Active 3D Classification of Multiple Objects in Cluttered Scenes

Yiming Wang[1], Marco Carletti[2], Francesco Setti[2], Marco Cristani[2] and Alessio Del Bue[1]
[1] Visual Geometry and Modelling (VGM) Lab, Istituto Italiano di Tecnologia, Genova, Italy
[2] Department of Computer Science, University of Verona, Verona, Italy*
yiming.wang@iit.it

Abstract

Autonomous agents that need to effectively move and interact in a realistic environment have to be endowed with robust perception skills. Among many, accurate object classification is an essential supporting element for assistive robotics. However, realistic scenarios often present scenes with severe clutter, that dramatically degrades the performance of current object classification methods. This paper presents an active vision approach that improves the accuracy of 3D object classification through a next-best-view (NBV) paradigm to perform this complex task with ease. The next camera motion is chosen with the criteria that aim to avoid object self-occlusions while exploring as much as possible the surrounding area. An online 3D reconstruction module is exploited in our system in order to obtain a better canonical 3D representation of the scene while moving the sensor. By reducing the impact of occlusions, we show with both synthetic and real-world data that in a few moves the approach can surpass a state-of-the-art method, PointNet with single view object classification from depth data. In addition, we demonstrate our system in a practical scenario where depth sensor moves to search and classify a set of objects in cluttered scenes.

1. Introduction

Assistive robots have attracted increasing attention from both academics and industries [5] with the objective of supporting daily life, in particular, of people with disability. Robots are desired to substitute human to interact or manipulate with entities in the physical world. More frequently than ever, such robots need to interact with and manipulate several objects in an unknown scene to perform specific tasks such as grasping and placing [10, 19, 31, 35].

One essential capability to achieve such task is to be able to perceive the environment with both geometric ("where are the objects?") and semantic understanding ("what are the objects?"). However, object recognition in unstructured and cluttered 3D environment can be very challenging with only single-shot-based method [26]. Moreover, perceiving objects' positions and their pose is not a trivial task even with a single object in an uncluttered environment. Multiple objects in arbitrary positions exacerbate the problem, possibly leading to catastrophic failures if the object detector/pose estimator makes wrong decisions about the scene structure. Occlusions can contribute greatly to the loss of accuracy of vision systems which are fundamental to correctly recognise and estimate the 6D pose of objects. As a consolidated experimental fact, even the best single object classifiers decrease their performance when consistent occlusions appear [15].

In this context, active vision plays a fundamental role in best analysing the objects in the scene given the current observation and deciding which next best view (NBV) might have the chance to achieve a better recognition score [1, 21, 24, 25, 29] . In particular, when scenes are cluttered, objects are often incorrectly classified since they are possibly occluded by itself or other elements in the environment. Active object recognition (AOR) becomes a promising strategy for actively covering more viewpoints which eases the classification task [2, 7, 17, 25].

Following this idea, our paper proposes an active strategy to 3D object classification (see Figure 1), which can iteratively select viewpoints where clutter is reduced (*i.e.* more visible object surfaces) and thus improving the probability of best classifying the objects. Moreover, our approach also advocates the use of online 3D reconstruction of the scene to better solve for the 3D classification problem: As soon as a new depth frame of the scene is available, the method integrates the frame into a 3D representation of the scene. The classification of objects runs over the reconstructed point cloud instead of single depth frames. To select the NBV, our active strategy initially builds a scene representation by aligning the canonical volumes with the classes and poses estimated from the reconstructed point clouds, and then approximates the visibility of objects on

*This work has been partially supported by the project of the Italian Ministry of Education, Universities and Research (MIUR) "Diparti-menti di Eccellenza 2018-2022".

image plane by projecting the bounding cuboid of each canonical volume through a z-buffer indicating for occlusions. Each candidate viewpoint is further weighted based on its vicinity to already visited viewpoints to avoid revisit. We validate the proposed active recognition framework using a Kinect-equipped UR5 robotic arm. This setup is functional for the repeatability and assessment of the real experiments but our NBV proposed method can be generalised to other robotic scenarios. With both synthetic and real-world dataset, we demonstrate that the proposed method outperforms the baseline methods as well as multiple variants of our proposed NBV criteria.

The rest of the paper is organised as follows: Section 2 presents a review on related literature; Section 3 details our proposed framework, Section 4 contains the experimental results and discussion. Finally, Section 5 draws conclusions.

2. Related Work

In this section, we will review the state-of-the-art in active object classification considering two main aspects: multiview/3D object classification and motion policies for active classification.

Multiview/3D object classification. Several approaches have been proposed for 3D object classification, and can be categorised as view-based, voxel-based and point-cloud-based techniques. In view-based methods each 3D shape is represented by a set of frames generated from multiple viewpoints. Traditional approaches train one classifier for each viewpoint, assuming that the object lies in a known pose; these methods mostly use hand-engineered features like Fourier descriptors [6], local Gabor filters [8] and Fisher vectors [32]. In [36], a CNN model is trained to extract features from each available view, while a pooling layer fuses these features together and passes them to a subsequent NN architecture for classification. RotationNet [16] is a CNN-based model that takes as input a partial set of multiview RGB images and jointly estimates object's pose and category. All these methods are low-dimensional in the input space, computationally efficient and fairly robust to 3D shape representation artefacts such as holes and noise; despite that, they miss the ability to generalise over the complex 3D shape of an object.

In voxel-based methods, depth data are represented in a fixed-size 3D space in form of occupancy maps. In VoxNet [22] each voxel is assumed to have a binary state, occupied or unoccupied, while in ShapeNets [41] the 3D shape is represented as a probability distribution of binary variables; both of them use a 3D CNN architecture to perform object recognition. These methods suffer from the rigid space representation that limits the expressiveness of the input data; this limitation is overcome by point-based methods.

PointNet [26] learns a set of spatial features of each point independently and then accumulates the features by a symmetric function (*i.e.* max-pooling layer). This model has a relatively simple architecture that takes as input the complete point cloud of an object and performs single object recognition and part segmentation. Subsequent models have shown that the classification performance can be further improved by considering the neighbourhoods of points rather than treating points independently due to a better leverage on the local structure features [27, 34]. Despite working very well on single objects where the whole point cloud is available, these methods suffer when part of the point cloud is missing, *i.e.* in case of occlusions. In this paper we leverage the power of point cloud representation by proposing a strategy to intelligently explore the environment and acquire data to fill the gaps in the 3D representation of objects.

Motion policies for active classification. Several recent works on AOR model the problem with Reinforcement Learning techniques, such as Partially Observable Markov Decision Processes (POMDP) solved by means of point-based algorithms [1, 29], Monte Carlo tree search [24], Self-Organising Maps [3] or Belief Tree Search [21]. Despite presenting very advanced planning strategies, all these methods are very expensive to compute online.

A complementary strand of works focus on selecting the NBV within a finite set of candidates by maximising a cost function in the form of the unknown volume explored [2], the information gain [25], the conditional entropy [17], or exploiting unsupervised features learned from depth-invariant patches using sparse autoencoders [7]. All these methods require the 3D model of the object to search. To relax this hypothesis, Wu *et al.* [41] propose a feature-based model to compute the NBV in 2D space by predicting both visibility and likelihood of feature matching in a mobile robot setup; unfortunately this approach can only handle very simple object geometries. Many AOR approaches assume that there is a single object in the scene and its location is known [11, 18, 28, 29, 37, 38]. In this paper we address the case of multiple objects in the scene in unknown locations, handling occlusions and providing an iterative solution to the NBV selection problem by accounting for all the objects as a whole.

3. Active 3D Object Classification Framework

Let us consider a scenario where N objects belonging to the set of classes $\mathcal{C}$ are distributed in an area and a depth sensor with known camera pose acquires a set of depth frames

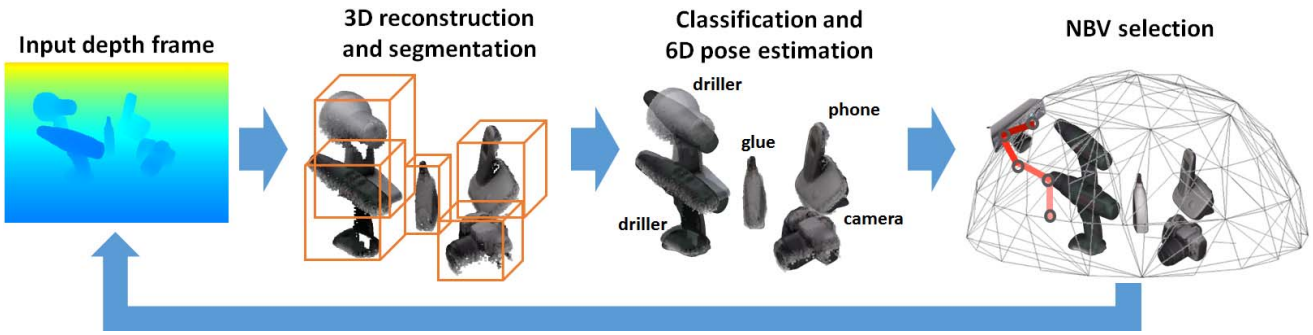

Figure 1. Overall view of the proposed active 3D classification system with a typical scenario with a set of objects occluding each other. The system reconstructs the scene and performs classification and pose estimation with refinement using geometric alignment. Given the 3D scene, the system chooses a next best view that maximises the visible object surfaces while avoiding already visited areas in order to achieve a better 3D classification (best viewed in colour).

at each camera move[1]. Our goal is to assign the correct label to each of the N objects using a finite set of camera moves. The overall view of the proposed active 3D object classification pipeline is depicted in Figure 1. We assume to have a CAD model for each object class. The number of objects in the scene is also assumed to be known to simplify the process of point cloud segmentation.

At each time step the sensor acquires a depth map of the scene and we isolate the foreground by first truncating the depth within a predefined distance and then removing the plane where the objects are lying on, i.e. $z = h$ where h is the height of table surface w.r.t to the world coordinate. The foreground is then segmented in object candidates (see Section 3.1) and each segment is processed by a PointNet [26] model to generate class candidates. For each segment, we take into account the top class candidates for further refinement in the pipeline. Given the segmented point cloud (for each segment) and the class candidates, we use DenseFusion [39] to provide an initial pose estimation (see Section 3.2). Then we use the Iterative Closest Points (ICP) algorithm [30] (see Section 3.2) to perform geometric refinement among the top candidate labels and updates the segment label as well as its pose.

Lastly, we select the NBV that maximises the visibility of all objects (accounting for occlusions) while avoiding already visited areas. We approximate the visibility by projecting the corners of the bounding cuboid of each segment onto the image plane. To account for occlusions, we project the segments sequentially with a z-buffer, from the closest to the farther from the sensor. The number of pixels within the convex hull bounded by the projected corners of the object is used to approximate the visibility of each segment. We keep moving the sensor until the camera positions reaches a fixed number of steps.

[1]Computing a reliable camera pose while the sensor is moving is not the focus of this paper. The sensor's pose can be given by direct kinematics if the camera is equipped on a robot, by a SLAM approach if hand-held, or by integrating both information.

3.1. 3D Reconstruction and Segmentation

Let the observation input be $\mathcal{O}(t) = \{\mathcal{D}(t),\ \mathbf{v}_W(t)\}$ at a time instance t, where $\mathcal{D}$ refers to a depth map and $\mathbf{v}_W(t) = \{\mathbf{R}_W(t),\ \mathbf{t}_W(t)\}$ refers to a viewpoint pose characterised by rotation matrix $\mathbf{R}(t)$ and 3D translation $\mathbf{t}_W(t)$ in the world coordinate. We acknowledge that there are a few popular real-time SLAM methods with RGB-D stream, such as KinectFusion [23] or ElasticFusion [40], that perform dense reconstruction without the prior knowledge of camera poses. While in our setting, since the camera poses are available thanks to the known robotic kinematics and hand-eye calibration, we opt to a volume integration method using the Truncated Signed Distance Function (TSDF) [43, 20].

At each time t, the depth image $\mathcal{D}(t)$ is registered to the previously reconstructed point cloud (if $t \neq 0$, otherwise we use the first depth image as a reference) using the camera viewpoint pose $\mathbf{v}_W(t)$ to produce a canonical volumetric representation of the scene. The reconstructed scene is firstly truncated within a cubic space of interest then segmented by means of DBSCAN algorithm for unsupervised clustering [9]. However, due to object's self-occlusion and inter-object occlusions, there is a tendency to oversegment. We further apply K-means clustering [13] on the center points to force the generation of N clusters. In such way, we produce a set of point cloud segments $\mathcal{S}(t) = \{S^1(t), \ldots,\ S^N(t)\}$, where each segment corresponds to an unknown object.

3.2. 3D Object Detection and Pose Estimation

Classification with partial reconstruction. We build our classifier based on PointNet architecture, which directly

takes unstructured point clouds as input [26]. The classification network firstly applies input and feature transformations in order to manage unordered point clouds, where transformation functions are trained as multi-layer perceptron (MLP) networks. Secondly, it aggregates points features by max pooling. Final global features, that are the shape features of the input point cloud, have been fed to a MLP to extract the classification scores for the number of classes $M = |\mathcal{C}|$.

At run time, we provide each point cloud segment $S^j(t)$ after zero-mean and normalisation into our fine-tuned PointNet classifier (check Section 4 for details) at each time step t. The classifier returns a score vector $\mathbf{z}^j(t)$ over all the classes in $\mathcal{C}$ such that $\mathbf{z}^j(t) = \{z_1^j(t), \dots, z_M^j(t)\}$, $\Sigma_1^M z_m^j = 1$. With a max pooling strategy, each segment $S^j(t)$ has a prior class prediction $C^{j-}(t)$ given by:

$$C^{j-}(t) = \underset{m \in \mathcal{C}}{\arg\max}\ \mathbf{z}^j(t). \tag{1}$$

In addition to the most likely class prediction, we also cache the top most likely classes, $\mathcal{C}^{j-}(t)$ for further refinement. We use the superscript "$-$" to indicate all the classes/poses estimation prior to the refinement stage.

In order to obtain the initial pose guess at $t = 0$ for each segment, we make use of a CNN-based pose estimator, DenseFusion [39] that takes as inputs the cropped depth images corresponding to an object class. More in details, for each segment $S^j(t)$, we input to DenseFusion the class prediction $C^{j-}(t)$ and the segmented observation $\mathcal{O}^j(t)$ obtained via camera projection using the pre-calibrated camera intrinsics and the viewpoint pose $\mathbf{v}_W(t)$, and obtain the prior pose estimate $\mathbf{p}_C^{j-}(t)$ for each segment in the camera coordinate. By applying coordinate transformation, $\mathbf{p}_W^{j-}(t) = \mathbf{p}_C^{j-}(t)\mathbf{v}_W(t)$, we obtain the prior pose estimate for each segment in the canonical 3D representation (*i.e.* the world coordinates).

Note that at the initial steps of the algorithm, the pose obtained can be inaccurate, in particular for the translation, due to wrong classification predictions. We further refine $\mathbf{p}_W^{j-}(t)$ by anchoring the estimated translation vector to the centroid of each segment. Although the centroid of each segment does not necessarily coincide with the centroid of its corresponding object's canonical volume, such rough translation fixation can facilitate a reasonable pose initialisation for the geometric refinement as described in the next section.

Geometric class and pose refinement. Since the 3D classifier can provide unreliable class prediction, especially at early stages, we propose a new refinement strategy using a geometrical approach as sketched in Figure 2. First, each point cloud segment $S^j(t)$ will be aligned with each top-ranking classified object using the associated 3D CAD

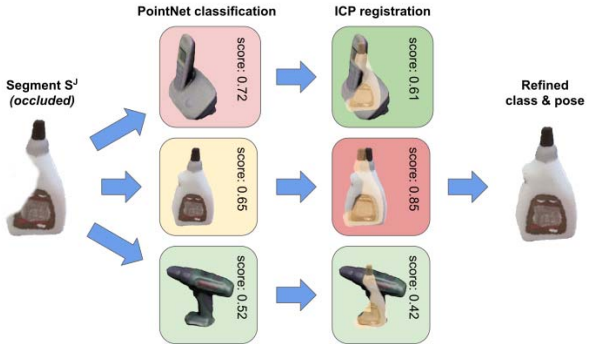

Figure 2. An example for the geometric class and pose refinement procedure. A reconstructed segment S^j is classified with top three candidate classes (highest classification confidence on top). By aligning the segment with the CAD models of the three candidate classes using ICP, the class and object pose can be refined by selecting the best aligned model (best viewed in colour).

model. Let $O^m \in \mathcal{C}^{j-}(t)$ be the 3D model of a candidate object in the point cloud. The estimated pose at the previous step $\mathbf{p}_W^{j-}(t)$ is used as an initialisation for the Iterative Closest Point (ICP) method [44]. After registration, we have a set of registered points, $S_m^{j,R}(t)$ and $O_m^{j,R}(t)$, belonging to the segment point cloud $S^j(t)$ and the object model O^m respectively. Together with the registered points, ICP also provides the pose transform $\Delta\mathbf{p}_m^j(t)$ that is applied to align the object model to the point cloud segment.

We quantify how well two point clouds are registered, given possibly mis-classified classes, by the ratio of the registered points over the total number of points of the two point clouds. Let $\rho_m^j(t)$ be the registration ratio of the segment and a candidate object model, and we can compute $\rho_m^j(t)$ as:

$$\rho_m^j(t) = \frac{|S_m^{j,R}(t)|}{|S^j(t)|}\frac{|O_m^{j,R}(t)|}{|O^m|}, \tag{2}$$

where the operation $|\cdot|$ gives the number of points of a point cloud. The value $\rho_m^j(t)$ is larger when the 3D reconstruction of the object is complete and when the candidate class prediction is the correct one. As a result, we can obtain the refined object class at time step t as:

$$C^j(t) = \underset{O^m \in \mathcal{C}^{j-}(t)}{\arg\max}\ \rho_m^j(t). \tag{3}$$

The refined object pose at time step t will be updated as:

$$\mathbf{p}_W^j(t) = \Delta\mathbf{p}_m^j(t)\mathbf{p}_C^{j-}(t). \tag{4}$$

3.3. Next Best View Selection (NBV)

The NBV is selected among a set of candidate viewpoints $\mathcal{V}_W(t) = \{\mathbf{v}_W^1(t), \dots, \mathbf{v}_W^I(t)\}$ at time step t. The candidate viewpoints $\mathcal{V}_W(t)$ are defined as those viewpoints that lie within the circular range of the current viewpoint

$\mathbf{v}_W(t)$ with radius r. The movement aims to achieve more complete object reconstruction, while avoiding to revisit the areas that the camera has already visited.

We devise a utility $U_i(t)$ for each candidate viewpoint $\mathbf{v}_W^i(t) \in \mathcal{V}_W(t)$ that encourages high object surface visibility and punishes revisiting same areas. Let $O_i^j(t)$ be the function that quantifies the visibility of a segment $S^j(t)$ at viewpoint $\mathbf{v}_W^i(t)$. We approximate the visibility for segments by replacing each segment with its 3D model of the predicted class aligned by the estimated pose as given in the previous section. The corners of the bounding cuboid of each aligned object model are projected onto the image plane. The projection is then performed sequentially using a z-buffer, *i.e.* an ordered vector of the distance between the segments to the camera, in order to account for occlusions. In this way, $O_i^j(t)$ is defined as the number of visible pixels within the convex hull bounded by the projected corners for each segment. The visibility utility at viewpoint $\mathbf{v}_W^i(t)$ that accounts for all segments is defined as the sum of visibility of each segment:

$$O_i(t) = \sum_{S^j(t) \in \mathcal{S}(t)} O_i^j(t). \tag{5}$$

In order to penalise the system to visit already seen areas, we further introduce a weight $W_i(t) \in [0, 1)$ for each candidate viewpoint. The value of $W_i(t)$ is designed to be smaller if the candidate viewpoint is closer to the viewpoint that has been visited, otherwise the value will be larger. Let the set of previously visited viewpoints be $\mathcal{V}'_W(t) = \{\mathbf{v}_W(\tau) | \forall \tau \in [1, t)\}$, and let $D(\tau, t)$ be the distance between the candidate viewpoint and a visited viewpoint at a previous time step τ. We make use of the normalised Gaussian distribution over the distance $D(\tau, t)$ in order to compute $W_i(t)$, such that:

$$W_i(t) = \max_{\mathbf{v}(\tau) \in \mathcal{V}'(t)} 1 - G\left(D(\tau, t), \mu, \sigma\right), \tag{6}$$

where $G(\cdot)$ defines the normalised Gaussian distribution function with the mean $\mu = 0$ and the standard deviation $\sigma = \frac{r}{2}$, where r is the radius that defines the set of candidate view points.

The final utility $U_i(t)$ combines the visibility utility $O_i(t)$ and the weight $W_i(t)$. Because the value of $W_i(t)$ is constrained within $[0,\ 1)$ while $O_i(t)$ is not constrained, for fair combination, we normalise $O_i(t)$ by its maximum value within $\mathcal{O}(t) = \{O_i(t) | \forall \mathbf{v}_i(t) \in \mathcal{V}(t)\}$, *i.e.* $N(O_i(t)) = \frac{O_i(t)}{\max(\mathcal{O}(t))}$. The utility $U_i(t)$ is therefore computed as:

$$U_i(t) = W_i(t) N\left(O_i(t)\right). \tag{7}$$

The NBV $\mathbf{v}_W(t)$ is selected as the one providing the highest utility:

$$\mathbf{v}_W^*(t) = \underset{\mathbf{v}_W^i(t) \in \mathcal{V}_W(t)}{\arg\max}\ U_i(t). \tag{8}$$

(a) Synthetic scene (b) Real scene

Figure 3. Examples of the synthetic and real scenes of our dataset, where severe inter-object occlusion can be observed.

Once the NBV is selected, we can move the depth sensor in the desired position if no stop condition is satisfied and acquire a new measurement $\mathcal{O}(t+1) = \{\mathcal{D}(t+1),\ \mathbf{v}_W(t+1)\}$ and iterate the procedure until the method converges.

4. Experiments

We first describe how we create a dataset with ground truth information followed by the experimental protocol. Results over several trials with both synthetic and real data will prove the effectiveness of the proposed active classification method.

Dataset creation. We selected 7 objects from the LINEMOD dataset [14] and generated a set of different scenarios with 5 objects in each one. In every scene, all objects lie on a planar surface defined at $z = 0$ and are placed in their *canonical pose*, *i.e.* with the z axis of the model's reference frames facing up. The dataset is generated with both synthetic and real-world setup.

We generated 7 synthetic scenes where depth maps are rendered using Blender by projecting the models on 100 viewpoints uniformly distributed on a hemispherical surface (*i.e.* $z \geq 0$) with the sphere's centroid coinciding with the centroid of all the objects in the scene and the radius of 1m. The rendering engine is set up to emulate a real acquisition device: the pinhole camera matrix is the same as a Kinect V1 device with resolution 640×480 pixels and focal length of 26mm.

In addition, we acquired two scenes in a real-world setup with a Kinect-equipped Universal Robots UR5 using 3D-printed instances of the selected object classes. This specific robotic configuration is chosen for the experiments since it allows accurate camera poses and repeatability during the testing procedure. The arm and sensor have been hand-eye calibrated in order to obtain the correct camera viewpoint poses from the robot's encoder poses. We used ArUco fiducial markers and its off-the-shelf libraries to annotate the pose of each object and to perform hand-eye calibration of the system [12]. Each scene contains 5 objects where 3 of

them are to be classified and the other 2 are used to simulate generic clutter. For each scene, we acquired RGB-D observations from a pre-defined set of 138 viewpoints sampled on a hemispherical surface of radius 0.9m centered in the centroid of the objects. The viewpoints are approximately uniform on the hemisphere due to the constrained reachability of the robotic arm.

PointNet fine-tuning. The original PointNet is trained on the ShapeNet and ModelNet dataset which do not include most of the object classes that are supported by the DenseFusion pose estimator which is trained on YCB_Video [42] and LINEMOD dataset [14]. In order to have compatible classifiers and pose estimators, we train the PointNet network by generating a new dataset covering the classes of the LINEMOD objects. The classes are identified based on the availability of the 3D models of object instances from existing dataset (*e.g.* VANDAL dataset [4]) and web resources (*e.g.* 3D Warehouse[2]). For each class, we have 3D models of eight instances in the format of dense point clouds.

At training time, each point cloud contains 2500 points uniformly sampled from the object surface. Each cloud is zero-mean and normalised into a unit sphere as in the standard PointNet training. In order to be more robust to rotation, we follow the data augmentation strategy proposed by the original PointNet work [26] by randomly rotating each point cloud by an angle in the range $[0,\ 2\pi]$ along Y axis. At training phase, PointNet converges in 250 epochs on a total of 7 classes. We used Adam optimizer with learning rate of 0.001, decreasing it by a factor of 2 every 20 epochs, as suggested in the original paper [26].

Evaluation analysis. We first test the PointNet model by feeding it with point clouds extracted directly from depth maps at each time step, named as **PointNetSingle**, as well as with point clouds covering all the 3D shape, named as **PointNetFull**. Please note that PointNetFull uses all the available information from the scene (*i.e.* complete point clouds for each object) and thus this can be considered as the upper bound that can be reached with every classifier tested.

As for the evaluation of active classification, we compare our motion strategy, **VisOcclHist**, for improving the classification performance against three baseline motion strategies: Random, VisNoOccl and VisOccl. All baseline strategies follow the same pipeline for active classification with the only difference in NBV selection. In **Random** strategy, the system randomly selects the next viewpoint to visit within the candidates that have not been visited yet. The **VisNoOccl** strategy selects the next viewpoint using the visibility score that maximizes the area of visible object surfaces without accounting for occlusions and already observed portions of the objects. The **VisOccl** strategy selects the next viewpoint using the visibility score that maximizes the area of visible object surfaces accounting for the occlusions but not for the previously observed points.

In all the cases, the next viewpoint is selected within a restricted set of points that are at a maximum distance of 0.5m from the current position. In addition, we also validate for all the baselines the classification improvement brought by the refinement module using Geometric Refinement (see Figure. 2) after the PointNet classifier (named with the suffix **GR**).

As for the evaluation metrics, we use standard classification measures: accuracy, precision, recall and F_1 score. We compute all these metrics for each class and then we average over all the classes, this strategy is usually dubbed as *macro-averaged*. Please note that this leads to accuracy scores higher than precision and recall since in a one-vs-all setup true negatives are usually higher than true positives [33].

Results discussion. Classification results are reported in Table 1. For fair comparison, all active classification methods stop when the maximum number of moves is reached, that is $T = 10$ in our experiments; results are computed at the final step of each run. All results are averaged over 10 runs with a random starting position at each run.

The introduction of GR improves results of about 10% on F_1 score on synthetic scenes, and increases to about 20% in real scenarios. This indicates that the GR can effectively correct the classification in terms of more true positives. The improvement by GR is higher in real scenes compared to the synthetic scenes because the depth images from the real acquisition is noisier compared to the synthetic dataset, which makes the naïve PointNet classification worse. For real scenes, we also observe marginal classification improvement, in terms of the F_1 score brought by the NBV criterion when considering the occlusion on visibility and the weight for avoiding visited areas. All experiments are performed on a Alienware Aurora Desktop with i7 core. The averaged processing time between consecutive moves for reconstruction, segmentation, geometric refinement and NBV are 0.12 s, 0.06 s, 0.88 s and 0.19 s, respectively.

Figure 4 shows the averaged classification performance in terms of F_1 score over time in both synthetic and real scenes. Without GR, the proposed active strategy can mostly outperform the random strategy in both synthetic and real scenes. With GR, the performance of all active strategies are boosted at each time step. To conclude, the proposed NBV strategy stands out with respect to all the baseline approaches, especially at increasing time steps as the method more efficiently covers the scene while avoid-

[2]https://3dwarehouse.sketchup.com

Table 1. Object classification results after the system has reached the stop condition ($T = 10$). Values are averaged over 10 runs for each scenario with random starting points. (Best results are in bold, upper bounds are in italic.)

	Synthetic				Real			
Approach	Accuracy	Precision	Recall	F1 score	Accuracy	Precision	Recall	F_1 score
PointNetSingle	0.80	0.32	0.40	0.34	0.71	0.37	0.41	0.38
PointNetSingleGR	0.81	0.39	0.50	0.42	0.71	0.40	0.48	0.43
PointNetFull	*0.90*	*0.64*	*0.71*	*0.67*	*0.78*	*0.50*	*0.67*	*0.56*
PointNetFullGR	*0.91*	*0.70*	*0.77*	*0.72*	*0.89*	*0.75*	*0.83*	*0.78*
Random	0.87	0.55	0.62	0.57	0.73	0.41	0.55	0.46
RandomGR	0.89	0.63	0.72	0.66	0.82	0.63	0.73	0.66
VisNoOccl	0.87	0.54	0.62	0.56	0.72	0.41	0.53	0.45
VisNoOcclGR	0.89	0.62	0.71	0.65	0.85	0.66	0.75	0.69
VisOccl	0.87	0.53	0.61	0.56	0.76	0.46	0.61	0.51
VisOcclGR	0.89	0.63	0.72	0.66	0.83	0.64	0.73	0.67
VisOcclHist	0.87	0.54	0.62	0.57	0.78	0.50	0.62	0.54
VisOcclHistGR	**0.90**	**0.64**	**0.73**	**0.67**	**0.87**	**0.70**	**0.80**	**0.73**

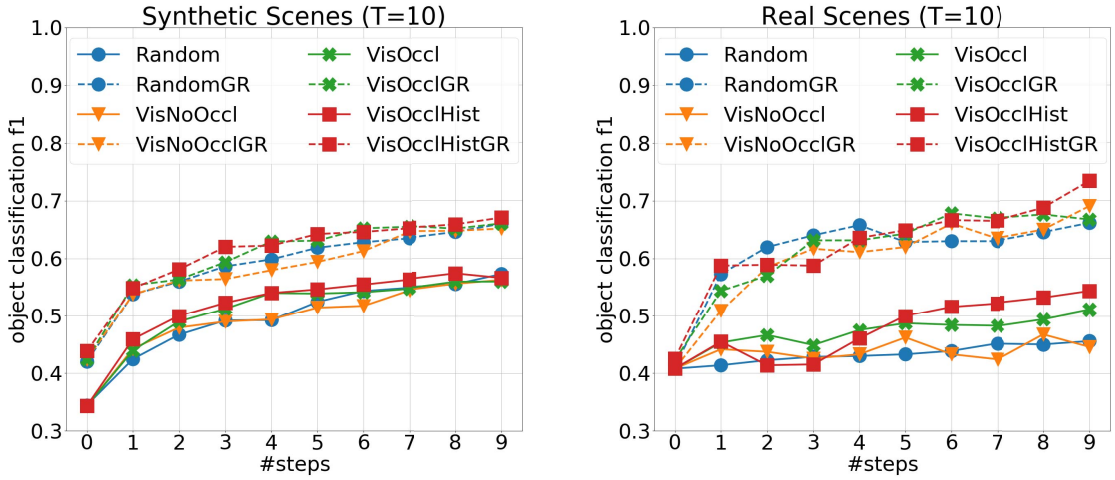

Figure 4. Comparison of active classification methods using both the synthetic (left) and real (right) datasets. Our method shows steady improvement with increasing number views, outperforming all the baselines.

ing to revisit previous viewpoints. The supplementary material shows a video of the robotic movement with various NBV criteria along with the classification performance at each step.

5. Conclusions

In this paper we have proposed a novel active vision approach to improve 3D object classification through shape reconstruction using depth data. Experimental results in both synthetic and real scenarios with severe occlusions show that both geometric refinement and NBV improve the object classification performance compared to the method using only a single depth frame, while approaching the performance achieved by the naïve PointNet with complete object point cloud. We performed a detailed study to demonstrate the effectiveness of each NBV criterion with/without the geometric refinement. Both the consideration of occlusion and view-point history in NBV brings marginal improvement while the geometric refinement improves up to about 20% in terms of F_1 score. As future work, we plan relax the proposed method to address 3D object recognition with arbitrary number of objects arranged in more complex scenes. It is also of our interests to improve the method with a learnt metric for NBV by encoding the status of object reconstruction.

References

[1] N. Atanasov, B. Sankaran, J. Le Ny, G. J. Pappas, and K. Daniilidis. Nonmyopic view planning for active object classification and pose estimation. *IEEE Transactions on Robotics*, 30(5):1078–1090, 2014.

[2] J. E. Banta, L. M. Wong, C. Dumont, and M. A. Abidi. A next-best-view system for autonomous 3D object reconstruction. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 30(5):589–598, 2000.

[3] G. Best, J. Faigl, and R. Fitch. Online planning for multi-robot active perception with self-organising maps. *Autonomous Robots*, 42(4):715–738, 2018.

[4] F. M. Carlucci, P. Russo, and B. Caputo. A deep representation for depth images from synthetic data. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 1362–1369, May 2017.

[5] K. Charalampous, I. Kostavelis, and A. Gasteratos. Recent trends in social aware robot navigation: A survey. *Robotics and Autonomous Systems*, 93:85 – 104, 2017.

[6] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung. On visual similarity based 3d model retrieval. *Computer Graphics Forum*, 22(3):223–232, 2003.

[7] A. Doumanoglou, R. Kouskouridas, S. Malassiotis, and T. Kim. Recovering 6d object pose and predicting next-best-view in the crowd. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3583–3592, Jun. 2016.

[8] M. Eitz, R. Richter, T. Boubekeur, K. Hildebrand, and M. Alexa. Sketch-based shape retrieval. *ACM Transactions on Graphics*, 31(4):31:1–31:10, 2012.

[9] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[10] K. Fang, Y. Zhu, A. Garg, A. Kurenkov, V. Mehta, L. Fei-Fei, and S. Savarese. Learning task-oriented grasping for tool manipulation from simulated self-supervision. *arXiv preprint arXiv:1806.09266*, 2018.

[11] S. Foix, G. Aleny, J. Andrade-Cetto, and C. Torras. Object modeling using a tof camera under an uncertainty reduction approach. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 1306–1312, May 2010.

[12] S. Garrido-Jurado, R. Muñoz Salinas, F. Madrid-Cuevas, and M. Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, Jun. 2014.

[13] J. A. Hartigan and M. A. Wong. A K-means clustering algorithm. *Journal of the Royal Statistical Society (Applied Statistics)*, 28(1):100–108, 1979.

[14] 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 *Proc. of Asian conference on Computer Vision (ACCV)*, pages 548–562, Nov. 2013.

[15] E. Hsiao and M. Hebert. Occlusion reasoning for object detection under arbitrary viewpoint. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(9):1803–1815, 2014.

[16] A. Kanezaki, Y. Matsushita, and Y. Nishida. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5010–5019, Jun. 2018.

[17] V. Karasev, A. Chiuso, and S. Soatto. Control recognition bounds for visual learning and exploration. In *Proc. of Information Theory and Applications Workshop (ITA)*, pages 1–8, Feb. 2013.

[18] S. Kriegel, C. Rink, T. Bodenmller, A. Narr, M. Suppa, and G. Hirzinger. Next-best-scan planning for autonomous 3d modeling. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2850–2856, Oct 2012.

[19] A. Leeper, K. Hsiao, E. Chu, and J. K. Salisbury. Using near-field stereo vision for robotic grasping in cluttered environments. *Experimental Robotics, Springer Tracts in Advanced Robotics*, pages 253–267, 2014.

[20] W. Li, X. Xiao, and J. Hahn. 3d reconstruction and texture optimization using a sparse set of rgb-d cameras. In *Proc. of IEEE Winter Conf. on Applications of Computer Vision*, pages 1413–1422, Jan 2019.

[21] M. Malmir and G. W. Cottrell. Belief tree search for active object recognition. In *Proc of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4276–4283, Sep. 2017.

[22] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928, Sep. 2015.

[23] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Proc. of IEEE Int'l Symp. on Mixed and Augmented Reality*, pages 127–136, Oct 2011.

[24] T. Patten, W. Martens, and R. Fitch. Monte Carlo planning for active object classification. *Autonomous Robots*, 42(2):391–421, 2018.

[25] C. Potthast and G. S. Sukhatme. A probabilistic framework for next best view estimation in a cluttered environment. *Journal of Visual Communication and Image Representation*, 25(1):148–164, 2014.

[26] C. R. Qi, H. Su, M. Kaichun, and L. J. Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, July 2017.

[27] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *Proc. of the International Conference on Neural Information Processing Systems (NeurIPS)*, pages 5105–5114, Dec. 2017.

[28] A. Roberti, M. Carletti, F. Setti, U. Castellani, P. Fiorini, and M. Cristani. Recognition self-awareness for active object

recognition on depth images. In *Proc. of British Machine Vision Conference (BMVC)*, page 15, Sep. 2018.

[29] A. Roberti, R. Muradore, P. Fiorini, M. Cristani, and F. Setti. An energy saving approach to active object recognition and localization. In *Proc. of Annual Conference of IEEE Industrial Electronics Society (IECON)*, pages 3153–3158, Oct. 2018.

[30] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *Proc. of International Conference on 3-D Digital Imaging and Modeling*, pages 145–152, May 2001.

[31] A. Saxena, L. Wong, M. Quigley, and A. Y. Ng. A vision-based system for grasping novel objects in cluttered environments. In M. Kaneko and Y. Nakamura, editors, *Proc. of International Symposium Robotics Research (ISRR)*, pages 337–348, 2011.

[32] R. G. Schneider and T. Tuytelaars. Sketch classification and classification-driven analysis using fisher vectors. *ACM Transactions on Graphics*, 33(6):174:1–174:9, Nov. 2014.

[33] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, Mar. 2002.

[34] Y. Shen, C. Feng, Y. Yang, and D. Tian. Neighbors do help: Deeply exploiting local structures of point clouds. *arXiv preprint arXiv:1712.06760*, 1(2), 2017.

[35] T. Siméon, J.-P. Laumond, J. Cortés, and A. Sahbani. Manipulation planning with probabilistic roadmaps. *International Journal of Robotics Research*, 23(7-8):729–746, 2004.

[36] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 945–953, Dec. 2015.

[37] L. Torabi and K. Gupta. Integrated view and path planning for an autonomous six-dof eye-in-hand object modeling system. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4516–4521, Oct. 2010.

[38] G. Walck and M. Drouin. Automatic observation for 3d reconstruction of unknown objects using visual servoing. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2727–2732, Oct. 2010.

[39] 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. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, Jun. 2019.

[40] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger. Elasticfusion: Real-time dense slam and light source estimation. *The Int'l J. of Robotics Research*, 35(14):1697–1716, 2016.

[41] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015.

[42] 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.

[43] Q.-Y. Zhou and V. Koltun. Dense scene reconstruction with points of interest. *ACM Transactions on Graphics*, 32(4):112:1–112:8, July 2013.

[44] Q.-Y. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.