow$ CAPTURE(**view**);
4 **foreach** *object* **o** *in the scene and* **M** **do**
5 **Cloud[o]** $= \emptyset$;
6 **foreach** **view** $\in \mathbf{P}_{cam}$ **do**
7 **bbox** $\leftarrow$ SIM_DETECT(RGB_{view});
8 **if** CONFIDENCE(**bbox**) $>$ *threshold* **then**
9 **3DPts** $\leftarrow$ GET_3DPTS(**bbox**, $\mathbf{D}_{view}$);
10 **Cloud[o]** $\leftarrow$ **Cloud[o]** $\cup$ **3DPts**;
11 OUTLIER_REMOVAL(**Cloud[o]**);
12 **P[o]** $\leftarrow$ SUPER4PCS(**Cloud[o]**, **M[o]**);
13 **foreach** **view** $\in \mathbf{P}_{cam}$ **do**
14 { **labels, bboxs** } $\leftarrow$ PROJECT($\mathbf{P}[o]$, **view**);
15 **dataset** $\leftarrow$ **dataset** $\cup$ ($\mathbf{RGB}_{view}$, **labels**, **bboxs**);
16 **randObj** $\leftarrow$ SAMPLE_RANDOM_OBJECT(**M**);
17 **objConfig** $\leftarrow$ PICK_CONFIG();
18 RECONFIGURE_OBJECT(**randObj**, **objConfig**);
19 NEW_DETECT($\cdot$) $\leftarrow$ FASTER-RCNN(**dataset**);
20 **return** NEW_DETECT($\cdot$);

The detector trained using physics-aware simulation is then used to extract bounding boxes **bbox** corresponding to each object o in the scene (line 7). There might exist a bias in simulation either with respect to texture or poses, which can lead to imprecise bounding boxes or complete failure in certain views. For the detection to be considered for further processing, a threshold is considered on the confidence value returned by RCNN (line 8).

The pixel-wise depth information **3DPts** within the confidently detected bounding boxes **bbox** (line 9) is aggregated in a common point cloud per object **Cloud[o]** given information from multiple views (line 10). The process employs environmental knowledge to clean the aggregated point cloud (line 11). points outside the resting surface bounds are removed and outlier removal is performed based on k-nearest neighbors and a uniform grid filter.

Several point cloud registration methods were tested for registering the 3D model **M[o]** with the corresponding segmented point cloud **Cloud[o]** (line 12). This included SUPER4PCS [24], fast global registration [25] and simply using the principal component analysis (PCA) with Iterative Closest Point (ICP) [23]. The SUPER4PCS algorithm [24] used alongside ICP was found to be the most applicable for the target setup as it is the most robust to outliers and returns a very natural metric for confidence evaluation. SUPER4PCS returns the best rigid alignment according to the Largest Common Pointset (LCP). The algorithm searches for the best score, using transformations obtained from four-point congruences. Thus, given enough time, it generates the optimal alignment with respect to the extracted segment.

After the 6DoF pose is computed for each object, the scene is recreated in the simulator using object models placed at the pose **P[o]** and projected to the known camera views (line 14). Bounding boxes are computed on the simulated setup and transferred to the real images. This gives precise bounding box labels for real images in all the views (line 15).

To further reduce manual labeling effort, an autonomous scene reconfiguration is performed (lines 16-18). The robot reconfigures the scene with a pick and place manipulation action to iteratively construct scenes and label them, as in Fig. 4. For each reconfiguration, the object to be moved is chosen randomly and the final configuration is selected from a set of pre-defined configurations in the workspace.

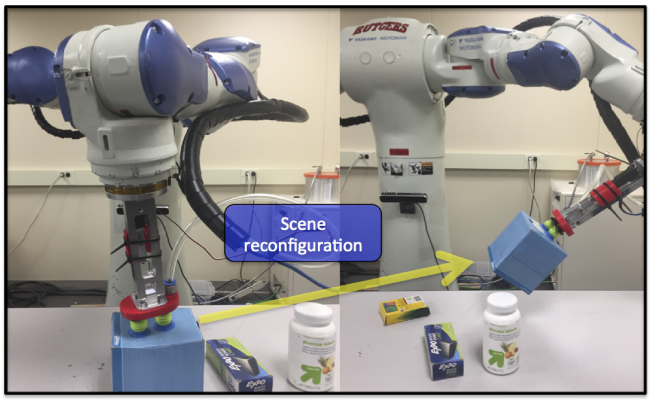

Fig. 4. Manipulator performing scene reconfiguration by moving an object from one configuration on the table to another

V. Evaluation

This section discusses the datasets considered, it compares different techniques for generating synthetic data and evaluates the effect of self-learning. It finally applies the trained detector on the 6DoF pose estimation task. The standard Intersection-Over-Union (`IoU`) metric is employed to evaluate performance in the object detection task.

A. Datasets

Several RGB-D datasets have been released in the setting of the Amazon Picking Challenge [3], [4], [5]. They proposed system was evaluated on the benchmark dataset released by Team MIT-Princeton called the Shelf&Tote dataset [5]. The experiments are performed on 148 scenes in the shelf environment with different lighting and clutter conditions. The scenes include 11 objects used in APC with 2220 images and 229 unique object poses. The objects were chosen to represent different geometric shapes but ignoring the ones which did not have enough depth information. Thus, the results can be generalized to a large set of objects.

The proposed system has been also evaluated on a real-world table-top setup. The corresponding test dataset was generated by placing multiple objects in different configurations on a table-top. An `Intel RealSense` camera mounted on a `Motoman` robot was used to capture images of scenes from multiple views. Images corresponding to 41 cluttered scenes, with 11 `APC` objects and 473 detection instances were collected and manually labeled.

B. Evaluating the Object Detector trained in Simulation

To study how object pose distribution effects the training process, different techniques for synthetic data generation are evaluated. The results of experiments performed on the Shelf&Tote dataset are presented in Table I.

1) Generating training data using test data distribution: The objective here is to establish an upper bound for the performance of a detector trained with simulated images. For this purpose, the object detector is trained with the knowledge of pose distribution in the test data. This process consists of estimating the density of the test data with respect to object poses using *Kernel Density Estimation*, and generating training data according to this distribution, as follows:

- Uniformly simulate many scenes using a simulator and record the poses for each object in the scene.
- Weigh each generated scene according to its similarity to test data. This is the sum of the number of objects in the scene for which the pose matches (rotation difference less than 15^o and translation difference less than 5cm) at least one pose in their corresponding test pose distribution.
- Normalize the weights to get a probability distribution on the sampled poses.
- Sub-sample the training poses using the normalized probability distribution

The sampled scenes were used to train a Faster-RCNN detector, which achieved an accuracy of 69%.

2) Uniformly sampled synthetic data: This alternative is a popular technique of generating synthetic data. It uses 3D models of the objects to render their images from several viewpoints sampled on a spherical surface centered at the object. The background image corresponded to the `APC` shelf, on top of which randomly selected objects were pasted at sampled locations. This process allows to simulate occlusions and mask subtraction provides the accurate bounding boxes in these cases. The objects in these images are not guaranteed to have physically realistic poses. This method of synthetic data generation does not perform well on the target task, giving a low accuracy of 31%.

3) Generating training data with physics-aware simulation: The accuracy of 64% achieved by the proposed physics-aware simulator is close to the upper bound. By incorporating the knowledge of the camera pose, resting surface and by using physics simulation, the detector is essentially over-fitted to the distribution of poses from which the test data comes, which can be useful for robotic setups.

	Method	Success(IoU>0.5
	Team MIT-Princeton [5] (Benchmark)	75%
Simulation	Sampled from test data distribution	69%
	Sampled from uniform distribution	31%
	Physics-aware simulation	64%
	Physics-aware simulation + varying light	**70%**
Self-Learning	Self-learning (2K images)	75%
	Self-learning (6K images)	81%
	Self-learning (10K images)	**82%**

TABLE I
Evaluation on Princeton's Shelf&Tote dataset [5]

The results discussed until now were with respect to a constant lighting condition. As the dataset grows, then a dip in the performance is observed. This is expected as the detector overfits with respect to the synthetic texture, which does not mimic real lighting condition. This is not desirable, however. To deal with this issue, the lighting conditions are varied according to the location and color of the light source.

This does resolve the problem to some extent but the dataset bias still limits performance to an accuracy of 70%.

On the table-top setup, the detector trained by the physics-based simulation has a success rate of 78.8%, as shown in Table II.

Method	Success(IoU>0.5)
Physics aware simulation	78.8%
Self-learning (140 images)	**90.3%**

TABLE II

DETECTION ACCURACY ON TABLE-TOP EXPERIMENTS

C. Evaluating Self-learning

The self-learning pipeline is executed over the training images in the `Shelf&Tote` [5] training dataset to automatically label them using multi-view pose estimation. The real images are incrementally added to the simulated dataset to re-train the `Faster-RCNN`. This results in a performance boost of 12%. This result also outperforms the training process by [5] which uses approximately 15,000 real images labeled using background subtraction. The reason that the proposed method outperforms a large dataset of real training images is mostly because the proposed system can label objects placed in a clutter.

On the table-top setup, pose estimation is performed using the trained detector and model registration. The estimated poses with high confidence values are then projected to the known camera views to obtain the 2D bounding box labels on real scenes. This is followed by reconfiguring the scenes using pick and place manipulation. After generating 140 scenes with a clutter of 4 objects in each image, the automatically labeled instances are used to retrain the `Faster-RCNN` detector. The performance improvement by adding these labeled examples is presented in Table II. The overall performance improvement is depicted in Fig. 5, while an example is shown in Fig.6.

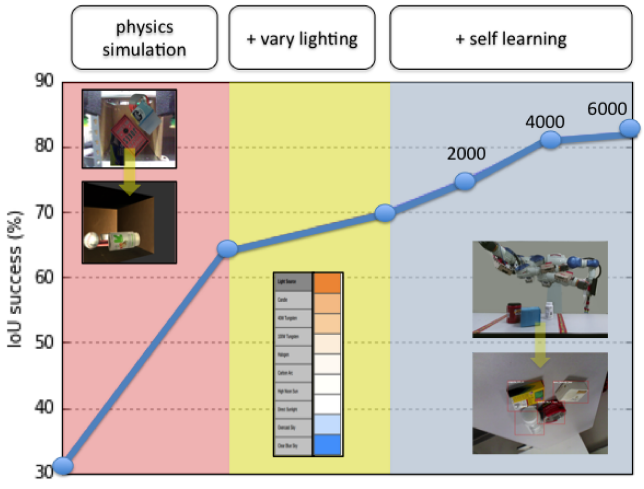

Fig. 5. Plot depicting the performance improvement by adding different components of the system

D. Evaluating the detector for 6DoF Pose estimation

Success in pose estimation is evaluated as the percentage of predictions with an error in translation less than 5cm and mean error in the rotation less than 15^o. The results of pose estimation are compared to the pose system proposed by the APC Team MIT-Princeton [5] in addition to different model registration techniques. The results are depicted in Table III. Given the above specified metric, the proposed approach outperforms the pose estimation system proposed before [5] by a margin of 25%. It is very interesting to note that the success in pose estimation task is at par with the success achieved using ground truth bounding boxes.

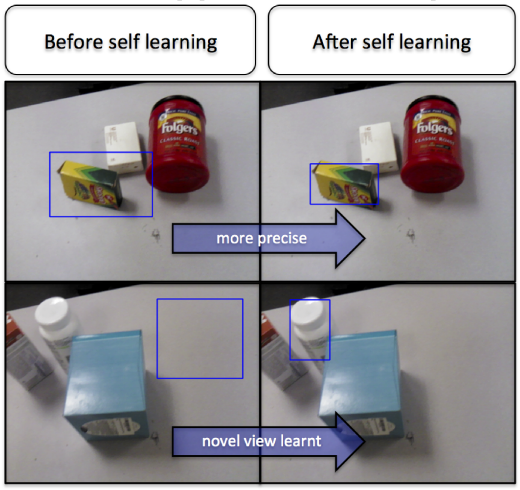

Fig. 6. Results of object detection before and after training with the self-learning process. The detector learns to predict more precise bounding boxes. It can also detect objects better from novel views.

VI. DISCUSSION

This work provides a system that autonomously generates data to train `CNNs` for object detection and pose estimation in robotic setups. Object detection and pose estimation are tasks that are frequently required before grasping [38] or rearranging objects with a robot [39], [40]. A key feature of the proposed system is physical reasoning. In particular, it employs a physics engine to generate synthetic but physically realistic images. The images are very similar to real-world scenes in terms of object pose distributions. This helps to increase the success ratio of `CNNs` trained with simulated data and reduces the requirements for manual labeling.

Nevertheless, synthetic data may not be sufficient as they cannot always generalize to the lighting conditions present in the real-world. For this purpose and given access to a robotic setup, this work proposes a lifelong learning approaches for a manipulator to collect additional labeled data in an autonomous manner. In particular, the method utilizes successful, high confidence detections from multiple views to perform pose estimation. This avoids over-fitting to simulated conditions. The overall combination of physical reasoning and self-learning results in a success ratio that outperforms current state-of-the-art systems in robot vision.

A future objective remains to achieve similar quality of object detection and pose estimation only with simulated data. This would minimize the dependency of having access to a robotic setup for adaptation of the learning procedure to real-world conditions. Furthermore, it would be interesting, to extend the training process to facilitate semantic segmentation of scenes, which could lead to an even more robust pose estimation.

2D-Segmentation Method	3D-registration Method	Mean-error Rotation (deg)	Mean-error Translation (m)	Success(%)
Ground-Truth Bounding-Box	PCA + ICP	7.65	0.02	84.8
FCN (trained with [5])	PCA + ICP	17.3	0.06	54.6
FCN (trained with [5])	Super4PCS + ICP	16.8	0.06	54.2
FCN (trained with [5])	fast-global-registration	18.9	0.07	43.7
RCNN (Proposed training)	PCA + ICP	**8.50**	0.03	**79.4**
RCNN (Proposed training)	Super4PCS + ICP	8.89	**0.02**	75.0
RCNN (Proposed training)	fast-global-registration	14.4	0.03	58.9

TABLE III

COMPARING THE PERFORMANCE OF THE PROPOSED SYSTEM TO STATE-OF-THE-ART TECHNIQUES FOR POSE ESTIMATION.

REFERENCES

[1] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *Intern. J. of Computer Vision (IJCV)*, vol. 115, pp. 211–252, 2015.

[2] N. Correll, K. E. Bekris, D. Berenson, O. Brock, A. Causo, K. Hauser, K. Osada, A. Rodriguez, J. Romano, and P. Wurman, "Analysis and Observations From the First Amazon Picking Challenge," *IEEE Trans. on Automation Science and Engineering (T-ASE)*, 2016.

[3] A. Singh, J. Sha, K. S. Narayan, T. Achim, and P. Abbeel, "Bigbird: A large-scale 3d database of object instances," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014.

[4] C. Rennie, R. Shome, K. E. Bekris, and A. F. De Souza, "A dataset for improved rgbd-based object detection and pose estimation for warehouse pick-and-place," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 1179 – 1185, 2016.

[5] A. Zeng, K.-T. Yu, S. Song, D. Suo, E. Walker Jr, A. Rodriguez, and J. Xiao, "Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[6] X. Peng, B. Sun, K. Ali, and K. Saenko, "Learning deep object detectors from 3D models," in *IEEE Intern. Conf. on Computer Vision*, 2015.

[7] H. Su, C. R. Qi, Y. Li, and L. J. Guibas, "Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3d model views," in *IEEE Intern. Conf. on Computer Vision*, 2015.

[8] B. Sun and K. Saenko, "From virtual to reality: Fast adaptation of virtual object detectors to real domains," in *British Machine Vision Conf.*, 2014.

[9] Y. Movshovitz-Attias, T. Kanade, and Y. Sheikh, "How useful is photo-realistic rendering for visual learning?" in *ECCV 2016 Workshops*, 2016.

[10] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91–99.

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2014.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2016.

[13] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[14] R. Girshick, "Fast R-CNN," in *IEEE Intern. Conf. on Computer Vision*, 2015, pp. 1440–1448.

[15] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[16] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014.

[17] J. Dai, K. He, and J. Sun, "Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation," in *IEEE Int. Conf. on Computer Vision*, 2015, pp. 1635–1643.

[18] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei, "Fully convolutional instance-aware semantic segmentation," *arXiv preprint arXiv:1611.07709*, 2016.

[19] K. Pauwels and D. Kragic, "Simtrack: A simulation-based framework for scalable real-time object pose detection and tracking," in *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2015.

[20] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," in *Asian Conference on Computer Vision*, 2012.

[21] A. Krull, E. Brachmann, F. Michel, M. Ying Yang, S. Gumhold, and C. Rother, "Learning analysis-by-synthesis for 6d pose estimation in rgb-d images," in *IEEE Intern. Conf. on Computer Vision*, 2015.

[22] V. Narayanan and M. Likhachev, "Discriminatively-guided Deliberative Perception for Pose Estimation of Multiple 3D Object Instances," in *Robotics: Science and Systems (RSS)*, 2016.

[23] P. J. Besl and N. D. McKay, "Method for registration of 3D shapes," *International Society for Optics and Photonics*, 1992.

[24] N. Mellado, D. Aiger, and N. K. Mitra, "Super 4pcs fast global pointcloud registration via smart indexing," *Computer Graphics Forum. Vol. 33. No. 5*, 2014.

[25] Q.-Y. Zhou, J. Park, and K. Koltun, "Fast Global Registration," *European Conference on Computer Vision*, 2016.

[26] M. Stark, M. Goesele, and B. Schiele, "Back to the future: Learning shape models from 3d cad data." in *British Machine Vision Conf.*, 2010.

[27] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik, "Aligning 3d models to rgb-d images of cluttered scenes," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2015.

[28] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Intern. Conf. on Machine Learning*, 2009.

[29] M. P. Kumar, B. Packer, and D. Koller, "Self-paced learning for latent variable models," in *Advances in Neural Information Processing Systems*, 2010.

[30] X. Liang, S. Liu, Y. Wei, L. Liu, L. Lin, and S. Yan, "Towards computational baby learning: A weakly-supervised approach for object detection," in *IEEE Intern. Conf. on Computer Vision*, 2015.

[31] E. Sangineto, M. Nabi, D. Culibrk, and N. Sebe, "Self paced deep learning for weakly supervised object detection," *arXiv preprint arXiv:1605.07651*, 2016.

[32] Blender. [Online]. Available: https://www.blender.org/

[33] Buller physics engine. [Online]. Available: http://bulletphysics.org/

[34] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, 1981.

[35] J. Hastings-Trew. (2012) Reproducing real world light. [Online]. Available: http://planetpixelemporium.com/tutorialpages/light.html

[36] A. Kimmel, A. Dobson, Z. Littlefield, A. Krontiris, J. Marble, and K. Bekris, "Pracsys: An extensible architecture for composing motion controllers and planners," *Int. Conf. on Simulation, Modeling, and Programming for Autonomous Robots*, pp. 137–148, 2012.

[37] Z. Littlefield, A. Krontiris, A. Kimmel, A. Dobson, R. Shome, and K. E. Bekris, "An Extensible Software Architecture for Composing Motion and Task Planners," in *Int. Conf. on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR)*, 2015.

[38] V. Azizi, A. Kimmel, K. E. Bekris, and M. Kapadia, "Geometric Reachability Analysis for Grasp Planning in Cluttered Scenes for Varying End-Effectors," in *IEEE CASE*, 2017.

[39] H. Shuai, N. Stiffler, A. Krontiris, K. E. Bekris, and J. Yu, "High-Quality Tabletop Rearrangement with Overhand Grasps: Hardness Results and Fast Methods," in *RSS*, Cambridge, MA, 2017.

[40] A. Krontiris and K. E. Bekris, "Dealing with Difficult Instances of Object Rearrangement," in *Robotics Science and Systems (RSS)*, 2015.