 DOI: 10.3217/978-3-85125-663-5-31

6D Object Pose Verification via Confidence-based Monte Carlo Tree Search and Constrained Physics Simulation

Dominik Bauer[1], Timothy Patten[1], Markus Vincze[1]

***Abstract*— Precise object pose estimation is required for robots to manipulate objects in their environment. However, the quality of object pose estimation deteriorates in cluttered scenes due to occlusions and detection errors. The estimates only partially fit the observed scene, or are physically implausible. As a result, robotic grasps based on these poses may be unsuccessful and derived scene descriptions may be unintelligible for a human observer. We propose a hypotheses verification approach that detects such outliers while, at the same time, enforces physical plausibility. On one hand, this is achieved by a tight coupling of hypotheses generation with the verification stage to guide the search for a solution. On the other hand, we integrate a constrained physics simulation into the verification stage to constantly enforce physical plausibility. By constraining the simulated objects to the most confident point correspondences, we prevent the estimated poses from erroneously diverging from the initial predictions. We thereby generate a plausible description of the observed scene. We evaluate our method on the `LINEMOD` and `YCB-VIDEO` datasets, and achieve state-of-the-art performance.**

I. INTRODUCTION

For robots to autonomously operate in the real world, they require a reliable estimate of the pose of objects around them to be able to manipulate objects of interest [17]. In human-robot interaction, it is furthermore essential for the robot to be able to explain why and how its estimates are computed. Thereby, the robot's actions become understandable to the human interaction partner as to build and maintain trust and transparency [3]. Such explanations help to make the robot's belief state and decisions understandable and the human interaction partner more tolerant to observed errors.

The recent SIXD challenge [10] shows advances in object pose estimation; but also current limitations. Pose estimation methods begin to fail when the objects of interest have little textural information, the robot perceives noisy RGB-D data, or the scene exposes a high amount of clutter. Due to detection errors, such as confusing different objects in the observation, the estimated poses can be far-off the actual object pose. Furthermore, the precision of the estimated poses can be low even matching the correct object. Besides falsely matching to parts of an object, not considering global scene consistency is a source for inaccuracies, resulting in estimates that may feature intersecting or floating objects. The computed scene descriptions are, therefore, potentially inconsistent on a global level and physically implausible. In object manipulation, for example, this may result in failed grasps.

Fig. 1: The initial pose hypothesis with the object half-way below the ground plane (left) and the refined pose after the verification stage with the object resting on the plane (right).

Existing hypotheses verification frameworks [2], [1], [21], [14], [15] aim to reduce inconsistencies in the scene description by reasoning about the scene on a global level, i.e., considering all hypotheses at once. Verification is considered a post-processing step in these frameworks, ignoring most of the information gathered during hypotheses generation. These methods construct complex cost functions [2], [1] for which parameters must be tuned for a specific dataset. Creating an exhaustive set of hypotheses [14] or performing search exhaustively [15] are alternative approaches to increase performance in the absence of discriminative information; albeit significantly increasing computation time.

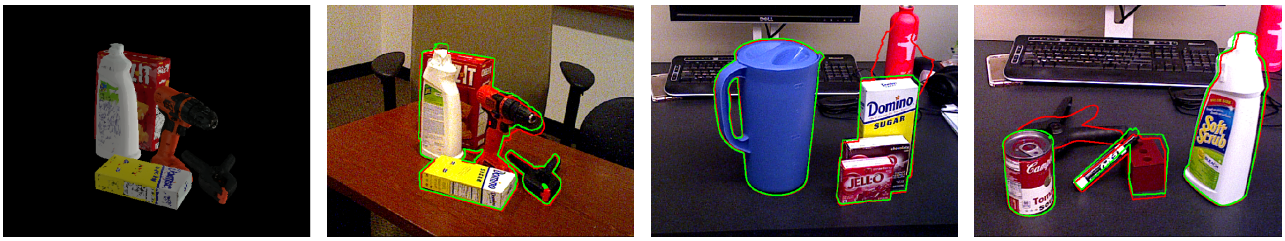
During hypotheses generation, information beyond the pose is generated that enables the discrimination between hypotheses. For example, the ranking, confidence or fitness of a pose hypothesis can be used to consider more promising candidates first. Also, the position and fitness of individual point correspondences with the observation contain information that can reduce the solution space the hypotheses verification has to consider.

To that end, we propose a two-staged framework that closely couples hypotheses generation and verification to leverage available information. We incorporate physics simulation into the verification stage similar to [14] but impose soft-constraints to limit the objects' movements to remain close to the original estimate. This is done by scaling the forces of the point constraints by the confidence of point correspondences. The verification is guided by a heuristic that uses the confidence of the generated hypotheses. The result is a subset of pose hypotheses that best explains the observation on a global scene level.

We propose a hypotheses verification framework based on Monte Carlo tree search (MCTS). By the use of physics simulation during the simulation phases of the tree search,

*Funded by the TU Wien Doctoral College TrustRobots and partially funded by OMRON Corporation.

[1]The authors are with Faculty of Electrical Engineering and Information Technology, Institute of Automation and Control, TU Wien, 1040 Vienna, Austria {bauer,patten,vincze}@acin.tuwien.ac.at



(a) Rendered view of the scene in (b). Note the cap on the model of the bleach bottle that is missing in (b) and (c).

(b) By enforcing the drill to stand on the table, we achieve a plausible and more precise pose.

(c) The bottle in the back is falsely matched to the drill during HG but detected as false positive during HV.

(d) Error case: The clamp is falsely omitted. The predicted foam brick (second from right) moves the pen resting on it.

Fig. 2: Qualitative examples. The red contours show the output of our HG, green contours show the solution after HV.

solutions are generated that are inherently physically plausible. We show improvement over previous work by:

- using confidence-based heuristic to guide verification
- integrating physics simulation, constrained by point correspondences, with hypotheses verification
- detecting false positives that are far-off from the true object pose while refining near-by hypotheses in a physically plausible manner
- achieving state-of-the-art performance on the LINEMOD and YCB-VIDEO datasets.

In the following, we discuss related work in Section II, give a detailed description of our proposed approach in Section III, and present results on LINEMOD and YCB-VIDEO using the ADD, ADD-S and VSD metrics in Section IV. A discussion of the improvements and limitations can be found in Section V.

II. RELATED WORK

A. Object Pose Estimation

Hodan et al. [10] compare several of state-of-the-art methods in object pose estimation and identify four major research directions. Template-based methods [8], [11] precompute different views of the objects of interest observed under a discrete set of rotation angles. During inference, the task is to first locate known objects in the observation and then find the corresponding view to deduce the object rotation. Methods based on Point Pair Features (PPF) [7], [18] are trained by precomputing these features for the objects of interest and storing them in a discretized hash table that is used to match scene points to model points via a voting scheme during pose estimation. For every two model points, such a feature consists of their distance, the angle between their normals and the angle between each normal and the connecting line. Methods based on 3D local descriptors, such as SHOT [16] or PPFH [6], find correspondences between features computed on a model and the observed scene to generate pose candidates that are then refined using ICP.

Finally, several learning-based methods [20], [12], [19] tackle pose estimation using CNNs. Xiang et al. [20] propose an architecture that jointly estimates semantic labels, translation and rotation from RGB images. Li et al. [12] furthermore include depth information in their architecture. Building on this work, Wang et al. [19] propose to fuse appearance features predicted from RGB with geometric features predicted from the depth image into a single pixel-wise embedding. Based on these pixel-wise features, a candidate pose is predicted per pixel. The highest scoring prediction per object instance mask is returned as a pose for this object. We base our hypotheses generation on this work.

B. Hypotheses Verification

Narayanan et al. [15] approach the pose estimation problem by generating an exhaustive set of possible scene configurations and then searching for the best solution in a verification scheme. Each scene configuration is rendered and the resulting depth image is compared to the observed depth image. While the resulting estimates have high accuracy, the authors also report an average runtime of 6.5 minutes.

Aldoma et al. [1] use SHOT to generate a set of pose hypotheses. The authors use Simulated Annealing to search for the subset of hypotheses that best fits the observation on a scene level. To quantify the fitness of a candidate solution, a cost function is constructed based on geometric cues. Additional terms penalize multiple assignments of scene points to different objects and a so-called "clutter term" penalizes hypotheses that only partly fit smooth surface patches. In follow up work [2], the cost function is extended by considering color information and a comparison of different meta-heuristics for finding a solution. Our solution definition is closely related to the formulation presented in this work.

Most related to our hypotheses verification method, Mitash et al. [14] propose the use of MCTS for hypotheses verification and the use of physics simulation to enforce scene consistency. The authors utilize the basic UCT algorithm for MCTS and evaluate their method on a non-public dataset.

III. OBJECT POSE VERIFICATION

Given a RGB-D image as input, object-instance masks and their corresponding class labels are estimated using SegNet [4]. To integrate the information from hypotheses generation (HG) with hypotheses verification (HV), we require a HG method to provide a set of diverse pose hypotheses, information on their quality as well as a set of correspondences that allow us to constrain the object pose in physics simulation. We use DenseFusion, proposed by Wang et al. [19], to

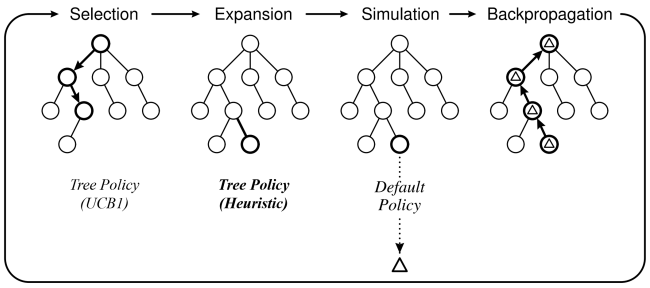


Fig. 3: Overview of the MCTS algorithm. For our method we adapt the tree policy during expansion (bold) and the computation of the reward (triangle). Adapted from [5].

generate such hypotheses; although our approach is able to deal with other HG methods as long as they can provide the required information just mentioned. For example, the PPF-based method by Vidal et al. [18] could be used instead.

The HV stage is tasked with determining the set of hypotheses that globally best explain the observed scene. For this, we initialize MCTS with the generated pose hypotheses. Starting from the empty set, MCTS adds one hypothesis per expansion and evaluates the new solution set by simulating the predicted scene. To determine which hypothesis should be selected next for expansion, we propose a heuristic based on the hypotheses' confidence values. MCTS is stopped after a fixed number of iterations to limit total runtime and the highest-reward solution is returned.

In the following, we describe our approach in more detail.

A. Hypotheses Generation

For each object-instance mask, DenseFusion predicts a pose hypothesis for each corresponding scene point and provides a confidence value for the respective estimate. The highest-confidence pose hypothesis is refined and returned as a pose estimate for this object instance. To generate a set pose hypotheses with more variation, we trigger HG repeatedly with different random samples of the observed object instance. This variation is desired as to generate a set of hypotheses that has a higher chance of containing the true object pose that should be found by HV. As a measure of quality for each pose hypothesis, the confidence value that is returned by the network is used. Finally, as each sampled point predicts a pose hypothesis and reports a confidence value, the 100 most confident hypotheses are selected as soft-constraints that should be satisfied during physics simulation in HV.

B. Hypotheses Verification

The task of HV is to select a subset of hypotheses that best explains the scene. A solution X can be described by $n \cdot N$ binary variables, where n is the number of hypotheses and N the number of object instances. Each $x \in X$ takes a value of 1 if the corresponding hypothesis is used in the solution and 0 otherwise. Exhaustively searching through all possible permutations quickly becomes intractable with increasing number of objects and hypotheses per object. For example, the `YCB-VIDEO` dataset contains scenes with 3 to 9 objects per scene. For 5 hypotheses per object and no further limitations on the solution there are 2^{15} to 2^{45} possible solutions.

1) Monte Carlo tree search: MCTS is an algorithm that is successfully used in game-playing to solve similar tasks with large search spaces and potentially costly state evaluations. An illustration of the basic algorithm is shown in Figure 3. The initial step in each iteration is to *select* the most promising candidate solution for which not all direct children have been expanded. Using the UCB1 policy, the algorithm balances the exploitation of known rewarding solutions with the exploration of new regions of the search tree. Once this node is selected, a new child solution is created and the tree is *expanded*. In the basic version, also used in previous work [14], this new child solution is chosen randomly. We propose to, instead, use a heuristic to expand more promising candidates first. The next step in the algorithm is to evaluate the expected reward of the new solution by *simulating* a rollout. In a rollout, the child solution is expanded using the default policy until a terminal state is reached. The reward for this terminal state is computed and *backpropagated*, starting from the new child solution and following the selection path up to the root node. Thereby, the reward statistics for each node can be updated for use by the UCB1 policy in the next iteration. The search can be stopped any time, offering speed to be traded for solution quality.

2) Solution space: Assuming there can only be one true hypothesis per object, we can reduce the number of possible solutions for the previous example of 3 object instances with 5 hypotheses each from 2^{15} to 915 possibilities. We model the problem as follows: Starting with the empty set as an initial candidate solution, we iteratively activate hypotheses. Activating more than one hypothesis per object instance is not allowed. The hypothesis for an object instance can therefore not change once it has been activated. Other subtrees of the search tree evaluate these alternative hypotheses.

3) Heuristic tree policy: Evaluation individual solutions is nevertheless costly. Guiding the search towards promising regions first allows the search to be stopped early, which saves computation time. We propose to leverage the confidence values of hypotheses compute during HG for this task. Instead of randomly selecting the next hypothesis, the probability of a hypothesis to be chosen is weighted by its confidence value. In a first step, one of the currently inactive object instances is chosen with a probability proportional to the sum of the confidence values of their inactive hypotheses. Next, one of the currently inactive hypotheses of this object instance is randomly chosen, again weighted by their confidence value. As a result, the search is guided towards the more promising objects and their hypotheses first.

4) Physics simulation: To determine the reward of a new candidate solution, we apply a physics simulation, render the object models in the resulting poses and compare the rendered depth values with the observed ones. However, naïvely applying physics simulation to a candidate solution has two main problems: First, the unknown mass distribution of the objects may cause unexpected ways. Without, for

example, considering the heavy battery pack of a power drill, the object easily topples over in simulation. Second, as the solution may include intersecting objects, the resulting repulsion forces can push close-by objects away or over.

The center of mass of the objects is moved close to their bottom, with respect to their canonical pose, to tackle the problem of unknown mass distribution. By computing a convex decomposition of the object meshes used as colliders, the effect of intersecting objects is lessened. In addition, soft-constraining the movement of the objects to their highest-confidence point correspondences limits the effect of the repulsion forces. Per point and confidence value, we set a constrain between the point on the object and its predicted position in world space. Each constrain can apply a maximal force that is proportional to its confidence value. The constrained physics simulation is able to generate plausible poses while circumventing the problem of exaggerating the instability resulting from estimation inaccuracy.

5) Reward function: After the physics simulation, the final step of evaluating a candidate solution consists of rendering the corresponding depth image and comparing it to the observed depth. We mask-out all points that have neither a depth value in the rendering nor the observation. The reward r is then computed as follows:

$$r = \sum \delta(R,S) \tag{1}$$

$$\delta_1(R,S) = \begin{cases} 1, & \text{if } |d_R - d_S| < \tau \text{ for } d_R, d_S \in R,S \\ -1, & \text{otherwise} \end{cases}$$
$$\delta_2(R,S) = \begin{cases} 1 - \frac{|d_R - d_S|}{\tau}, & \text{if } |d_R - d_S| < \tau \text{ for } d_R, d_S \in R,S \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where R is the masked rendered depth and S the masked observed depth. We use δ_1 with $\tau = 1cm$ on LINEMOD and δ_2 with $\tau = 3cm$ on YCB-VIDEO.

IV. RESULTS

For our evaluation, we want to compare the proposed HV pipeline (*Ours*) against two state-of-the-art pose estimation methods, the baseline method DenseFusion [19] (*Baseline*) and *PoseCNN* [20]. To ensure comparability with the two methods, we use the trained weights from [19] and the pre-computed segmentation masks provided by [20] for evaluation on the YCB-VIDEO dataset. Both methods also provide results for the LINEMOD dataset. However, the classes 3 and 7 of LINEMOD are not included in the pre-trained network and are thus missing from our evaluation. The evaluation uses the same test set as Wang et al. [19].

On the LINEMOD dataset, we evaluate the non-symmetric objects on the ADD [8], the symmetric objects on the ADD-S [20] metrics and all objects the VSD metric [10], [9]. The results on the ADD(-S) metric are reported in terms of recall score for a threshold of 0.1 times the object's diameter ($\leq 0.1d$). For the VSD metric, the threshold is set to 0.3. On the YCB-VIDEO dataset, all objects are evaluated on the ADD-S metric. We report the Area under Curve (AUC) as well as the recall scores for thresholds of $2cm$ and $1cm$. Qualitative results on this dataset are shown in Figure 2. Following Wang et al. [19], we compare against PoseCNN with DeepIM [20], [13] (*PoseCNN + DeepIM*) on LINEMOD and against PoseCNN with ICP (*PoseCNN + ICP*) on YCB-VIDEO.

A. Results on LINEMOD

For the LINEMOD dataset, we are able to report a significantly increased performance over the baseline method on both the ADD(-S) and the VSD metrics. Since the LINEMOD dataset features only one target object per scene, the results only show the performance of the constrained physics simulation and reward function. The MCTS has no impact on the solution as it will exhaustively try all solutions. We set the number of hypotheses per object instance to 10 and the threshold τ in our reward function to 1cm. The HG takes approximately 20ms per pose hypothesis and the HV takes approximately 60ms to evaluate a candidate solution. The total runtime, including the segmentation stage and additional processing, is 1–2s per test target.

Table I shows the results on the LINEMOD dataset for the ADD(-S) metric in the left sub-table and for the VSD metric in the right sub-table. Note that we do not report values for classes 3 and 7 as our HG does not include them.

B. Results on YCB-VIDEO

The YCB-VIDEO dataset features 3 to 9 objects per scene. In the test set used in [19] and [20], there are 3 to 6 objects per scene. Moreover, the pre-computed segmentation masks contain misdetected object instances. This allows us to evaluate the performance of our MCTS-based HV as false positives have to be omitted and different pose hypotheses may influence one-another.

Table II shows the results on the YCB-VIDEO dataset for the ADD-S metric. The results for *PoseCNN + ICP* are re-computed from the provided result poses [20]. The results for *Baseline* are taken from the same run as *Ours*. Overall, we report a slight increase of our HV method over previous work. As the results with a threshold of $2cm$ are already saturated for many classes, we also use a stricter threshold of $1cm$, showing an improvement of up to 3.3% on individual classes over the baseline. For the AUC measure, in accordance with [19], we set the highest threshold on the ADD-S to 10cm. The results on this measure show a slight improvement of *Ours* over the baseline of up to 0.5%. However, *PoseCNN + ICP* is the best performing method on several classes in terms of AUC.

The number of hypotheses per object instance is set to 5, the threshold τ in our reward function to 3cm and the number of iterations in the MCTS was limited to 300. This means that, out of all possible combinations of hypotheses for the 3 to 6 objects in the test set, at most 300 solutions are evaluated. We motivate this choice as follows: In the worst case of 6 objects with 5 hypotheses each, we spend 30 iterations on the initial expansion and need at least 105

TABLE I: Results on LINEMOD. Symmetric objects in italics, best results are highlighted bold. The difference between *Baseline* and *Ours* is given in parentheses. Left: Recall scores in percent for the ADD-S metric [20] for symmetric objects, ADD metric [8] otherwise. Right: Recall scores in percent for the VSD metric ([10], [9]) computed with $\tau = 0.02$, $\delta = 0.015$ and $\theta = 0.3$.

Class	PoseCNN+DeepIM [20], [13] $\leq 0.1d$	Baseline $\leq 0.1d$	Ours $\leq 0.1d$	Baseline $\theta < 0.3$	Ours $\theta < 0.3$
01 ape	77.0	91.3	**94.1** (+2.8)	96.5	**97.7** (+1.2)
02 vise	**97.5**	92.1	95.6 (+3.5)	83.4	**90.9** (+7.5)
04 camera	93.5	92.9	**97.1** (+4.2)	91.8	**96.2** (+4.4)
05 can	**96.5**	92.8	95.8 (+3.0)	90.2	**94.5** (+4.3)
06 cat	82.1	96.1	**97.1** (+1.0)	94.7	**97.0** (+2.3)
08 driller	**95.0**	87.4	90.6 (+3.2)	80.0	**87.3** (+7.3)
09 duck	77.7	91.6	**95.4** (+3.8)	97.5	**99.0** (+1.5)
10 eggbox	97.1	99.7	**99.7** (–)	81.7	**93.0** (+11.3)
11 glue	99.4	**100.0**	**100.0** (–)	86.5	**92.0** (+5.5)
12 puncher	52.8	89.9	**96.0** (+6.1)	90.0	**95.4** (+5.4)
13 iron	98.3	96.1	**98.5** (+2.4)	92.7	**97.0** (+4.3)
14 lamp	**97.5**	95.5	97.4 (+1.9)	89.9	**93.0** (+3.1)
15 phone	87.7	90.8	**95.4** (+4.6)	84.5	**91.7** (+7.2)
Overall	88.6	93.6	**96.4** (+2.8)	89.2	**94.2** (+5.0)

iterations to get to a full candidate solution of 6 hypotheses. To allow for different candidate solutions to be considered, and to add some safety in expectation of additional false positives while still keeping computation time reasonable, we increase this to a maximum number to 300 iterations. The hypothesis generation takes 20ms per pose hypothesis. The runtime per MCTS iteration including evaluation of the candidate solution is 60ms on average, resulting in a total runtime of about 18s.

Although we choose a tighter threshold than previous work [19], the performance on the dataset is saturated for some classes – a stricter metric would be required to further differentiate between methods. This is due to the ADD-S metric ignoring rotation errors around symmetry axes.

Another aspect that we observed is that the solution after the verification can be worse than simply using the first generated hypothesis per object. On one hand, this is due to the limited number of 300 solutions we explore during the verification. The search may, as a result, only return a local minimum. On the other hand, we rather strictly evaluate candidate hypotheses. The resulting reward seems to underestimate the quality of certain hypotheses or to aggressively classify them as false positive. To lessen this effect, we augment the final solution with high-confidence hypotheses after the search is stopped.

V. DISCUSSION

As discussed in Section IV, the performance on the LINEMOD and YCB-VIDEO datasets is already saturated. We plan to evaluate our method on more challenging datasets with, for example, heavier occlusion or texture-less objects. Stricter metrics and thresholds may also further differentiate between methods. For our goal of achieving a physically plausible scene description, a stricter metric could penalize intersecting objects or missing support relations.

However, a general problem when using physics simulation is the need for knowing the direction of gravity and, ideally, the supporting plane and all supporting objects. Without gravity, we can only resolve intersections. Without support, objects simply fall to the ground. How these cases can be handled would be a route for future improvements to the presented method. For example, further constraining object movement based on the observed evidence could remove the need to know all interacting objects a-priori.

The performance of our HV method is currently limited by the amount of solutions we can consider. The results on the YCB-VIDEO dataset, presented in Section IV-A, illustrate this problem: Since we only allow for 300 iterations, MCTS might not be able to explore better performing areas of the tree. A suboptimal candidate solution is returned in such cases. This problem could be tackled by constraining the search space further or constructing a more informative reward function. However, the main limiting factor is the runtime of an individual MCTS iteration.

In our current implementation, one iteration takes approximately 60ms. While this is an improvement over the runtime of 200ms per iteration reported by [14], it is still restrictive for use in a robotic application. The main part of these 60ms are due to the physics simulation with 30ms and the read-back of the rendered image with 20ms. Especially the read-back could be accelerated by computing the reward on the GPU and returning one value instead of rendered images.

VI. CONCLUSION

We presented a hypotheses verification method for 6D object pose estimation. Our approach is based on closely integrating information from hypotheses generation with verification. The confidence values per hypotheses are used to guide verification while the confidence values and positions of point correspondences are used to constrain physics simulation. This physics simulation is an integral step of our verification scheme that uses Monte Carlo tree search to find a solution that fits the observation in a globally consistent and plausible manner. Using our proposed hypotheses verification method, we are able to achieve state-of-the-art performance on both the LINEMOD and the YCB-VIDEO dataset.

TABLE II: Results on YCB−VIDEO (ADD-S [20]). Symmetric objects in italics, best results are highlighted bold. The difference between *Baseline* and *Ours* is given in parentheses.

Class	PoseCNN + ICP [20]			Baseline			Ours		
	AUC	$< 2cm$	$< 1cm$	AUC	$< 2cm$	$< 1cm$	AUC	$< 2cm$	$< 1cm$
002_master_chef_can	95.8	**100.0**	99.5	96.4	**100.0**	**100.0**	**96.5** (+0.1)	**100.0** (–)	**100.0** (–)
003_cracker_box	92.7	91.6	84.8	95.8	99.4	96.7	**96.3** (+0.5)	**99.7** (+0.3)	**98.6** (+1.9)
004_sugar_box	**98.2**	**100.0**	**100.0**	97.6	**100.0**	**100.0**	97.7 (+0.1)	**100.0** (–)	**100.0** (–)
005_tomato_soup_can	**94.5**	**96.9**	96.8	**94.5**	**96.9**	96.8	**94.5** (–)	**96.9** (–)	**96.9** (+0.1)
006_mustard_bottle	**98.6**	**100.0**	98.9	97.3	**100.0**	97.8	97.6 (+0.3)	**100.0** (–)	**99.4** (+1.6)
007_tuna_fish_can	97.1	**100.0**	97.7	97.1	**100.0**	99.4	**97.2** (+0.1)	**100.0** (–)	**99.7** (+0.3)
008_pudding_box	**97.9**	**100.0**	**100.0**	95.9	99.5	98.6	96.2 (+0.3)	99.5 (–)	99.1 (+0.5)
009_gelatin_box	**98.8**	**100.0**	**100.0**	98.0	**100.0**	**100.0**	98.1 (+0.1)	**100.0** (–)	**100.0** (–)
010_potted_meat_can	**92.7**	**93.6**	83.3	90.7	92.8	87.1	90.3 (–0.4)	92.4 (–0.4)	**87.3** (+0.2)
011_banana	**97.1**	99.7	95.0	96.2	99.7	98.4	96.7 (+0.5)	**100.0** (+0.3)	**99.7** (+1.3)
019_pitcher_base	**97.8**	**100.0**	99.7	97.5	**100.0**	99.7	97.7 (+0.2)	**100.0** (–)	**100.0** (+0.3)
021_bleach_cleanser	**96.9**	99.4	95.1	95.9	99.9	99.1	96.2 (+0.3)	**100.0** (+0.1)	**99.7** (+0.6)
024_bowl	81.0	54.9	42.9	89.4	94.6	55.4	**89.6** (+0.2)	**96.1** (+1.5)	**55.7** (+0.3)
025_mug	95.0	99.8	97.6	96.7	**100.0**	98.1	**96.8** (+0.1)	**100.0** (–)	**99.1** (+1.0)
035_power_drill	**98.2**	**99.6**	**99.3**	96.0	99.0	97.4	96.5 (+0.5)	**99.6** (+0.6)	98.7 (+1.3)
036_wood_block	87.6	80.2	74.4	92.8	**100.0**	88.0	**93.3** (+0.5)	**100.0** (–)	**91.3** (+3.3)
037_scissors	91.7	95.6	68.0	92.0	**100.0**	71.8	**92.4** (+0.4)	**100.0** (–)	**73.5** (+1.7)
040_large_marker	97.2	99.7	97.1	97.6	**100.0**	**100.0**	**97.7** (+0.1)	**100.0** (–)	**100.0** (–)
051_large_clamp	**75.2**	74.9	**67.3**	72.5	**78.7**	33.3	72.5 (–)	**78.7** (–)	33.7 (+0.4)
052_extra_large_clamp	64.4	48.8	**38.4**	**69.9**	74.8	17.2	**69.9** (–)	**75.1** (+0.3)	17.2 (–)
061_foam_brick	**97.2**	**100.0**	99.7	92.0	**100.0**	99.7	92.0 (–)	**100.0** (–)	**100.0** (+0.3)
Overall	93.0	93.2	**89.6**	93.2	96.7	88.9	**93.3** (+0.1)	**96.8** (+0.1)	89.5 (+0.6)

REFERENCES

[1] A. Aldoma, F. Tombari, L. Di Stefano, and M. Vincze, "A global hypotheses verification method for 3d object recognition," in *European conference on computer vision*. Springer, 2012, pp. 511–524.

[2] ——, "A global hypothesis verification framework for 3d object recognition in clutter," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 7, pp. 1383–1396, 2016.

[3] P. Andras, L. Esterle, M. Guckert, T. A. Han, P. R. Lewis, K. Milanovic, T. Payne, C. Perret, J. Pitt, S. T. Powers, *et al.*, "Trusting intelligent machines: Deepening trust within socio-technical systems," *IEEE Technology and Society Magazine*, vol. 37, no. 4, pp. 76–83, 2018.

[4] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[5] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in games*, vol. 4, no. 1, pp. 1–43, 2012.

[6] A. G. Buch, D. Kraft, and D. Odense, "Local point pair feature histogram for accurate 3d matching," in *Proceedings of British Machine Vision Conference*, 2018.

[7] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3d object recognition," in *Proceedins of IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 998–1005.

[8] 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 *Proceedings of Asian Conference on Computer Vision*, 2012, pp. 548–562.

[9] T. Hodaň, J. Matas, and Š. Obdržálek, "On evaluation of 6d object pose estimation," in *Proceedings of European Conference on Computer Vision*, 2016, pp. 606–619.

[10] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, *et al.*, "Bop: benchmark for 6d object pose estimation," in *Proceedings of European Conference on Computer Vision*, 2018, pp. 19–34.

[11] T. Hodaň, X. Zabulis, M. Lourakis, Š. Obdržálek, and J. Matas, "Detection and fine 3d pose estimation of texture-less objects in rgb-d images," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 4421–4428.

[12] C. Li, J. Bai, and G. D. Hager, "A unified framework for multi-view multi-class object pose estimation," in *Proceedings of European Conference on Computer Vision*, 2018, pp. 254–269.

[13] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox, "Deepim: Deep iterative matching for 6d pose estimation," in *Proceedings of European Conference on Computer Vision*, 2018, pp. 683–698.

[14] C. Mitash, A. Boularias, and K. E. Bekris, "Improving 6d pose estimation of objects in clutter via physics-aware monte carlo tree search," in *Proceedings of IEEE International Conference on Robotics and Automation*, 2018, pp. 1–8.

[15] V. Narayanan and M. Likhachev, "Perch: Perception via search for multi-object recognition and localization," in *Proceedings of IEEE International Conference on Robotics and Automation*, 2016, pp. 5052–5059.

[16] S. Salti, F. Tombari, and L. Di Stefano, "Shot: Unique signatures of histograms for surface and texture description," *Computer Vision and Image Understanding*, vol. 125, pp. 251–264, 2014.

[17] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," in *Proceedings of Conference on Robot Learning*, 2018, pp. 306–316.

[18] J. Vidal, C.-Y. Lin, and R. Martí, "6d pose estimation using an improved method based on point pair features," in *Proceedings of International Conference on Control, Automation and Robotics*, 2018, pp. 405–409.

[19] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6d object pose estimation by iterative dense fusion," *arXiv preprint arXiv:1901.04780*, 2019.

[20] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *arXiv preprint arXiv:1711.00199*, 2017.

[21] W. Zhou, C. Ma, and A. Kuijper, "Hough-space-based hypothesis generation and hypothesis verification for 3d object recognition and 6d pose estimation," *Computers & Graphics*, vol. 72, pp. 122–134, 2018.