UNSUPERVISED ROBOTIC SORTING: TOWARDS AUTONOMOUS DECISION MAKING ROBOTS

Joris Gu´Erin, St´Ephane Thiery, Eric Nyiri And Olivier Gibaru

Arts et M´etiers ParisTech, Lille, FRANCE

ABSTRACT

Autonomous sorting is a crucial task in industrial robotics which can be very challenging depending on the expected amount of automation. Usually, to decide where to sort an object, the system needs to solve either an instance retrieval (known object) or a supervised classification (predefined set of classes) problem. In this paper, we introduce a new decision making module, where the robotic system chooses how to sort the objects in an unsupervised way. We call this problem Unsupervised Robotic Sorting (URS) and propose an implementation on an industrial robotic system, using deep CNN feature extraction and standard clustering algorithms. We carry out extensive experiments on various standard datasets to demonstrate the efficiency of the proposed image clustering pipeline. To evaluate the robustness of our URS implementation, we also introduce a complex real world dataset containing images of objects under various background and lighting conditions. This dataset is used to fine tune the design choices (CNN and clustering algorithm) for URS. Finally, we propose a method combining our pipeline with ensemble clustering to use multiple images of each object. This redundancy of information about the objects is shown to increase the clustering results.

KEYWORDS

Autonomous robotics, Robotic sorting, Image clustering, Ensemble clustering.

1.INTRODUCTION

The problem of automatic sorting has a long history in industry, with the first tomatoes sorting system dating back to the 70's [1]. Since then, it has received a lot of attention, with important focus on combining computer vision and robotics manipulator to solve the pick-and-place task [2].

Although it was among the first robotics tasks, designing a pick-and-place application is challenging and requires solving multiple subtasks. If the objects to be sorted are cluttered, the system first needs to segment the scene and identify the different objects [3]. Then, the systems must use various sensors (2D or 3D cameras, color sensors, bar code readers, ...) to gather data about the different objects. These data are used to find a grasping strategy for the object [4] and to decide where to place the object. Naturally, all along this process, smart motion planning and control is also required [5].

Although every brick of the pick-and-place pipeline are interesting and challenging problems, the scope of this paper is limited to the decision making subproblem, i.e. how to decide where to place an object after grasping it? As discussed later in Section 2, previous implementations of automatic sorting deal with objects which are either known or belong to some predefined class. In

machine learning terminology, one could say that the robot intelligence usually resides in solving either an instance retrieval or a supervised classification
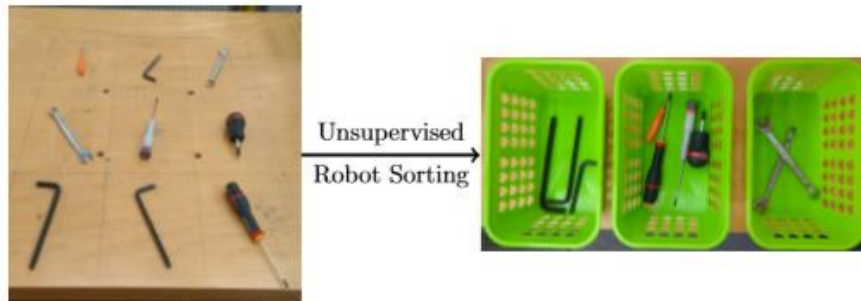

Figure 1: An instance of the unsupervised robotic sorting problem.

problem. In this paper, a new kind of pick-and-place application is introduced, consisting in sorting unknown objects, which do not belong to any predefined class. Given a set of previously unseen objects, the robotic system needs to sort them such that objects stored together are similar and different from other objects. Such problem definition is the definition of a clustering problem in which the outputs consist in physically sorted objects, hence, we call this problem Unsupervised Robotic Sorting (URS).

The above statement of the URS problem neither defines the notion of similarity nor the number of groups to be made. Different definitions of these concepts defines different instances of the URS problem. In this paper, we propose a solution to solve a URS in which the number of groups is imposed by the number of storage spaces available and the notion of similarity is defined by "how a human would solve the problem". Figure 1 shows a practical example of inputs and expected output. In practice, each object is sensed with a camera and represented as an image, the semantic content of the image is extracted using a pretrained convolutional neural network (CNN) ([6]) and the classification is done using a "standard" clustering algorithm. A complete description of the algorithm used is given in Section 3. A video of our implementation can be found at https://youtu.be/ NpZIwY3H-gE.

URS has various potential applications in industry such as organizing workspaces before the work day and sorting unsorted goods before packaging. However, to be useful in a real industrial shop floor, a URS implementation needs to work under a broad range of conditions. Object poses, background and lighting conditions are factors which vary a lot among industrial settings and they can impact widely the quality of a computer vision application. Hence, we introduce a new image dataset specifically designed to evaluate industrial robustness of our application. The dataset contains object which can be found on a shop-floor, under different orientations, background and lighting conditions. Together with artificial lighting modification, the dataset is used to choose the most robust components of the proposed clustering pipeline (CNN architecture and clustering algorithm).

It is also important to note that URS consists in sorting real objects, not only images. Hence, we propose to further improve the robustness of our implementation by introducing redundancy of information for each objects. After choosing the final elements of the image clustering pipeline, we propose to embed it in a multi-view clustering pipeline, taking as input several images, under different orientations, of each object. Leveraging ensemble clustering, we show that URS robustness can be increased using multiple views of each objects.

The main contributions of this paper consist in the formalization, together with a working implementation, of the unsupervised robotic sorting problem. Such implementation is, to the best of our knowledge, the first application of image-set clustering to robotics and constitutes a new step towards autonomous decision-making robots. To validate our solution, a new image dataset is proposed. Such dataset, which appears to be challenging for the image-set clustering task, is made publicly available in order to help other researchers to test the robustness of their image-set clustering algorithms. By implementing a multi-view clustering algorithm based on ensemble clustering, we also show that URS robustness can be improved by using multiple images of the objects.

The paper is organized as follows. In Section 2, we present a literature review on image-set clustering, multi-view clustering and decision making in robotic sorting. In Section 3, we define in details the vision pipeline proposed to solve the URS problem and validate our choice on several public datasets. In Section 4, we describe the complete URS implementation, introduce the validation dataset and test our approach on complex cases. Then we propose, in Section 5, a multi-view approach, based on ensemble clustering, to improve the robustness of URS. Finally, Section 6 discusses our findings and future working directions.

2. PREVIOUS WORK

2.1. Decision making in autonomous sorting

Although scene segmentation, robotic grasping, motion planning and control are crucial problems, which have received major attention recently, the scope of this paper is limited to the decision making subproblem of the pick-and-place pipeline. We classify the approaches to decision making into three categories: known object (instance retrieval), unknown object member of a known class (supervised classification), unknown object and no predefined class (unsupervised classification). Different approaches in solving such decision-making problems differ in the kind of sensing devices used, as well as the algorithmic choices.

Instance retrieval. Several implementations have considered the problem of recognizing each object as a member of a known database, and sorting it according to an associated predefined rule. In [7], RFID markers are used to recognize and sort objects. An implementation of a surgical tools sorting application using barcode reading and template matching to find and sort objects from a cluttered scene is proposed in [8]. Finally, the application presented in [9] sorts real-world objects (usb, glue-stick, ...) with a serial manipulator using template matching.

Supervised classification. Many robotics supervised classification applications sort objects based on simple rules (thresholding) applied to simple features. For example, [10, 11, 12] all use color features extracted from either images or color sensors, [13] extend these approaches by adding shape features. In [14], robot manipulation is used to unclutter Duplo bricks and sorts them

according to their length and color. In [15], a classical computer vision framework is implemented to carry out faulty parts removal. A robotic system able to recycle construction wastes using features provided by metal detectors and sensors sensitive to visual wavelengths is presented in [16]. Recently, researchers have started to use CNN methods to carry out supervised classification in robotic sorting. The authors of [17] have used a CNN on RGBD images to find the class of an object and sort it. In [18], a fast RCNN proposal network is combined with a pretrained VGG16 CNN architecture to jointly localize and sort objects with a robot manipulator. Unsupervised classification. In previous work, we proposed a first implementation of unsupervised robotic sorting [19]. The approach consisted in clustering a set of Lego

bricks using color and length features extracted with a computer vision pipeline. We also note that this paper is an extension of our previous work on transferring pretrained CNN features to unsupervised tasks [20], where the single-view implementation of URS was discussed.

2.2. Image-set clustering

Given a set of unlabeled images, the image-set clustering problem consists in finding subsets of images based on their content: two images representing the same object should be clustered together and separated from images representing other objects (Figure 1). This problem should not be confused with image segmentation [21], which is also sometimes called image clustering. Image-set clustering has been used to solve various problems over the last two decades. Web-scale image clustering has been tackled with variants of K-Means in both [22] and [23]. Concept discovery in image categories is solved using spectral clustering in [24]. In [25] a clustering based solution to story-line reconstruction in streams of images is proposed and in [26] image-set clustering is applied to large-scale medical image annotation.

The first successful methods focused on feature selection and used sophisticated algorithms to deal with complex features. For instance, in [27], images are represented by Gaussian Mixture Models fitted to the pixels and the set of images is clustered using the Information Bottleneck (IB) method. Features resulting from image joint segmentation and sequential IB for clustering are used in [28]. In [29], commonality measures are defined from Bags of Features with local representations (SIFT, SURF, ...) and used for agglomerative clustering. Recently, image-set clustering algorithms have started to use features extracted with deep CNN. [23] applies a variant of approximate K-means on features extracted with a pretrained AlexNet and [30] uses deep auto-encoders combined with ensemble clustering to generate feature representations suitable for clustering. As of today, a new family of methods consisting in learning jointly the clusters and the representation using alternating optimization has established a new baseline on the performance of image-set clustering algorithms ([31, 26, 32]).

Literature review of image-set clustering algorithms raises two useful remarks for the URS implementation. First, objects representations should be extracted from images using a pretrained CNN. Then, given the scale of the clustering problems to solve (#images < 100), it is unlikely that jointly optimizing the CNN parameters would improve the final sorting results. We give a detailed description of our image-set clustering pipeline as well as experimentally motivated design choices in the next section.

2.3. Multi-view clustering

Multi-view learning improves results classification results by leveraging complementary information from multiple views of a dataset, which can be obtained by various sensors or represented with different descriptors. Recently, multi-view clustering (MVC) has received a lot of attention. In [33] different loss functions are introduced and applied on the concatenated views. In both [34] and [35] lower dimensional subspaces are learned before clustering with standard methods. Recently, it has been shown in [36] that good results can be obtained by embedding MVC in the ensemble clustering framework [37]. The authors leveraged the different views to generate independent partitions and then used a co-association based method to obtain the consensus.

3. IMAGE CLUSTERING PIPELINE

In this section, we discuss the vision module of our URS application. First, the generic form of the image-set clustering pipeline is described. Then, we compare several sets of hyperparameters on the Pascal VOC 2007 test set. These preliminary results are used to prune the bad combinations and select only relevant design choices for further testings in section 4. Then, we validate the relevance of the chosen approach by comparing the best pipeline obtained with VOC against other approaches from the literature.

3.1. Pipeline description

The pipeline we propose for image set clustering is fairly straightforward. It consists in extracting deep features from all the images in the set, by using a deep convolutional neural network pretrained on a large supervised dataset for image classification and then apply a "standard" clustering algorithm to these features. The proposed vision module can be visualized later in Figure 2b.

To implement this unsupervised image classification pipeline, we first need to answer four questions:

- What dataset should be used for pretraining?
- What CNN architecture should we use?
- Which layer output should be chosen for feature extraction?
- What clustering algorithm should be used?

As of today, ImageNet [38] is the only very large labelled public dataset which has enough variety in its classes to be a good feature extractor for a variety of tasks. It has been shown than features extracted from CNN pretrained on ImageNet can be successfully transferred to a variety of tasks [6]. In addition, there are plenty of CNN pretrained on ImageNet already available online. Hence, we naturally use CNNs pretrained on ImageNet. The three other questions are answered experimentally. We use the VOC2007 [39] test set without labels to compare performances of the different options and select a small subset of parameters for further investigations on URS.

To ease development and accelerate implementation, we compare the Keras [40] implementations of ResNet50 [41], InceptionV3 [42], VGG16, VGG19 [43] and Xception [44] with the pretrained weights provided by Keras. For the clustering algorithms, we use the scikit-learn [45]

implementations of K-means (KM) [46], Minibatch K-means (MBKM) [47], Affinity Propagation (AP) [48], Mean Shift (MS) [49], Agglomerative Hierarchical Clustering (AC) [50] and DBScan (DBS) [51]. For each CNN, the layers after which the features are extracted can be found in Table 1 (Layers names are the same as in the Keras implementations).

In the image-set clustering problem, the expected classes are represented by objects present on the picture and for this reason we need semantic information, which is present in the final layers of the network. Thus, we only choose layers among the last layers of the networks. On the one hand, the last one or two layers might provide better results as their goal is to separate the data (at least for the fully-connected layers). On the other hand, the opposite intuition is also relevant as we can imagine that these layers are too specialized to be transferable. These two contradictory arguments motivated the following experiment.

We also note that the test set of VOC2007 has been modified for this validation. We removed all the images containing two or more labels in order to have ground truth to compute Normalized Mutual Information (NMI) scores. Indeed, if an image has several labels we cannot judge if the clustering pipeline prediction is correct or not. We note VOC2007-SL (single label) the modified VOC2007 test set.

3.2. Hyperparameters choice

To answer the questions stated above, we try to cluster the VOC2007-SL set using all combinations of CNN architectures, layer choices and clustering algorithms. To compare performances, we use NMI scores, a standard clustering evaluation metric. We also report clustering time for completeness. Only scikit-learn default hyperparameters of the different clustering algorithms are used, which illustrate the simplicity of this approach. For KM and MBKM, as the results depend on a random initialization, experiments are run ten times and the results reported are averaged over the different runs.

_____

[1]DB-Scan poor results might come from the default parameters and using different configurations might improve clustering, but this is out of the scope of this paper.

Table 1: NMI scores (in black) and time in seconds (in blue, italics) on Pascal VOC2007-SL test set using different CNN, different output layers and different clustering algorithms. (Layers names are the same as in the Keras implementations).

		KM	MBKM	AP	MS	AC	DBS†
Inception V3	mixed9	0.108	0.105	0.219	0.153	0.110	0
		374	*7.3*	*12.7*	*16281*	*525*	*138*
	mixed10	0.468	0.401	0.442	0.039	0.595	0
		609	*5.1*	*8.5*	*12126*	*525*	*119*
	avg_pool	0.674	0.661	0.621	0.024	0.686	0
		6.3	*0.2*	*7.7*	*230*	*8.5*	*1.8*
Resnet 50	avg_pool	0.6748	0.641	0.587	0.043	0.640	0
		7.0	*0.1*	*4.6*	*197*	*8.0*	*1.9*
VGG 16	block4_pool	0.218	0.085	0.133	0.124	0.277	0
		278	*3.6*	*6.0*	*10010*	*391*	*82.8*
	block5_pool	0.488	0.048	0.262	0.194	0.530	0
		78	*1.1*	*9.3*	*2325*	*99*	*21*
	fc1	0.606	0.458	0.421	0.187	0.617	0
		17	*0.2*	*4.8*	*365*	*17*	*3.8*
	fc2	0.661	0.611	0.551	0.085	0.673	0
		16	*0.2*	*4.3*	*373*	*15.9*	*3.8*
VGG 19	block4_pool	0.203	0.139	0.124	0.135	0.234	0
		220	*3.7*	*6.3*	*10298*	*388*	*83*
	block5_pool	0.522	0.321	0.250	0.198	0.540	0
		74	*0.9*	*9.3*	*2353*	*97*	*20*
	fc1	0.607	0.471	0.449	0.188	0.628	0
		17	*0.2*	*9.3*	*365*	*17*	*3.9*
	fc2	0.672	0.615	0.557	0.083	0.671	0
		15	*0.2*	*5.6*	*391*	*17*	*3.9*
Xception	block13_pool	0.376	0.264	0.351	0.044	0.473	0
		410	*4.8*	*10*	*9677*	*403*	*87*
	block14_act	0.574	0.428	0.584	0.071	0.634	0
		935	*10*	*10*	*24809*	*820*	*180*
	avg_pool	0.692	0.636	0.636	0.052	**0.726**	0
		7.1	*0.1*	*4.9*	*201*	*8.5*	*5.5*

These experiments show that, for every CNN architecture and clustering algorithm pairs, the results get better when we choose layers which are further in the architecture. Hence, from now on, we only consider the last layer before classification for each network. For example, in the rest of the paper, VGG16 now denotes the features extracted from fc2 in the VGG 16 architecture. Looking at the results, we also keep only two clustering algorithm (KM and AC) for our URS testings. This leaves us with 10 configurations to test in section 4.

In the next section, the relevance of the proposed pipeline is further evaluated. We compare the most successful pipeline (Agglomerative Clustering on features extracted from the final layer of an Xception CNN) against other methods from the literature on other public datasets.

3.3. Literature comparison

To justify the use of the proposed clustering routine, we compare its results against other image clustering methods on several public datasets, which are described in Table 2. The tasks involved by these datasets are different from each others (Face recognition, grouping different objects, recognizing different pictures of the same object). In addition, the content of the classes differs from the ones in ImageNet. For these reasons, the four datasets constitute a good benchmark to quantify the robustness of transfer learning for unsupervised object categorization.

Table 2: Several key features about the datasets used for the vision pipeline validation.

	Problem type	Image Size	# Classes	# Instances
COIL100 [52]	Object recognition	128×128	100	7201
Nisters [53]	Object recognition	640×480	2550	10200
ORL [54]	Face recognition	92×112	40	400
VOC2007-5-ML [2]	Object recognition	variable	5	3376

We propose a comparison with the results reported in the following papers:

• [29] proposes different clustering algorithms applied on bags of features. In Table 3, we note "BoF" the best results obtained by such pipeline on the different datasets.
• [30] proposes a method called infinite ensemble clustering (IEC). In the paper, IEC algorithm is compared to several other deep clustering algorithms and ensemble clustering algorithms. In Table 3, we report the best results obtained using Deep Clustering (DC) and Ensemble Clustering (EC) for each datasets. We note that for VOC2007-5-ML, [30] also uses deep features as clustering inputs (the CNN used is not reported).
• [31] proposes a method called Joint Unsupervised Learning (JULE) of Deep Representations and Image Clusters, based on Alternating optimization between clustering and weight optimization of the CNN feature extractor. Results from this work are reported in Table 3.

For each dataset groundtruth is known as they are intended for supervised classification. We compute both NMI scores and purity for each dataset/method pair.

Table 3: NMI scores and purity comparison on various public datasets. (A result that is not reported in the papers cited above is denoted N.R.)

NMI scores

	DC	EC	BoF	JULE	Xception + AC
COIL100	0.779	0.787	N.R.	0.985	0.951
Nisters	N.R.	N.R.	0.918	N.R.	0.991
ORL	0.777	0.805	0.878	N.R.	0.93
VOC2007-5-ML	0.265	0.272	N.R.	N.R.	0.367

Purity

	DC	EC	Xception + AC
COIL100	0.535	0.546	0.882
Nisters	N.R.	N.R.	0.988
ORL	0.578	0.630	0.875
VOC2007-5-ML	0.513	0.536	0.622

_____

[2]The data used for VOC2007 in [30] are irrelevant for clustering with hard assignment. The VOC2007 subset used in [30] contains images with several objects but only one label. However, we still

ran our clustering method on this data to be able to compare results. This second modified VOC2007 set is denoted VOC2007-5-ML (5 classes - multiple labels).

Table 3 shows that features extracted by the final layer of Xception combined with agglomerative Clustering performs better than or close to state-of-the-art methods at unsupervised object classification as well as fine-grained image clustering (ORL). Results on the ORL dataset are interesting as they show that pretrained Xception is able to classify different faces without supervision, although ImageNet does not deal with human faces at all.

We underline the very good results of JULE [31] at clustering COIL100. It appears that fine tuning feature extraction using alternating optimization is a good way of improving clustering results. However, the simple approach proposed here, which provides slightly lower results, still keeps the advantage of being very fast (as it only requires to evaluate the network once for each sample and apply AC), which is useful for our application. Moreover, the very small sample size of URS problems makes retraining impractical. From our attempts of using JULE for URS, it seems that the approach in this paper is more relevant.

It is also interesting to notice that [30] is also using CNN features for clustering VOC20075-ML. The fact that our pipeline outperform largely DC and EC for this dataset illustrate that when using transfer learning for a new problem, careful selection of the CNN used is important, which is often neglected.

4. URS IMPLEMENTATION

4.1. Physical implementation

In our implementation of URS, we use a KUKA LBR iiwa robot. Both a camera and a parallel gripper are mounted on the end-effector. First, different previously unseen objects are placed in the robot workspace. The robot scans the scene and gathers pictures of each object to be sorted. The above computer vision pipeline is applied to the image set and returns a bin number for each object. Finally, the robot physically sorts the objects according to the clustering results. Figure 2 shows a schematic view of the system, and a video of the implementation can be viewed at https://youtu.be/NpZIwY3H-gE.

The application was implemented in a shopfloor with unmastered lighting (glass roof) and a wooden table background. We tested the URS using sets of different tools (screw drivers, allen keys, flat keys, clamps, ...) and the goal was to correctly group together the different models of the same tool. For this particular task, we recorded only one misclassification in over 100 runs, which was due to very high brightness (white image).
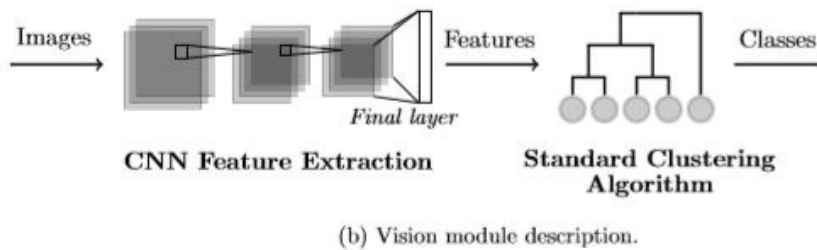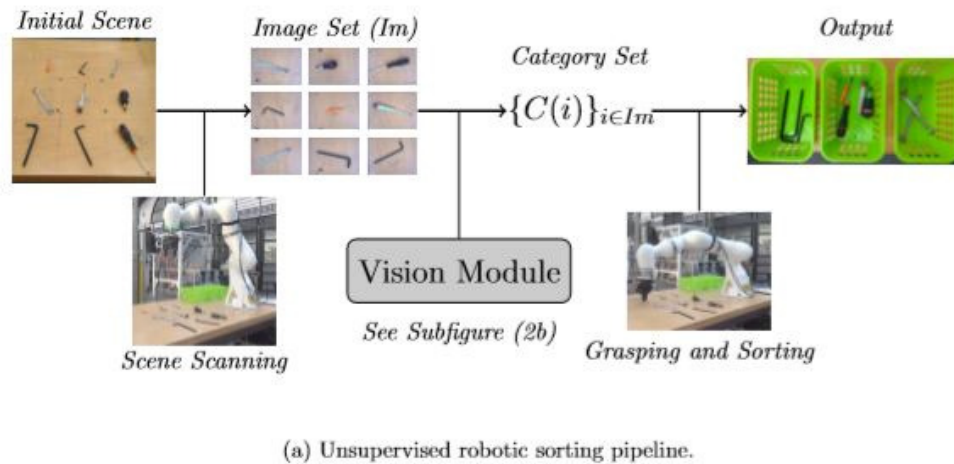
(a) Unsupervised robotic sorting pipeline.

(b) Vision module description.

Figure 2: Unsupervised robot sorting implementation.

4.2. Robustness Evaluation And fine Tuning Of The Clustering Pipeline

4.2.1. Dataset

To test the robustness of the URS application, we build a challenging dataset for image clustering, composed of pictures of objects which can be found in industrial workstations. Objects are chosen from seven classes, and pictures of each objects are taken under five different background/lighting conditions (BLC). This dataset appears to be challenging for image-set clustering because of the BLC variations but also because some classes have low intra-cluster similarity (usb) and extra-cluster similarity between some classes is relatively high (pens/screws). For each object and each BLC, the dataset contains four pictures under different position/orientation. This is used to test the robustness to objects pose variation and to test the multi-view method in Section 5.

The dataset statistics are summarized in Table 4 and sample images, illustrating the different objects and BLC, can be seen on Figure 3. The dataset, together with its description, can be downloaded at https://github.com/jorisguerin/toolClustering_dataset.

Table 4: Dataset statistics (BLC stands for Background/Lighting Conditions).

#Images	Images size	#Classes	#Images per class	#BLC
560	640 x 480	7	12 to 24	5

90

For further evaluation of the robustness to lighting conditions, we computationally modify the brightness of the pictures under conditions 2 only. Example images with the applied

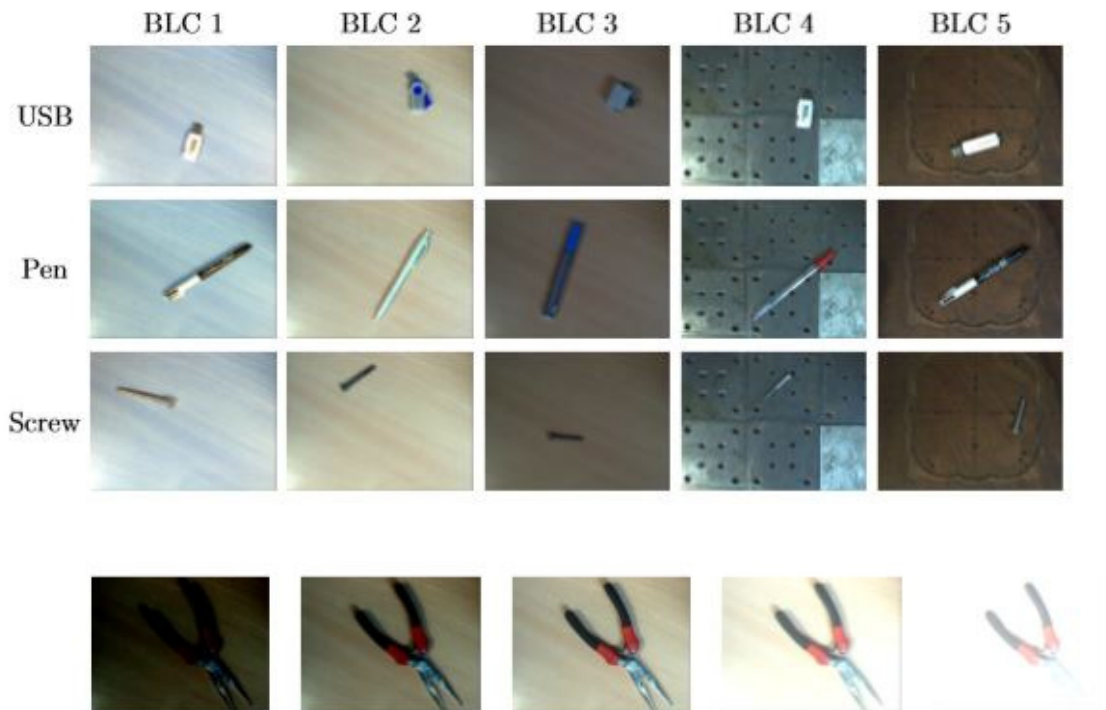Figure 3: Sample images from the robustness validation dataset.

Figure 4: Example images of the 5 artificial brightness conditions used to test
different pipelines robustness.

brightness filters can be visualized on Figure 4. This allows to isolate the influence of brightness in the clustering results.

Then, for each BLC, we create a clustering problem by sampling one image (i.e. one position/orientation) for each object. To evaluate the 10 remaining pipelines (5 CNN architectures and 2 clustering algorithms), we sample 1000 clustering problems as explained above and try to cluster them with each of the pipelines. These experiments are used to choose the most robust clustering routine for our application.

4.2.2. Results

Results from the above experiments are reported in both Tables 5 (physical BLC variation) and 6 (artificially modified brightness). Reported results are the averages over the 1000 runs. Both tables show that agglomerative clustering always present better results than KMeans. Also,

Inception-like architectures tend to outperform their competitors, particularly Xception, which outperforms all the other CNN architectures for every clustering algorithm and every BLC.

Based on these result, we choose Xception + AC as our full vision pipeline. This clustering routine is used in the next section in a framework making use of redundancy of images to improve URS robustness. The chosen pipeline seems to be robust to reasonable changes in lighting conditions (Dark and Bright) but its performances start to decrease when the brightness is really high or low. This makes sens as it also becomes difficult for a human to identify these objects. On the other hand, background changes seem to have a more important impact on the clustering results. Indeed, BLC4 shows much lower results than other BLC. This might come from the fact that the background in BLC4 contains geometric shapes (lines, circles) which are distractors for the network.

Table 5: Clustering results for different CNN architectures and clustering algorithms on the tool clustering dataset. NMI scores are in black and cluster purity in blue, italics.

		BLC1	BLC2	BLC3	BLC4	BLC5
Inception V3	AC	0.82	0.82	0.80	0.65	0.79
		0.81	*0.81*	*0.80*	*0.65*	*0.76*
	KMeans	0.80	0.79	0.76	0.63	0.75
		0.79	*0.78*	*0.76*	*0.64*	*0.73*
Resnet 50	AC	0.81	0.74	0.74	0.62	0.72
		0.81	*0.74*	*0.75*	*0.59*	*0.71*
	KMeans	0.77	0.71	0.71	0.58	0.70
		0.78	*0.72*	*0.72*	*0.58*	*0.70*
VGG 16	AC	0.76	0.74	0.73	0.61	0.70
		0.75	*0.73*	*0.72*	*0.60*	*0.69*
	KMeans	0.72	0.70	0.71	0.58	0.67
		0.72	*0.70*	*0.70*	*0.57*	*0.67*
VGG 19	AC	0.76	0.77	0.71	0.59	0.71
		0.76	*0.76*	*0.72*	*0.58*	*0.70*
	KMeans	0.73	0.73	0.69	0.56	0.67
		0.73	*0.73*	*0.70*	*0.56*	*0.67*
Xception	AC	0.86	0.90	0.84	0.69	0.83
		0.85	*0.90*	*0.85*	*0.69*	*0.81*
	KMeans	0.84	0.87	0.82	0.66	0.80
		0.83	*0.86*	*0.82*	*0.66*	*0.80*

Table 6: Clustering results for different CNN architectures and clustering algorithms for different artificially modified lighting conditions on the BLC2 subset. NMI scores are in black and cluster purity in blue, italics
.

		Very dark	Dark	Normal	Bright	Very bright
Inception V3	AC	0.74	0.81	0.82	0.80	0.71
		0.72	*0.79*	*0.81*	*0.80*	*0.70*
	KMeans	0.70	0.77	0.79	0.77	0.66
		0.70	*0.75*	*0.78*	*0.77*	*0.67*
Resnet 50	AC	0.67	0.73	0.74	0.69	0.61
		0.67	*0.73*	*0.74*	*0.68*	*0.61*
	KMeans	0.65	0.70	0.71	0.66	0.58
		0.66	*0.71*	*0.72*	*0.66*	*0.59*
VGG 16	AC	0.66	0.73	0.74	0.68	0.61
		0.66	*0.72*	*0.73*	*0.68*	*0.61*
	KMeans	0.62	0.69	0.70	0.65	0.57
		0.63	*0.69*	*0.70*	*0.66*	*0.58*
VGG 19	AC	0.67	0.76	0.77	0.74	0.64
		0.67	*0.75*	*0.76*	*0.72*	*0.65*
	KMeans	0.64	0.73	0.73	0.71	0.59
		0.65	*0.73*	*0.73*	*0.70*	*0.62*
Xception	AC	0.77	0.88	0.90	0.84	0.73
		0.77	*0.89*	*0.90*	*0.84*	*0.74*
	KMeans	0.74	0.85	0.87	0.82	0.70
		0.74	*0.86*	*0.86*	*0.82*	*0.71*

5. INCREASE ROBUSTNESS USING MULTIPLE VIEWS

In the previous sections, for each object, we only use one image. However, URS deals with physical objects and does not need to be limited to single-image clustering. Using multiple views can increase the performance of the application by using redundant and complementary information. Hence, we propose a multi-view clustering method to solve the URS problem more accurately.

5.1. Multi-view clustering using Ensemble-clustering

Let $\Omega = \{\Omega_1,...\Omega_n\}$ be the set of n objects to cluster. For each object Ωi, let $\omega_i = \{\omega_{i,1},...\omega_{i},m_i\}$ be the set of m_i views representing it. And let C denote the clustering pipeline chosen from the previous experiments (Xception + AC). The proposed MultiView Ensemble Clustering (MVEC) method, inspired by [36], goes as follows. For each object Ωi, we randomly select one image $\omega_{i,j}$. Then, we cluster the new image set using C, the cluster assignments (partition) generated, is denoted Pk. This procedure is repeated N times and we note $P = \{P_1,...P_N\}$ the set of N partitions. Once the N partitions have been generated by randomly sampling images in each ωi, MVEC must find a consensus partition P∗, maximizing the agreement between the different partitions. In this paper, we choose an approach based on objects co-occurrence to compute P∗. Since we deal with unsupervised classification, there is no correspondence between the class assignments of the different partitions of P. To avoid this problem, we use an intermediate representation of P, called the co-association matrix: CA. CA is a n×n matrix which entries are defined by

$$CA_{pq} = \frac{1}{N}\sum_{t=1}^{N} \delta(P_t(\Omega_p), P_t(\Omega_q)),$$

(1)

where $P_t(\Omega i)$ is the label associated with object Ω_i and generated by partition Pt, and $\delta(a,b)$ is the Kronecker symbol, which is 1 if a = b and 0 otherwise. Entry (i,j) of CA measures how many times objects Ω_i and $_{\Omega j}$ have been classified together by the different partitions. Finally, let C∗ be any connectivity-based clustering algorithm. For instance, C∗ can be variants of agglomerative clustering or spectral clustering. The consensus partition is obtain by applying C∗ using CA as the precomputed similarity measure between objects.

$$P^* = C^*(CA). \tag{2}$$

A schematic representation for the MVEC pipeline can be found in Figure 5. In our implementation, image sampling for partition generation is uniform, C∗ is agglomerative clustering and the number of partitions generated is N = 1000.

5.2. Results

The results using MVEC with the clustering pipeline chosen above are reported in Table 7. MVEC consistently performs significantly better than the corresponding single view approach (written in parenthesis in Table 7 as a reminder). It is more robust to poor background (BLC4) and lighting (Very bright) conditions.

The excellent results obtained by MVEC were expectable. Indeed, by increasing the number of views, we introduce complementary information available for clustering. MVEC
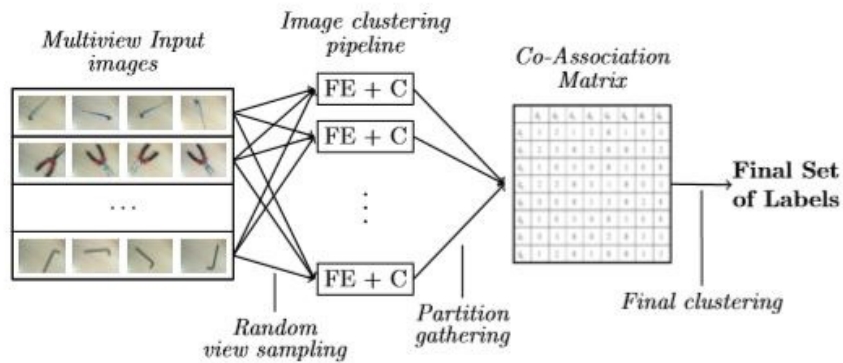

Figure 5: Proposed MVEC approach to use multiple views of each object.

is naturally more robust to the unlucky event of selecting a poor view. Robustness to poor lighting conditions also makes sense as light comes from a certain direction and there are always better angles to observe the objects. Moreover, the same object, observed from different angles, can look very different, hence, by increasing the number of views, we also make more likely the event of having similar images for similar objects.

Regarding computation time, although MVEC takes longer, this time can be drastically reduced by parallelizing both the partition generation process and the co-association matrix computation.

Table 7: Clustering results of MVEC for different BLC. For comparison, we remind the corresponding results using single views, in parenthesis.

		NMI	Purity
BLC1		0.95 *(0.86)*	0.96 *(0.85)*
BLC2	Very dark	0.91 *(0.77)*	0.93 *(0.77)*
	Dark	1.00 *(0.88)*	1.00 *(0.89)*
	Normal	1.00 *(0.90)*	1.00 *(0.90)*
	Bright	0.96 *(0.84)*	0.96 *(0.84)*
	Very bright	0.84 *(0.73)*	0.86 *(0.74)*
BLC3		0.95 *(0.84)*	0.96 *(0.85)*
BLC4		0.84 *(0.69)*	0.82 *(0.69)*
BLC5		0.95 *(0.83)*	0.96 *(0.81)*

6. CONCLUSION

6.1. Conclusive remarks

In this paper, we introduced a new kind of robotics pick-and-place application, called Unsupervised Robotic Sorting, which consists in physically grouping together previously unseen objects in a way that makes sense at a human level. We propose an implementation in which objects are represented by images and the unsupervised classification module is composed of a pretrained Xception feature extractor and Agglomerative Clustering. This pipeline was shown to be good through experiments on several public datasets, as well as a specifically created robustness testing dataset. This approach is shown to work in a practical implementation, in a real-world industrial environment.

The robustness testing dataset, which is challenging for image-set clustering, is made publicly available to help other researchers to test their image clustering algorithms. We also show, through further experimentation on this dataset, that using multiple images for each object can increase URS performance. By embedding the chosen image clustering pipeline into a co-association ensemble clustering framework, we obtain more accurate and more robust clustering results.

6.2. Future working directions

The proposed approach to solve URS demonstrates good results and is very encouraging. However, we have also shown that URS would benefit from using multiple views. Future work will consist in implementing such multi-view URS on a real robot. To do so, we first need to solve another problem of view selection. We have already started to investigate the question of autonomous optimal view selection [55], by training a neural network to predict the camera pose which is best suited for clustering. Further developments will consist in combining this approach with the multi-view framework proposed in this paper.
In this paper, the focus is placed on the decision making module of the autonomous sorting application. For this reason, the scene segmentation, data gathering and grasping were simple fixed modules. Now that the intelligent unsupervised sorting module has been shown to work, it

will be interesting to include it in a more complete pick-and-place pipeline, which handles automatic scene segmentation and robotic grasp detection.

7. REFERENCES

[1] R. G. Husome, R. J. Fleming, and R. E. Swanson, "Color sorting system," Dec. 26 1978. US Patent 4,131,540.

[2] C. C. Mouli and K. N. Raju, "A review on wireless embedded system for vision guided robot arm for object sorting," International Journal of Scientific & Engineering Research, July 2013.

[3] A. Ecins, C. Fermüller, and Y. Aloimonos, "Cluttered scene segmentation using the symmetry constraint," in Robotics and Automation (ICRA), 2016 IEEE International Conference on, pp. 2271–2278, IEEE, 2016.

[4] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesisa survey," IEEE Transactions on Robotics, vol. 30, no. 2, pp. 289–309, 2014.

[5] B. Siciliano and O. Khatib, Springer handbook of robotics. Springer, 2016.

[6] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-theshelf: an astounding baseline for recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp. 806–813, 2014.

[7] N. Giannoccaro, L. Spedicato, and A. Lay-Ekuakille, "A smart robotic arm for automatic sorting of objects with different tags," in 4th Imeko TC19 Symposium on Environmental Instrumentation and Measurements Protecting Environment, Climate Changes and Pollution Control, June, pp. 3–4, 2013.

[8] H. Tan, Y. Xu, Y. Mao, X. Tong, W. B. Griffin, B. Kannan, and L. A. DeRose, "An integrated vision-based robotic manipulation system for sorting surgical tools," in Technologies for Practical Robot Applications (TePRA), 2015 IEEE International Conference on, pp. 1–6, IEEE, 2015.

[9] M. Dragusu, A. N. Mihalache, and R. Solea, "Practical applications for robotic arms using image processing," in System Theory, Control and Computing (ICSTCC), 201216th International Conference on, pp. 1–6, IEEE, 2012.

[10] R. Szabo and I. Lie, "Automated colored object sorting application for robotic arms," in Electronics and Telecommunications (ISETC), 2012 10th International Symposium on, pp. 95–98, IEEE, 2012.

[11] A. Shum, Y. Wang, and S.-J. Hsieh, "Design, build and remote control of a miniature automated robotic sorting system," Design, Build, vol. 141, no. 3, 2016.

[12] M. Nkomo and M. Collier, "A color-sorting scara robotic arm," in Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on, pp. 763–768, IEEE, 2012.

[13] V. Pereira, V. A. Fernandes, and J. Sequeira, "Low cost object sorting robotic arm using raspberry pi," in Global Humanitarian Technology Conference-South Asia Satellite (GHTC-SAS), 2014 IEEE, pp. 1–6, IEEE, 2014.

[14] M. Gupta and G. S. Sukhatme, "Using manipulation primitives for brick sorting in clutter," in Robotics and Automation (ICRA), 2012 IEEE International Conference on, pp. 3883–3889, IEEE, 2012.

[15] T. Singh, D. Dhaytadak, P. Kadam, and R. Sapkal, "Object sorting by robotic arm using image processing," International Research Journal of Engineering and Technology (IRJET), 2016.

[16] T. J. Lukka, T. Tossavainen, J. V. Kujala, and T. Raiko, "Zenrobotics recycler– robotic sorting using machine learning," in Proceedings of the International Conference on Sensor-Based Sorting (SBS), 2014.

[17] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard, "Multimodal deep learning for robust rgb-d object recognition," in Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, pp. 681–687, IEEE, 2015.

[18] C. Zhihong, Z. Hebin, W. Yanbo, L. Binyan, and L. Yu, "A vision-based robotic grasping system using deep learning for garbage sorting," in Control Conference (CCC), 2017 36th Chinese, pp. 11223–11226, IEEE, 2017.

[19] J. Gu´erin, O. Gibaru, S. Thiery, and E. Nyiri, "Clustering for different scales of measurement-the gap-ratio weighted k-means algorithm," in Fourth International Conference on Artificial Intelligence and Applications (AIAP), 2017.

[20] J. Gu´erin, O. Gibaru, S. Thiery, and E. Nyiri, "Cnn features are also great at unsupervised classification," 4th International Conference on Artificial Intelligence and Applications (AIFU 2018), 2018.

[21] N. Dhanachandra and Y. J. Chanu, "A survey on image segmentation methods using clustering techniques," European Journal of Engineering Research and Science, vol. 2, no. 1, 2017.

[22] Y. Gong, M. Pawlowski, F. Yang, L. Brandy, L. Bourdev, and R. Fergus, "Web scale photo hash clustering on a single machine," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 19–27, 2015.

[23] Y. Avrithis, Y. Kalantidis, E. Anagnostopoulos, and I. Z. Emiris, "Web-scale image clustering revisited," in Proceedings of the IEEE International Conference on Computer Vision, pp. 1502–1510, 2015.

[24] Y. J. Lee and K. Grauman, "Shape discovery from unlabeled image collections," in Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pp. 2254–2261, IEEE, 2009.

[25] G. Kim, L. Sigal, and E. P. Xing, "Joint summarization of large-scale collections of web images and videos for storyline reconstruction," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4225–4232, 2014.

[26] X. Wang, L. Lu, H.-c. Shin, L. Kim, M. Bagheri, I. Nogues, J. Yao, and R. M. Summers, "Unsupervised joint mining of deep features and image labels for large-scale radiology image categorization and scene recognition," arXiv preprint arXiv:1701.06599, 2017.

[27] J. Goldberger, S. Gordon, and H. Greenspan, "Unsupervised image-set clustering using an information theoretic framework," IEEE transactions on image processing, vol. 15, no. 2, pp. 449–458, 2006.

[28] Y. Seldin, S. Starik, and M. Werman, "Unsupervised clustering of images using their joint segmentation," in Proceedings of the 3rd International Workshop on Statistical and Computational Theories of Vision (SCTV 2003), 2003.

[29] T. Fukui and T. Wada, "Commonality preserving image-set clustering based on diverse density," in International Symposium on Visual Computing, pp. 258–269, Springer, 2014.

[30] H. Liu, M. Shao, S. Li, and Y. Fu, "Infinite ensemble for image clustering," in Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016.

[31] J. Yang, D. Parikh, and D. Batra, "Joint unsupervised learning of deep representations and image clusters," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5147–5156, 2016.

[32] F. Li, H. Qiao, B. Zhang, and X. Xi, "Discriminatively boosted image clustering with fully convolutional auto-encoders," arXiv preprint arXiv:1703.07980, 2017.

[33] A. Kumar and H. Daum´e, "A co-training approach for multi-view spectral clustering," in Proceedings of the 28th International Conference on Machine Learning (ICML-11), pp. 393–400, 2011.

[34] H. Zhao, Z. Ding, and Y. Fu, "Multi-view clustering via deep matrix factorization.," in AAAI, pp. 2921–2927, 2017.

[35] Y. Wang, W. Zhang, L. Wu, X. Lin, M. Fang, and S. Pan, "Iterative views agreement: An iterative low-rank based structured optimization method to multi-view spectral clustering," arXiv preprint arXiv:1608.05560, 2016.

[36] Z. Tao, H. Liu, S. Li, Z. Ding, and Y. Fu, "From ensemble clustering to multi-view clustering," in Proc. of the Twenty-Sixth Int. Joint Conf. on Artificial Intelligence (IJCAI), pp. 2843–2849, 2017.

[37] S. Vega-Pons and J. Ruiz-Shulcloper, "A survey of clustering ensemble algorithms," International Journal of Pattern Recognition and Artificial Intelligence, vol. 25, no. 03, pp. 337–372, 2011.

[38] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., "Imagenet large scale visual recognition challenge," International Journal of Computer Vision, vol. 115, no. 3, pp. 211–252, 2015.

[39] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," International journal of computer vision, vol. 88, no. 2, pp. 303–338, 2010.

[40] F. Chollet, "Keras," 2015.

[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778, 2016.

[42] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826, 2016.

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

[44] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," arXiv preprint arXiv:1610.02357, 2016.

[45] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in ECML PKDD Workshop: Languages for Data Mining and Machine Learning, pp. 108–122, 2013.

[46] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, pp. 1027–1035, Society for Industrial and Applied Mathematics, 2007.

[47] D. Sculley, "Web-scale k-means clustering," in Proceedings of the 19th international conference on World wide web, pp. 1177–1178, ACM, 2010.

[48] D. Dueck and B. J. Frey, "Non-metric affinity propagation for unsupervised image categorization," in Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, pp. 1–8, IEEE, 2007.

[49] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," IEEE Transactions on pattern analysis and machine intelligence, vol. 24, no. 5, pp. 603–619, 2002.

[50] F. Murtagh, "A survey of recent advances in hierarchical clustering algorithms," The Computer Journal, vol. 26, no. 4, pp. 354–359, 1983.

[51] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al., "A density-based algorithm for discovering clusters in large spatial databases with noise.," in Kdd, vol. 96, pp. 226– 231, 1996.

[52] S. Nayar, S. Nene, and H. Murase, "Columbia object image library (coil 100)," Department of Comp. Science, Columbia University, Tech. Rep. CUCS-006-96, 1996.

[53] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in Computer vision and pattern recognition, 2006 IEEE computer society conference on, vol. 2, pp. 2161–2168, Ieee, 2006.

[54] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on, pp. 138–142, IEEE, 1994.

[55] J. Gu´erin, O. Gibaru, E. Nyiri, S. Thiery, and B. Boots, "Semantically meaningful view selection.." Submitted to International Conference on Robot Intelligence (IROS) 2018.