Comparison of Action-Grounded and Non-Action-Grounded 3-D Shape Features for Object Affordance Classification*

Barry Ridge[1], Emre Ugur[2], and Aleš Ude[1]

***Abstract*—Recent work in robotics, particularly in the domains of object manipulation and affordance learning, has seen the development of *action-grounded features*, that is, object features that are defined dynamically with respect to manipulation actions. Rather than using pose-invariant features, as is often the case with object recognition, such features are grounded with respect to the manipulation of the object, for instance, by using shape features that describe the surface of an object relative to the push contact point and direction. In this paper we provide an experimental comparison between action-grounded features and non-grounded features in an object affordance classification setting. Using an experimental platform that gathers 3-D data from the Kinect RGB-D sensor, as well as push action trajectories from an electromagnetic tracking system, we provide experimental results that demonstrate the effectiveness of this action-grounded approach across a range of state-of-the-art classifiers.**

I. INTRODUCTION

In the field of autonomous robotics, vision and control are often treated as distinct domains in which useful data and action commands are derived, processed, or manipulated separately to achieve a broader goal. In the context of robotic affordance learning [1], [2], this has traditionally been the standard approach, and simplifying assumptions are often made in order to make targeted problems more soluble. For example, in the case of object push affordance learning [3]–[7], if the desired result is to learn how the positions and orientations of objects change when pushed, the learning task can be simplified by selecting prior object models, using standard computer vision techniques to localise the object models within a scene, and inferring data such as end effector contact points on the objects using the models. However, when fewer assumptions are made about the shapes of objects, the types of push actions that might be performed, and the resulting affordances, such techniques may not be as feasible. On the other hand, the formation of dynamic object models, through interactive experience, that are semantically associated with actions at a more intrinsic level, is a promising research area in that it offers the potential to enhance developmental robotic learning and, ultimately, robotic autonomy.

In this paper, we explore the use of action-grounded 3-D object shape features in an object push affordance classification setting. Using an RGB-D sensor to gather 3-D point cloud data of objects, and using an electromagnetic tracking system to gather push trajectory data, objects on a table surface were pushed by a human experimenter (cf. Fig. 1) whose hand motion trajectories were tracked while 3-D point clouds of the objects were recorded. The objects were pushed from various different positions on their surfaces and from various different directions, exhibiting a number of different affordances such as forward translations, forward topples, left rotations and right rotations, depending broadly on the shapes of the objects, their orientations, and how they were pushed. Our chief point of investigation was to determine whether the proposed action-grounded 3-D shape feature approach to dynamic object affordance modeling in this setting offers any improvement over more standard methods.

A. Related Work

E. Gibson discusses that learning affordances refers to "narrowing down from a vast manifold of (perceptual) information to the minimal, optimal information that specifies the affordance of an event, object, or layout"[8]. Selection and use of the action-relevant features have already been studied and shown to be effective in learning and representing affordances. For example Montesano et al. [4] learned affordances only using the object properties that are found to be relevant to the corresponding actions, for more effective predictions and planning. Hart and Grupen also discussed the necessity of selecting the relevant features in order to capture the salient information while learning affordances [9]. Lately, it was also shown that use of relevant features in affordance learning can lead to the discovery of hierarchical structures in predicting interdependent affordances of different complexities[10]. While all these studies discuss how to select the relevant features from a general purpose feature set, we aim to find a transformed feature set based on the actions in consideration.

Implicit encoding of object manipulation information in object shape feature descriptors has been applied in the grasping field, often via the use of haptic or tactile sensors. Recently, Björkman et al. [11] employed tactile measurements from the haptic finger sensors on a robotic hand to enhance prior visual object models via implicit object surface modeling. Meier et al. [12] developed a probabilistic spatial approach for building compact 3-D representations of unknown objects probed by tactile sensors using Kalman filters to build a probabilistic model of the contact point cloud.

Object shape features from 3-D vision have been used in prior work on push affordance learning [6], [7], [13], [14], but grounding such features relative to pushing actions has not been studied as extensively. Recent work by Hermans et al. [15], [16] used shape features encoded in a reference frame defined by object centres and push locations based on 2-D projections of object point clouds. Krainin et al. [17] developed an approach to building 3-D models of unknown objects for grasping based on a depth camera observing a

*This research has been supported by EU FP7 project Xperience (ICT-270273).

[1]B. Ridge and A. Ude are with the Laboratory of Humanoid and Cognitive Robotics, Department of Automation, Biocybernetics and Robotics, Jožef Stefan Institute, Ljubljana, Slovenia. barry.ridge at ijs.si, ales.ude at ijs.si

[2]E. Ugur is with the Intelligent and Interactive Systems Laboratory, University of Innsbruck, Austria. emre.ugur at uibk.ac.at

978-1-4673-7509-2/15/$31.00 2015 European Union

robotic hand while moving an object and modeling the object surface dynamically using sets of small surface patches.

In our previous work [18], we developed a similar idea, employing full 3-D shape features grounded with respect 3-D action trajectories to perform bootstrap discovery and prediction of object affordance classes using a multi-view self-supervised learning algorithm. In this work, by contrast, we focus on examining action-grounded 3-D shape features in more detail and investigating whether or not they provide an improvement over features that are not implicitly defined with respect to the manipulation.

The remainder of the paper is structured as follows. In the following section, we give a brief overview of our experimental platform including the object point cloud segmentation process. In Section III we describe how the action-grounded features and non-grounded features are derived, including the object segmentation process, the definitions of both the action-grounded and non-action-grounded feature reference frames, and the descriptions of the features themselves. In Section IV we describe our experiments with various state-of-the-art classifiers and results. Finally, in Section V, we conclude and discuss potential future work.

II. EXPERIMENTAL SETUP

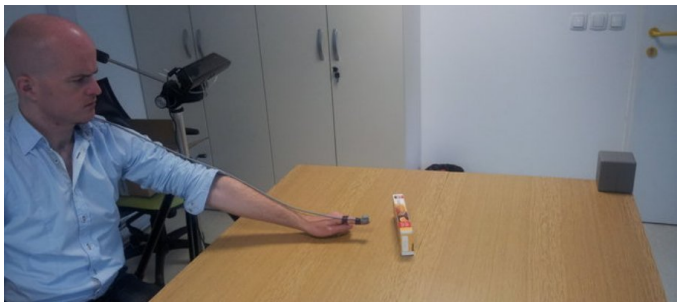


Fig. 1. Our setup for human object push affordance data gathering.

In our experimental setup for human object push data gathering, shown in Figure 1, we employed a Microsoft Kinect™ RGB-D sensor for gathering 3-D point cloud data of scenes and objects, and a Polhemus Patriot™ electromagnetic tracking system for gathering trajectory data of human hand motions. A wooden table with a wooden frame was used as the work surface in order to avoid electromagnetic interference from metallic objects in the environment. A tracking sensor was placed at the end of the index finger of a human experimenter, while the tracking source was located at a corner of the table with the Kinect facing the table at a 45° angle as shown in Figure 1. Objects were placed at arbitrary locations on the table surface where they were pushed from various directions and at various contact points by the experimenter. 3-D point clouds of the scene were recorded both before and after each push interaction while hand trajectories were tracked during the interaction. Both the point clouds and the trajectories were processed offline where the objects were segmented from the table surface, object point clouds and push trajectories were transformed into the push action reference frame, and action-grounded shape features were extracted.

We used tools from the Point Cloud Library (PCL)[1] to perform dominant plane segmentation on scene point clouds in order to acquire segmented point clouds of the objects lying on the table surface. This involved using a pass-through filter to subtract points in the scene cloud outside certain range limits, using RANSAC [19] to fit a plane model to the scene cloud, subtracting those scene points that were plane inliers, and clustering the remaining points to find the objects using Euclidean clustering [20].

III. ACTION-GROUNDED VS. NON-GROUNDED 3-D SHAPE FEATURES

In this section, we first discuss how the reference frames in our definitions of action-grounded and non-action-grounded features differ, before proceeding to describe the features themselves.

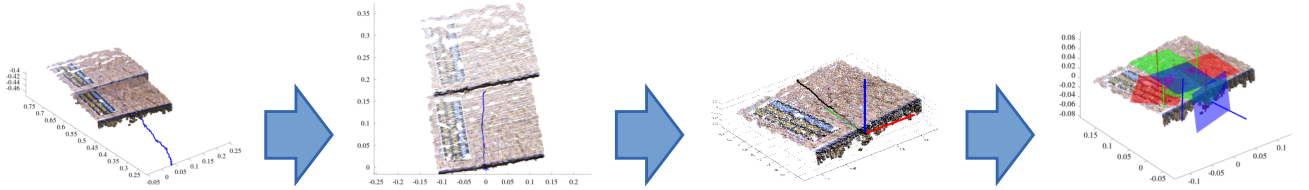
A. Action-grounded reference frame

We define the action-grounded frame to be the reference frame with its origin at the contact point on the object, its positive y-axis pointing in the direction of the pushing motion parallel to the table surface, its positive z-axis pointing upward from the table surface, and its positive x-axis pointing to the right of the object. In order to transform both the object point cloud and the push trajectory into the action-grounded frame, we perform the following procedure. Firstly, we transform the push trajectory from the Patriot tracker reference frame to the Kinect reference frame by using least-squares adjustment on a series of control points and calculating a rigid body transformation of the form $\mathbf{x}' = \mathbf{c} + \mathbf{R}\mathbf{x}$, where $\mathbf{x}'$ is the transformed vector, $\mathbf{x}$ is the initial vector, $\mathbf{c}$ is the translation vector, and $\mathbf{R}$ is a rotation matrix. The control points are gathered prior to performing pushing experiments by placing the tracking sensor at various positions in the workspace, recording the sensor position, recording the Kinect point cloud of the scene, then locating the sensor in the point cloud. Since the pushing motions performed in our experiments always follow an approximately linear trajectory, we proceed by using orthogonal distance regression via singular value decomposition to fit a 3-D line to the push trajectory. Finally, we find the point of intersection between this fitted line and the pre-push object point cloud, infer this to be the contact point, and finally transform the pre-push object point cloud as well as the push trajectory to the action frame as defined by the contact point and the fitted line. This process is visualized in Figure 2.

B. Non-action-grounded reference frame

In the case of the non-action-grounded features, we fit a bounding box to the object point cloud, found the centroid of the bounding box, and used that as the origin for our non-grounded reference frame. The axis orientations were found by transforming the object point cloud with respect to the Kinect reference frame such that the positive y-axis pointed from the origin of the Kinect frame towards the object, parallel to the table surface; the positive z-axis pointed upward from the table surface; and the positive x-axis pointed to the right of the object. This is illustrated for a sample object point cloud in Figure 3.

[1] http://pointclouds.org



(a) Object point clouds before/after interaction + hand push trajectory

(b) Point clouds + trajectory transformed to push action reference frame.

(c) Pre-push point cloud action reference frame grounding.

(d) Fitting planes to pre-push object cloud parts, grounded with respect to action.

Fig. 2. Action-grounded shape feature extraction pipeline.

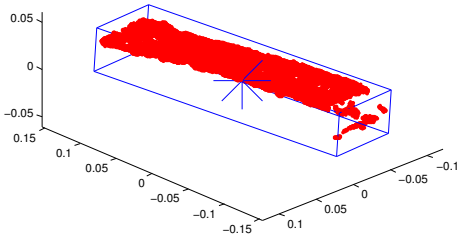


Fig. 3. Sample object point cloud with bounding box and centroid for definition of non-action-grounded reference frame.

C. Action-grounded 3-D shape features

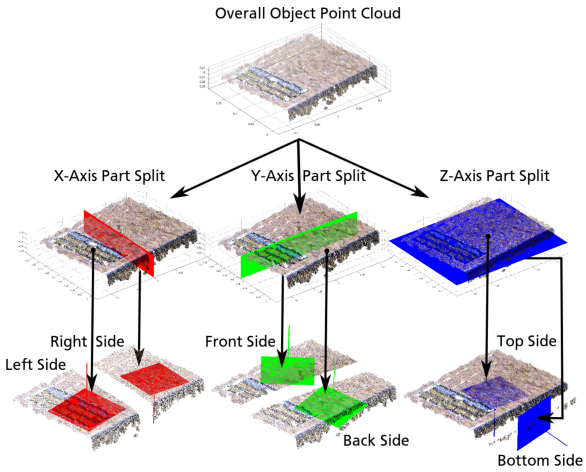


Fig. 4. Partitioning a sample object point cloud into sub-parts. Top row: original pre-push object point cloud. Middle row: partitioning planes divide the point cloud evenly in each dimension to create sub-parts. Bottom row: planes are fitted to each sub-part for feature extraction.

With the pre-push object point cloud now grounded in the action reference frame, we turn to generating a feature descriptor that describes the shapes of the object point clouds with respect to the pushing action and that is rich enough to capture the resulting affordance effects. The main idea behind our approach is to divide the object point clouds into cells of sub-parts and use the properties of the sub-parts of the point clouds as a basis for the feature descriptor. More concretely, we divide each object point cloud evenly with respect to its minimum and maximum points along each coordinate axis such that there are seven cells that overlap for redundancy: one for the overall point cloud, two for the x-axis, two for the y-axis, and two for the z-axis. We then use two types of feature descriptors in each cell. To gauge the position of the sub-part in each cell relative to the action frame, we find the centroid of the points in the cell, which gives us three features. To gauge the shape of the sub-part in each cell relative to the action frame, we fit a planar surface to the points within the cell and use the two largest coordinates of the plane normal as features. Examples of these features being extracted from different point clouds are shown in Figure 7.

Using these five types of features, three for relative part position and two for planar surface fit orientation, we extract the five features for each part. This results in the following list of 35 features that are extracted:

$O^a_{1\ldots5}$: Global point cloud centroid/plane features.
$O^a_{6\ldots10}$: x-split, left side centroid/plane features.
$O^a_{11\ldots15}$: x-split, right side centroid/plane features.
$O^a_{16\ldots20}$: y-split, front side centroid/plane features.
$O^a_{21\ldots25}$: y-split, back side centroid/plane features.
$O^a_{26\ldots30}$: z-split, top side centroid/plane features.
$O^a_{31\ldots35}$: z-split, bottom side centroid/plane features.

D. Non-grounded 3-D shape features

For the non-grounded feature set, we used the same methodolgy in terms of the object parts division as described in the previous sub-section, and we extracted the same features as before, but crucially, the reference frame was not defined with respect to the push action, as with the action-grounded features, but was defined with respect to the object centroid. Thus, we have:

$O^b_{1\ldots35}$: non-grounded analogues of features $O^a_{1\ldots35}$.

Since such an object centroid-oriented approach is a standard means of deriving features in many computer vision tasks, such as object recognition and classification, this seemed like the most reasonable methodology for performing a direct comparison between the action-grounded features and similar non-action-grounded features that might have been employed in their absence.

E. Action features from push trajectories

In order to make it feasible for the classifiers to be able to predict the affordances of the objects using non-action-grounded shape features, it is necessary to include information about the action in the feature descriptor in some form. We approached this by augmenting the non-grounded 3-D features with features derived from the object contact point and the push action trajectory. These were as follows:

$O^b_{36...38}$: x, y, z-coordinates of object contact point.
$O^b_{39,40}$: x, y-parameters of push trajectory line fit.

The z-parameter of the line that was fit to the push trajectory was deemed redundant since all pushes took place in the same plane, i.e. along the table surface. Combining these 5 features with the 35 features of the previous sub-section yielded a 40-dimensional feature vector for the non-grounded feature set.

Fig. 5. Test objects used in our experiments.

IV. EXPERIMENTS

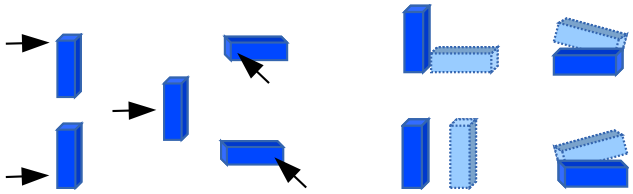


Fig. 6. The 5 different push types (left) and 4 affordance classes (right).

The experimental environment was set up as shown in Figure 1. When objects with arbitrarily complex geometries, or indeed sometimes objects with simple geometries like balls [21], are used for affordance learning experiments, the resulting affordances can be quite difficult to hand-label for a human. Thus, in the absence of unsupervised or self-supervised affordance class discovery [6], [22], which was not employed in this work, regularly shaped objects were used in order to allow for reliable human ground-truth labeling. We selected 5 household objects (cf. Fig. 5) for the experiments: 4 flat-surfaced objects; a book, a marshmallow box, a cookie packet, and a biscuit box, and 1 curved-surfaced object; a yoghurt bottle. A dataset was collected as follows, where object push tests were performed on each of the 5 objects and the resulting data was processed, leaving 120 data samples. Objects were placed at random start locations and in various poses within the workspace and within view of the Kinect sensor, and the human experimenter would perform straight-line pushes on the objects, attempting to keep the pushes within reasonable limits of 5 different push categories: pushing through the top, bottom, left, right and centre of the objects respectively, from the direction of the field of view of the Kinect. Table I contains a matrix detailing the numbers of samples collected for each of the objects in each different possible pose. Table II details the number of different affordances produced by each of the objects during the interactions. Certain objects prohibited certain poses and certain affordances. For example, neither the cookie pack nor the book could be placed in sideways or upright poses and thus did not produce instances of the toppling affordance.

TABLE I. OBJECT/POSE MATRIX

	Flat	Sideways	Upright	Total
Cookie Pack	18	0	0	18
Mallow Box	9	15	9	33
Biscuit Box	9	15	9	33
Book	18	0	0	18
Yoghurt Bottle	9	0	9	18
Total	63	30	27	120

TABLE II. OBJECT/AFFORDANCE MATRIX

	Topple	Translate	Left Rot.	Right Rot.	Total
Cookie Pack	0	6	6	6	18
Mallow Box	12	9	6	6	33
Biscuit Box	12	9	6	6	33
Book	0	6	6	6	18
Yoghurt Bottle	6	6	3	3	18
Total	30	36	27	27	120

The data collection process resulted, by inspection, in 4 different affordance categories being produced, and the samples were hand-labelled with four ground truth labels to reflect this: *left rotation*, *right rotation*, *forward translation* and *forward topple*. These various push types and resulting affordances categories illustrated conceptually in Figure 6 are sample object interactions are shown in Fig. 7.

A. Evaluation Procedure

In order to compare action-grounded and non-action-grounded features, the dataset described in the previous section was used to set up a classification task where a given classifier was trained separately with training sets derived from a portion of the dataset using both action-grounded features and non-action-grounded, and the learning objective was to predict the affordance ground truth labels from test sets derived from the remaining data. For the evaluation, we used 10-fold cross-validation with multiple different state-of-the-art classifiers. Two learning vector quantization (LVQ)-based algorithms [23], *generalized relevance learning vector quantization (GRLVQ)* [24] and *supervised relevance neural gas (SRNG)* [25] were used, alongside *multinomial logistic regression (MLR)*, *support vector machines (SVM)* and *random forests (RF)* [26]. The cross-validation was performed in ten separate trials where the samples in the dataset were randomized and divided into ten folds, and the results were averaged over the trials in order to account for random prototype initialization and other variations between runs in the various algorithms.

In the cases of both the GRLVQ and SRNG models[2], following the notation used in [25], the main learning rate parameter ϵ^+ was set to 0.2, while ϵ^- was set to 0.04, and ϵ

[2]http://www.mathworks.com/matlabcentral/fileexchange/17415-neural-network-classifiers

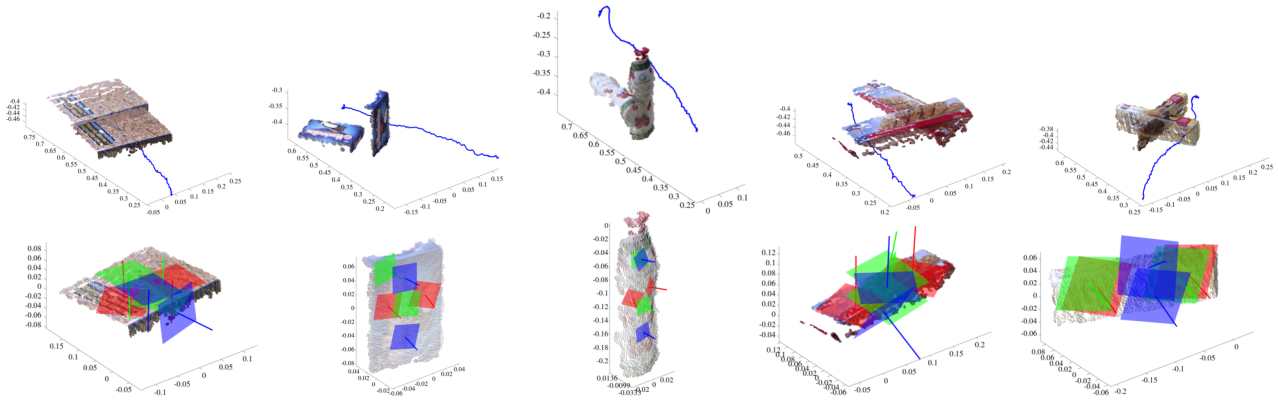


Fig. 7. Action-grounded shape feature extraction. Top row: pre-push and post-push 3-D point clouds and action trajectories for the five test objects being pushed in various different ways. Bottom row: action-grounded shape feature extraction (cf. Section III-C) for the pre-push point clouds. Plane fits are shown in red for the x-axis divisions of the point clouds, in green for the y-axis divisions, and in blue for the z-axis divisions. Four different affordances are visible in the columns from left to right: forward translation, forward topple, right rotation, and left rotation.

for the λ feature relevance update was set to 0.02. A set of 60 prototype vectors were used in each case, divided into 15 prototypes per class. Training ran for 5000 epochs over the training data in each case to ensure convergence. The logistic regression model used a multinomial logit link function and a confidence interval of 95%. In the case of the SVM model[3], parameters were optimized using cross validation over the training data prior to training. The random forests model[4] used 500 trees.

TABLE III. MEAN 10-FOLD CROSS-VALIDATION PREDICTION RESULTS: TRAINING SETS

	Non-Grounded	Action-Grounded
GRLVQ	95.1± 3%	97.1± 2%
SRNG	95.4± 3%	97.8± 2%
SVM	97.1± 2%	98.6± 1%
MLR	100.0± 0%	100.0± 0%
RF	100.0± 0%	100.0± 0%

TABLE IV. MEAN 10-FOLD CROSS-VALIDATION PREDICTION RESULTS: TEST SETS

	Non-Grounded	Action-Grounded
GRLVQ	87.8± 11%	92.2± 9%
SRNG	88.6± 11%	93.0± 8%
SVM	83.6± 10%	86.1± 10%
MLR	74.3± 12%	77.2± 11%
RF	84.8± 11%	95.7± 6%

B. Results

The discussion of the results is divided into two subsections. In the first of these, we analyse the performance of each of the classifiers on both the non-grounded and action-grounded datasets in an effort to evaluate whether action-grounding can result in classification performance gains. In the second, we look at how action-grounding affects the relevances of individual features by exploiting the feature relevance determination mechanism provided by the SRNG algorithm.

[3]http://www.csie.ntu.edu.tw/~cjlin/libsvm

[4]https://code.google.com/p/randomforest-matlab

TABLE V. MEAN 10-FOLD CROSS-VALIDATION CONFUSION MATRIX: NON-ACTION-GROUNDED FEATURES, SRNG

		Prediction			
		Topple	Translate	Left Rot.	Right Rot.
Truth	Topple	80.0%	20.0%	0%	0%
	Translate	13.9%	86.1%	0%	0%
	Left Rot.	0%	0%	100.0%	0%
	Right Rot.	0%	0%	0%	100.0%

TABLE VI. MEAN 10-FOLD CROSS-VALIDATION CONFUSION MATRIX: ACTION-GROUNDED FEATURES, SRNG

		Prediction			
		Topple	Translate	Left Rot.	Right Rot.
Truth	Topple	76.7%	23.3%	0%	0%
	Translate	5.6%	94.4%	0%	0%
	Left Rot.	0%	0%	100.0%	0%
	Right Rot.	0%	0%	0%	100.0%

1) Classifier Performance: The main classifier performance results are shown in Tables III and IV, where average cross-validation classification matching accuracies are shown for the 9-fold training sets and single-fold test sets respectively. GRLVQ and SRNG, the two learning vector quantization methods, perform robustly in all cases and benefit from action-grounding in both training and test cases. An improvement in the results for these two methods is also evident here compared to the results presented in [18] on the same dataset, which is due to the substantial increase in training epochs used in this paper (5000 vs. 100) to ensure convergence for better comparison with the other methods. Tables V and VI show confusion matrices for test set results for non-action-grounded features and action-grounded features respectively, where the results were summed over the ten folds in each trial and averaged over all ten trials. SRNG tends to confuse topples for translations and translations for topples in both cases, though when action-grounded features are used, its error rate for translations is reduced by almost 10%.

Interestingly, while the SVM also benefits from action-grounding, it does not perform quite as well as the LVQ-based methods, which are likely deriving a comparative benefit from their in-built feature relevance determination mechanisms which the SVM, in the implementation used here, does not

TABLE VII. MEAN 10-FOLD CROSS-VALIDATION CONFUSION MATRIX: NON-ACTION-GROUNDED FEATURES, RF

		Prediction			
		Topple	Translate	Left Rot.	Right Rot.
Truth	Topple	83.3%	16.7%	0%	0%
	Translate	22.2%	69.4%	2.8%	5.6%
	Left Rot.	7.4%	0%	92.6%	0%
	Right Rot.	3.7%	3.7%	0%	92.6%

TABLE VIII. MEAN 10-FOLD CROSS-VALIDATION CONFUSION MATRIX: ACTION-GROUNDED FEATURES, RF

		Prediction			
		Topple	Translate	Left Rot.	Right Rot.
Truth	Topple	96.7%	3.3%	0%	0%
	Translate	8.3%	91.7%	0%	0%
	Left Rot.	0%	0%	100.0%	0%
	Right Rot.	0%	0%	0%	100.0%

share. The MLR model, given its high performance on training data and relatively poor performance on test data in both the non-grounded and action-grounded cases clearly suffers here from overfitting. It could be the case that the high-dimensionality of the datasets causes it difficulty and that it would benefit from dimensionality reduction or regularization, though neither have yet been tested.

The random forests model, on the other hand, performs much more stably between training and testing sets and appears to benefit even more considerably than the other classifiers from the action-grounding of features, offering a $\sim 10\%$ performance increase in said case. This becomes even more clear when analysing the confusion matrices for random forests in Tables V and VI respectively. When using non-grounded features, the model tended to predict topples or rotations in some $\sim 30\%$ of cases when it should have been predicting translations. These errors were reduced significantly ($\sim 8\%$) when action-grounded features were used.

The action-grounded feature set, therefore, appears to induce superior performance over the non-grounded feature set in all classifier comparisons across both training and test sets in this experiment, which is encouraging, though as is discussed later in Section V, more extensive testing would be necessary before more general conclusions can be drawn.

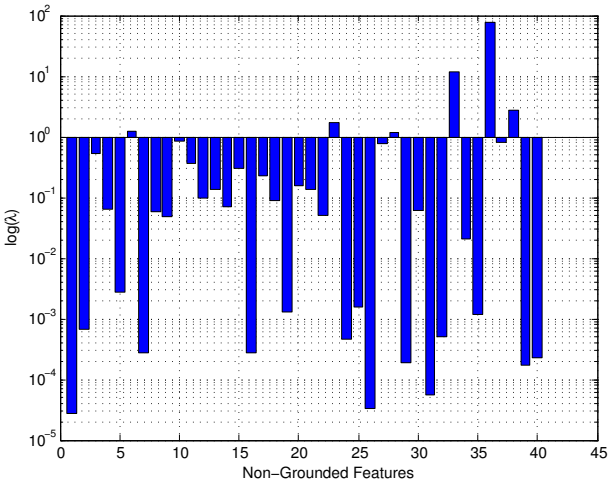


Fig. 8. SRNG feature relevance results for the non-grounded features dataset. Results are shown with a log scale.

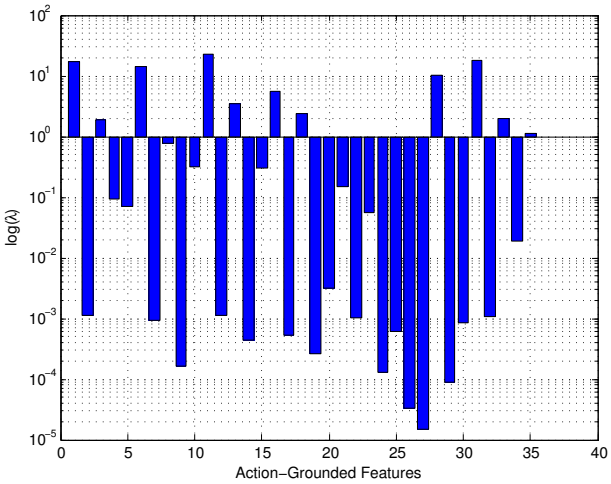


Fig. 9. SRNG feature relevance results for the action-grounded features dataset. Results are shown with a log scale.

2) Feature Relevance: Figures 8 and 9 show average feature relevance bar charts as derived from the SRNG classifier over the ten trials of 10-fold cross-validation for the non-grounded and action-grounded feature sets respectively. Perhaps the most prominent result here is the more distributed spread of different features that show significance in the case of the action-grounded dataset. Of the non-grounded features shown in Fig. 8, two of the push action features, those being the x and z coordinates of the object contact point respectively, as well as the z-coordinate of the z-axis bottom side part from the 3-D features, show significant relevance. This matches with intuition as to what might make good predictors given the affordance classes involved in this experiment, since these features, crucially, encode both push height and planar direction, necessary for distinguishing topples vs. translations and left vs. right rotations respectively.

It seems evident from these results therefore that, in the case of these experiments at least, most of the relevant information for affordance prediction was provided by the push features and not the shape features themselves. It is worth noting also that SRNG outperforms all the other classifiers in test cases in the non-grounded features experiment (cf. Table IV), which may be because it is able to single out these important features owing to its powerful in-built feature relevance determination mechanism. By contrast, in the case of the action-grounded features experiment, the action-grounding spreads this information out via intrinsic encoding across the feature set, thus allowing the other classifiers lacking such a feature relevance determining component to also benefit from it.

V. CONCLUSIONS AND FUTURE WORK

To conclude, in this paper, we have presented an experimental comparison between action-grounded and non-action-grounded features derived from 3-D point clouds of objects. The experimental results demonstrated that action-grounded features can be effective for scenarios like object affordance learning by showing increased performance over similar non-grounded features across a range of state-of-the-art classifiers.

With regard to future work, we would like to explore the possibilities of using different point cloud divisions, parts-based structures and potentially part-hierarchies, beyond what has been presented here with bipartite axis splits. It would also be interesting to investigate the use of different shape features within the parts and sub-parts. From the affordance learning perspective, we also aim to implement regression capabilities that would allow for continuous prediction of object and object part positions.

Although the results presented here are encouraging, it is also difficult to draw broader conclusions about the generalisation capabilities of action-grounded features from this one study alone. In particular, it would be both interesting and important to expand this work to more complex scenarios where the shape features, as opposed to the push features, play a more significant predictive role. Therefore, we hope to expand on this study in future work with more objects, affordances, and action types, as well as implementation on a humanoid robot platform.

REFERENCES

[1] J. J. Gibson. *The Ecological Approach to Visual Perception*. Houghton Mifflin, 1979.

[2] E. Sahin, M. Cakmak, M. R. Dogar, E. Ugur, and G. Ucoluk. To afford or not to afford: A new formalization of affordances toward affordance-based robot control. *Adaptive Behavior*, 15(4):447, 2007.

[3] P. Fitzpatrick, G. Metta, L. Natale, S. Rao, and G. Sandini. Learning about objects through action-initial steps towards artificial cognition. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, 2003.

[4] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor. Learning object affordances: From sensory-motor coordination to imitation. *IEEE Transactions on Robotics*, 24(1):15–26, 2008.

[5] D. Omrčen, C. Boge, T. Asfour, A. Ude, and R. Dillmann. Autonomous acquisition of pushing actions to support object grasping with a humanoid robot. In *Proceedings of the 9th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 277 –283, December 2009.

[6] B. Ridge, D. Skočaj, and A. Leonardis. Self-supervised cross-modal online learning of basic object affordances for developmental robotic systems. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5047–5054, Anchorage, USA, May 2010. IEEE.

[7] M. Kopicki, S. Zurek, R. Stolkin, T. Morwald, and J. Wyatt. Learning to predict how rigid objects behave under simple manipulation. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5722 –5729, May 2011.

[8] E. J. Gibson. The World Is So Full of a Number of Things: On Specification and Perceptual Learning. *Ecological Psychology*, 15(4):283–288, 2003.

[9] S. Hart and R. Grupen. Intrinsically motivated affordance discovery and modeling. In *Intrinsically Motivated Learning in Natural and ArtificialSystems*, pages 279–300. Springer, 2013.

[10] E. Ugur and J. Piater. Emergent Structuring of Interdependent Affordance Learning Tasks. In *IEEE Intl. Conf. on Development and Learning and on Epigenetic Robotics*. 2014.

[11] M. Bjorkman, Y. Bekiroglu, V. Hogman, and D. Kragic. Enhancing visual perception of shape through tactile glances. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3180–3186, November 2013.

[12] M. Meier, M. Schopfer, R. Haschke, and H. Ritter. A Probabilistic Approach to Tactile Shape Reconstruction. *IEEE Transactions on Robotics*, 27(3):630–635, June 2011.

[13] E. Ugur, E. Sahin, and E. Oztop. Self-discovery of motor primitives and learning grasp affordances. In *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3260 –3267, October 2012.

[14] E. Ugur, E. Oztop, and E. Sahin. Goal emulation and planning in perceptual space using learned affordances. *Robotics and Autonomous Systems*, 59(7–8):580–595, July 2011.

[15] T. Hermans, F. Li, J. M. Rehg, and A. F. Bobick. Learning Contact Locations for Pushing and Orienting Unknown Objects. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robotics (Humanoids)*, 2013.

[16] T. Hermans, F. Li, J. M. Rehg, and A. F. Bobick. Learning Stable Pushing Locations. In *Proceedings of the Third Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics (ICDL-EpiRob)*, Osaka, Japan, August 2013.

[17] M. Krainin, P. Henry, X. Ren, and D. Fox. Manipulator and object tracking for in-hand 3d object modeling. *The International Journal of Robotics Research*, 30(11):1311–1327, September 2011.

[18] B. Ridge and A. Ude. Action-grounded push affordance bootstrapping of unknown objects. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2791–2798, November 2013.

[19] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[20] R. B. Rusu. *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*. PhD thesis, Computer Science department, Technische Universitaet Muenchen, Germany, October 2009.

[21] B. Ridge, A. Leonardis, A. Ude, M. Deniša, and D. Skočaj. Self-Supervised Online Learning of Basic Object Push Affordances. *International Journal of Advanced Robotic Systems*, page 1, 2015.

[22] E. Ugur, E. Sahin, and E. Oztop. Unsupervised learning of object affordances for planning in a mobile manipulation platform. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4312–4317. IEEE, May 2011.

[23] T. Kohonen. *Self-organizing maps*. Springer, 1997.

[24] B. Hammer and T. Villmann. Generalized relevance learning vector quantization. *Neural Networks*, 15(8-9):1059–1068, 2002.

[25] B. Hammer, M. Strickert, and T. Villmann. Supervised neural gas with general similarity measure. *Neural Processing Letters*, 21(1):21–44, 2005.

[26] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.