SHAPE-AWARE SPATIO-TEMPORAL DESCRIPTORS FOR INTERACTION CLASSIFICATION

Sören Pirk[1] *Olga Diamanti*[1] *Boris Thibert*[1,2] *Danfei Xu*[1] *Leonidas Guibas*[1]

[1]Stanford University, USA [2]Université Grenoble Alpes, France

ABSTRACT

Many real-world tasks for autonomous agents benefit from understanding dynamic inter-object interactions. Detecting, analyzing and differentiating between the various ways that an object can be interacted with provides implicit information about its function. This can help train autonomous agents to handle objects and understand unknown scenes. We describe a general mathematical framework to analyze and classify interactions, defined as dynamic motions performed by an active object onto a passive one. We factorize interactions via motion features computed in the spatio-temporal domain, and encoded into a global, object-centric signature. Equipped with a similarity measure to compare such signatures, we showcase classification of interactions with a single object. We also propose a novel acquisition setup combining RGBD sensing with a virtual reality (VR) display, to capture interactions with purely virtual objects.

Index Terms— interactions, classification, object affordances, scene analysis, motion descriptor

1. INTRODUCTION

Understanding the functionality of objects is essential for exploring unknown scenes. An autonomous robotic agent attempting to navigate and operate in an unknown, unstructured environment could benefit from reasoning about the function of the various present objects. Since an exhaustive list of functionalities for all possible object categories is infeasible, the agent should ideally be able to deduce their function independently, for example through interaction [1, 2, 3]. A typical interaction example [4, 5] is grasping - much research has focused on learning its nuances [6] to enable motion planning of automatic grasps in high-uncertainty unstructured environments [7, 8]. The affordance of an object, namely the ways in which it can be handled and interacted with, can be crucial for understanding its functionality. Thus, building a knowledge base for detecting possible interactions becomes an important task in order to improve autonomous robotic operation [9].

Other than autonomous navigation, learning the set of possible interactions with objects and how to detect them can be an important tool for building smart environments that respond to human intent [10, 11]. A particular interaction is correlated to the state of the object and what the human intends to do - for example, one might grasp a bottle from the side to pour liquid into a glass, or grasp it from the top to unscrew a screw cap. Building tools to distinguish the different possible interactions performed on an object shape can thus facilitate determining human intent [12].

Regardless of the application, a first step in creating such a knowledge base of object affordances is to build tools for capture and classification of the allowed interactions [13]. This is similar to establishing an encyclopedia of possible interactions with classes of objects by detecting, understanding and organizing the possible interactions for each object. Based on such a database, intelligence capabilities could then be built to apply this knowledge to new objects: by interacting with a new object and matching the interactions to the database, the agent may be able to better decode its environment. We focus on the first step: identifying and labeling different physical interactions that can be performed on an object.

Current interaction representations can be either too abstract, providing high-level descriptions (e.g. human language) of the involved motions, or too detailed, requiring thorough simulation of the underlying processes (e.g. for motion control of robots) [14, 15]. In contrast, we define an intermediate representation for close-range interactions that abstracts the underlying complexity while providing nuanced details. We provide an object-centric way to distinguish between the various affordable interactions. Since object shape largely determines the set of possible interactions, we leverage shape information: a coffee mug with a handle can be grasped from the side or the top, or it can be held by the handle. We focus not only on spatial [16], but also temporal aspects of real interactions, which we model as motions caused by a moving object (e.g. a hand), exerted on the static object, e.g. a cup, and captured through standard RGBD sensors. Our current dataset consists only of hand-based interactions, detectable via dedicated hand-tracking solutions [17, 18, 19]. However we envision accommodating much more variable interactions in the future (e.g. sitting on chairs, opening doors) and therefore also showcase generic RGBD tracking techniques. To help with common occlusion problems [20], we couple RGBD sensing with a virtual reality (VR) feedback loop into a novel capture setup, and observe hand motions performed on *virtual* objects, extracted e.g. from repositories [21, 22]. Finally, we design a novel object-centered interaction descriptor and a corresponding interaction similarity measure for classification experiments.

2. METHOD

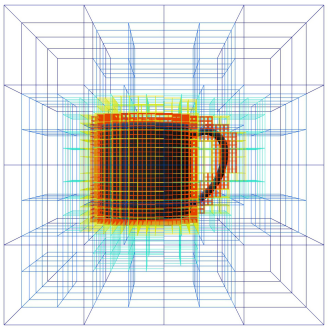
Our input is a set of interactions, each comprising a static (S) and a moving (M) object. We capture the motion of M during the interaction is captured as moving 3D points in the space around S .

2.1. The interaction descriptor

From the input (moving point cloud), we design an object-centric descriptor of each interaction, which considers both the 3D point motion and the object shape. The descriptor is computed via a set of virtual *"sensors"* (or *"cells"*), built in the 3D space around the object. Each sensor captures the motion of the moving object M inside a particular region of surrounding 3D space.

Sensor setup and layout. To create the layout for our virtual sensors, we place an axis-aligned 3D grid around the static object S, large and enough to cover the entire shape and its sur-

rounding space, in which the relevant motion of the interaction occurs. Instead of a fixed-resolution regular grid, we use an adaptive oct-tree structure with more grid cells closer to the object (inset), in order to capture the motion flow in more detail there - this renders the sensor layout shape-aware. Each grid cell is associated with a static virtual sensor that measures the interaction inside that cell; the sensors are independent (don't intersect). The adaptive oct-tree structure of the grid imposes a natural "layering" on the sensors: each sensor C_i^l is assigned to a single layer $l \in \{1, \dots L\}$ depending on the level of the oct-tree at which it lies. Sensors at $l = L$ are on the leaf nodes of the oct-tree, lie closest to the object and are the smallest in size; sensors at $l = 1$ are larger, further away from the object, and "near" the root in the oct-tree. Each layer contains similarly sized sensors roughly equidistant to the object.



Sensor functionality and output. Each sensor is designed to capture the part of the interaction that occurs inside its designated grid cell. We assume that a sequence of moving 3D point clouds has been obtained for the interaction via data acquisition. At every time t_κ of the sequence, each sensor C_i (layer index omitted for clarity) counts the number of points Φ_i^κ of the point cloud that lie inside its associated 3D cell. The output of the sensor is a time-sequence ("*flux*") $\Phi_i := \{(t_\kappa, \Phi_i^\kappa)\}_{1 \le \kappa \le n}$. The number of time steps n is fixed for all sensors for a given interaction. For a given interaction, we call a particular sensor *active*, if it has captured a signal for the interaction, i.e. if the time-sequence above has recorded a non-negligible amount (≥ 10) of moving 3D points during the interaction.

The descriptor. The interaction descriptor is a concatenation of (a) the list of active sensors at each layer l and (b) for each active sensor i, a $n \times 1$ vector Φ_i measuring the number of 3D points inside the sensor at each time.

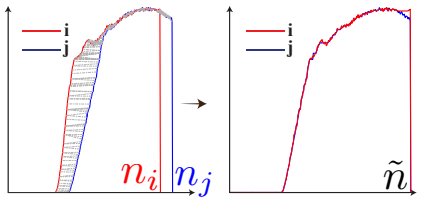
2.2. Similarity measure between interaction descriptors

Given two interactions $\mathcal{I}_1$, $\mathcal{I}_2$ on the same object, comparing the output of the sensor layouts gives a notion of difference between the motion flows of the two interactions. For this, we design a similarity measure to compare them by comparing the active sensor outputs. The similarity measure only compares sensors of the same layer to each other: for each layer l, we calculate the similarity measure $\mathcal{D}^l(\mathcal{I}_1, \mathcal{I}_2)$ between the captured outputs of that layer for $\mathcal{I}_1$, $\mathcal{I}_2$. Given the adaptive sensor layout, this allows for more resolution closer to the object. This is useful in various hand grasp scenarios, where the final finger configuration in the grasp is only attained when the hand reaches the object. We then add the per-layer distances to form the global interaction similarity measure.

Similarity cost between sensors. We compare two identical layers l (with possibly different active sensors) for two interactions by computing a similarity cost between any *two* active sensors of that layer. The most straightforward pairwise cost simply measures the sensors' proximity in 3D space - in theory, having two close-by sensors in space being active for both interactions is a strong indicator that the interactions are similar. However, this type of cost measure is not rotation-invariant: if an object is grasped twice in the same way but from a different direction, far-away sensors in the same layer might fire up.

To overcome this limitation, we compare two sensors by considering their time-sequence outputs: two sensors are similar if they captured similar signals (numbers of 3D points passing through) over time. Directly comparing the two time-sequences frame-per-frame is not possible: two interactions may have different time durations, and a sensor's response can be stretched/squeezed in time depending on the speed at which an interaction is performed. Such variability in interaction speed is typically unavoidable in datasets where more than one different users are handling an object.

Instead, we first apply a step of dynamic-time warping-DTW [23] to the two sensor outputs. DTW aligns these in time, accounts for different average speeds, and random accelerations and decelerations in the subject's motion. Given two time-sequences $\Phi_i := \{(t_\kappa, \Phi_i^\kappa)\}_{1 \le \kappa \le n_i})$ and $\Phi_j := \{(t_\lambda, \Phi_j^\lambda)\}_{1 \le \lambda \le n_j})$, for two sensors C_i, C_j, DTW optimally stretches/squeezes them in time onto a common set of time instants of the same duration $\tilde{n}$: Φ_i is mapped to $\tilde{\Phi}_i := \{(\tilde{t}_\tau, \tilde{\Phi}_i^\tau)\}_{1 \le \tau \le \tilde{n}}$, Φ_j is mapped to $\tilde{\Phi}_j := \{(\tilde{t}_\tau, \tilde{\Phi}_j^\tau)\}_{1 \le \tau \le \tilde{n}}$ (inset) and the pairwise similarity cost is given by $c_{\mathrm{DTW}}(C_i, C_j) = \sum_{\tau=1}^{\tilde{n}} \left(\tilde{\Phi}_i^\tau - \tilde{\Phi}_j^\tau\right)^2$.



We found that normalizing the two time sequences by dividing by their respective maximums $M_i = \max_\kappa \Phi_i^\kappa$, $M_j = \max_\lambda \Phi_j^\lambda$ prior to DTW is beneficial. We account for differences in magnitude by scaling the cost $c_{\mathrm{DTW}}(C_i, C_j)$ by $\frac{\max(M_i, M_j)}{\min(M_i, M_j)}$ after DTW.

Optimal transport between two sets of sensors. Recall that we can associate to each interaction L disjoint sets of sensors, corresponding to the various layers. Let $l \in \{1, \cdots, L\}$ be the index of such a layer. For our distance metric between two interactions $\mathcal{I}_1$ and $\mathcal{I}_2$, we will compare the sensors belonging to the same layer. Let I_1^l and I_2^l denote the sets of sensors belonging to the l^{th}-layer that were active for the two interactions; also, let N_1^l and N_2^l denote the cardinality of those sets (the number of active sensors in l for $\mathcal{I}_1$ and $\mathcal{I}_2$). Given our pairwise sensor similarity cost c_{DTW}, we define the per-layer similarity cost between the sets I_1^l and I_2^l by solving an optimal transport problem [24].

We first define the probability measures μ_1^l and μ_2^l on respectively I_1^l and I_2^l. For a sensor C_i at layer l, let $A_1(C_i) := \sum_\kappa \Phi_i^\kappa|_{\mathcal{I}_1}$ denote the total number of 3D points that passed through the sensor throughout the video sequence for interaction $\mathcal{I}_1$. Let also $A_1^l = \sum_{i \in I_1^l} A_1(C_i)$ denote the sum of these accumulated numbers for all the sensors of layer l. Then we define μ_1 by $\mu_1^l(C_i) = A_1(C_i)/A_1^l$ for every $C_i \in I_1^l$ and represent it as a column vector with N_1^l entries. Similarly for interaction $\mathcal{I}_2$, one defines $\mu_2^l(C_j) = A_2(C_j)/A_2^l$ for every $C_j \in I_2^l$. We then denote by $\Sigma(\mu_1^l, \mu_2^l)$ the set of transport plans between μ_1^l and μ_2^l. Each element $\pi \in \Sigma(\mu_1^l, \mu_2^l)$ is an $N_1^l \times N_2^l$ matrix with positive entries that satisfies $\pi \mathbb{1}_{N_2^l} = \mu_1^l$, $\pi^T \mathbb{1}_{N_1^l} = \mu_2^l$, where $\mathbb{1}_N$ is a column vector with N entries equal to 1.

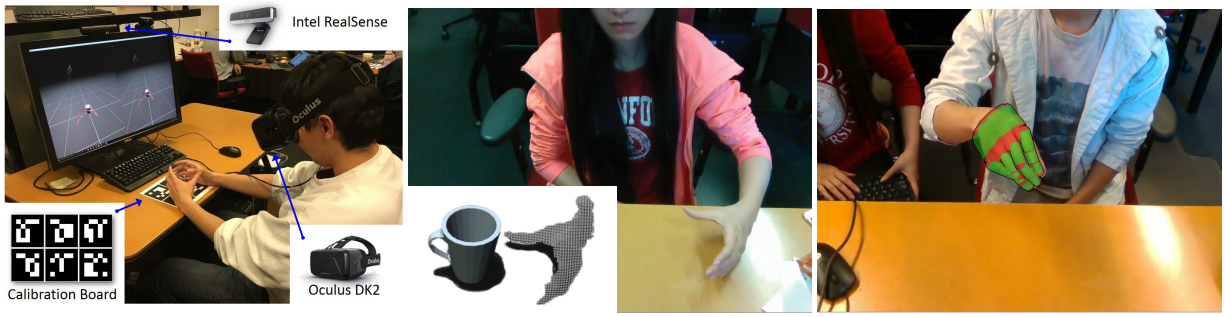


Fig. 1. Left - a novel capturing setup: RGBD camera coupled with a VR headset, which enables interaction with virtual objects. Middle and right- modalities for capturing the moving point clouds: generic RGBD scene flow, and hand-tracking.

With the above definitions, we define the total similarity cost $\mathcal{D}^l(\mathcal{I}_1, \mathcal{I}_2)$ between two equivalent layers l for the two interactions as the optimal transportation cost between the probability measures μ_1^l and μ_2^l for the cost c_{DTW}:

$$\mathcal{D}^l(\mathcal{I}_1, \mathcal{I}_2) := \min_{\pi \in \Sigma(\mu_1^l, \mu_2^l)} \sum_{i \in I_1^l, j \in I_2^l} \pi_{i,j} c_{\mathrm{DTW}}(C_i, C_j). \quad (1)$$

Note that the value $\mathcal{D}^l(\mathcal{I}_1, \mathcal{I}_2)$ is always larger than the minimal cost between any two sensors and lower than the maximal cost between any two sensors. In practice, we compute an approximation of this cost based on an entropic regularization [25]. If $I_1^l = I_2^l = \emptyset$, then we define the cost as being zero. If only one set is empty, we set $\mathcal{D}^l(\mathcal{I}_1, \mathcal{I}_2) = 1$.

Global similarity measure. We define the global similarity measure between interactions $\mathcal{I}_1$ and $\mathcal{I}_2$ as the weighted sum of the per-layer optimal transportation costs over the L pairs of sensor layers

$$\mathcal{D}(\mathcal{I}_1, \mathcal{I}_2) := \sum_{l=1}^{L} w_l \mathcal{D}^l(\mathcal{I}_1, \mathcal{I}_2)/W, \quad W = \sum_{l=1}^{L} w_l \quad (2)$$

The per-layer weights w_l are designed so that pairs of sensor layers with large total numbers of accumulated 3D points contribute more. This accounts for the difference in size between the sensors at different layers - the outermost sensors are fewer, but larger, hence they accumulate more points on average. Additionally, the weights penalize pairs of layers whose total number of accumulated points differ significantly. Namely

$$w_l = \left(1 + \frac{|A_1^l - A_2^l|}{\max(A_1^l, A_2^l)}\right) \frac{A_1^l + A_2^l}{A_1 + A_2}, \quad (3)$$

The coefficient $(A_1^l + A_2^l)/(A_1 + A_2)$ gives an importance proportional to the accumulated values of the pairs of subsets. The other term penalizes the asymmetry of the accumulated values. Note that this formula handles the case when one set I_2^l is empty (hence $A_2^l = 0$) as well as when both are empty.

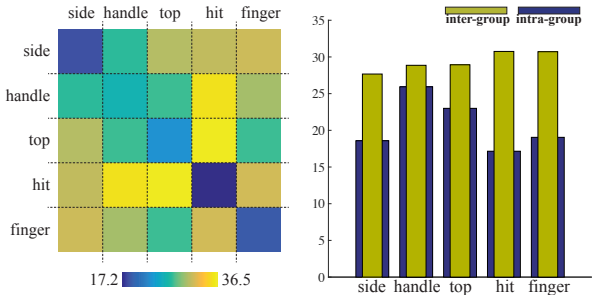


Fig. 2. Classification accuracy, with scene-flow [26] input.

3. EXPERIMENTS

3.1. Data capture and extraction of moving point clouds

Our novel data capturing setup consists of an RGBD camera (Intel RealSense RS300, 48Hz) and a head-mounted display (HMD) device (Oculus DK2, 30Hz) (Fig. 1). The depth sensor captures the human subject's motion as point clouds and is calibrated to match the rendering space of the HMD [27][28]. In real-time, the captured point clouds are rendered into the HMD, so that the subject perceives their own motion virtually from a first-person perspective. The 3D model of the target object is then placed in the same virtual space and displayed in the HMD. The subject interacts with the virtual object using the point cloud rendering of their body parts through the HMD as the only visual feedback. Moving 3D interaction point clouds are extracted from raw RGBD data via two different modalities (Fig. 1).

Scene Flow. For every two consecutive RGBD video frames, we utilize [26] to convert the input per-pixel RGBD data into a dense point cloud and its associated per-point motion field (referred to as *scene flow* in the computer vision literature). This yields a subset of the raw sensor point cloud data that can be reliably tracked between frames. We discard the point velocities and only use the detected points for generating our interaction descriptor. We prune any points that are far away ($\geq$ 30cm) from the object.

Hand-Tracking. The point cloud data acquired via [26] contains noise and outliers, due to RGBD capturing artifacts. Additionally, our single sensor setup only captures frontal views of the moving hand, often yielding incomplete point clouds. A dedicated hand tracker [18] yields a full tracked 3D mesh of a human hand, which produces cleaner 3D point trajectories. Naturally, this modality is only relevant if the moving objects are hands, and thus less appropriate than [26] for our overarching goal of accommodating more general types of interactions. However, it is still valuable as a more accurate testing baseline (although not entirely error-free in the presence of rapidly-changing hand-poses).

3.2. Results and Discussion

To evaluate our pipeline, we used our descriptors and interaction similarity measure to performed interaction classification. We captured a total of five different interactions performed onto the same

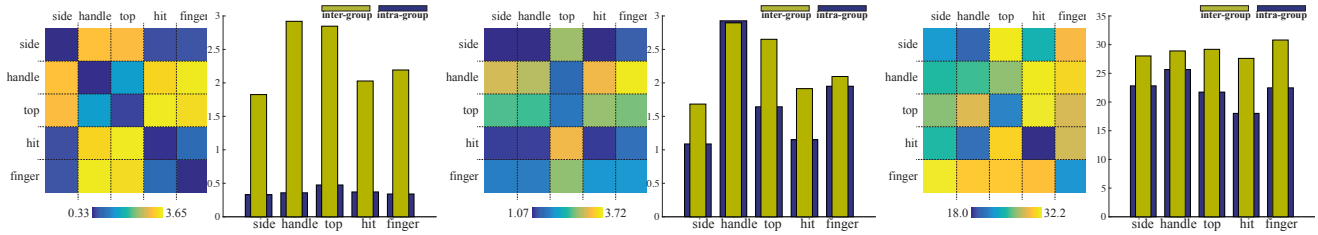


Fig. 3. Classification with a simpler, Euclidean proximity inter-sensor cost versus dynamic time warping. Left: Euclidean, on the original, non-rotated dataset. Middle resp. Right: Euclidean/DTW, on the dataset with the added rotated variants. With Euclidean cost, interaction "grasp-from-handle" is not reliably detected, and the distance matrix is highly non-diagonal.

virtual coffee mug: grasp from the side, the handle, or the top, hit from the side, and touch with one finger. Each interaction was performed at least 4 times by 3 different subjects, yielding 61 sequences. We used an oct-tree with $L = 4$ layers for our experiments, with a total of 568 sensors.

Classification accuracy. We group the interaction sequences according to the type of interaction performed (5 groups). We then use our descriptor and similarity measure to compute the average pairwise distance between interactions in the same group - e.g. between all pairs of "grasp-from-side" interactions; we call this the "intra-group" distance for this group. The "inter-group" distance is the average pairwise distance between interactions of this group and interactions in other groups. A smaller "intra-group" rather than "inter-group" distance indicates successful interaction classification. We report these results together with a pseudo-color rendering of the the pair-wise inter-group distances in Fig. 2.

Input modality. The classification results with the hand-tracker input modality are shown in Fig. 4. The overall increase in accuracy is due to the cleaner input: the scene-flow based input only contains partial point-clouds, due to our single camera setup. Instead, the hand-tracker provides full point clouds of the hands, without outlier points from other surfaces and typically at much higher resolution (tracking of individual fingers).

Rotation invariance. In an interaction classification pipeline, a side grasp should always be classified as such, regardless of from which side it is performed. To test for rotational invariance, we augmented our dataset of real interaction point clouds with sequences produced by synthetically rotating the point cloud data around the object axis at various angles. This produced 149 new sequences, each with in own interaction type group. We used these sequences as a "test"-set, and computed average pairwise inter-group and intra-group distances with the interactions in the "training"-set (the original unrotated sequences)

Results are shown in the rightmost columns of Fig. 3. Note that

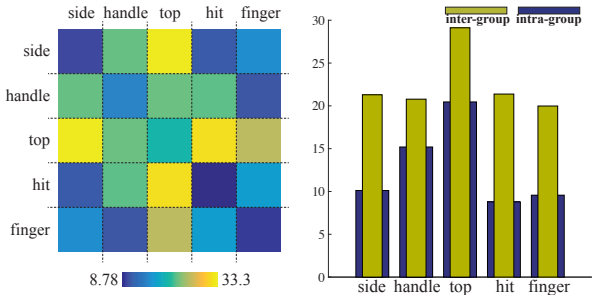


Fig. 4. Classification accuracy, with input from a hand-tracker [18].

the classification accuracy is not significantly different than Fig. 2, which indicates that our pipeline is able to correctly classify interactions even when they are performed from various angles and at different orientations. This means that to a certain extent the descriptor is invariant to in-plane changes of camera orientation. In comparison, the two leftmost plots in Fig. 3 show the results (without and with rotations in the dataset) where the pairwise inter-sensor cost in the optimal transport is the 3D Euclidean proximity distance (Section 2.2). This simple cost works well when no rotations are present, but fails in the more general case.

Timings. Our experiments were performed on a 2013 Mac Pro computer with an 3.5 GHz 6-Core Intel Xeon E5 processor and 64 GB of RAM. We used C++ for constructing the oct-tree sensor layout and extracting the descriptors, and MATLAB to compute the interaction classification distance. With $L = 5$ oct-tree layers, extracting the descriptor takes approximately 20.1 sec for a single interaction (around 385 RGBD video frames). Evaluating the distance for a pair of interactions takes approximately 7 seconds.

Interaction similarities. Our pipeline is able to detect similarities between different types of interactions. The pseudo-color plots shown in Fig. 3(right) indicate that similar interactions (e.g. a side grasp and a handle grasp) are detected to be closer together than to significantly different interactions.

4. CONCLUSION

Detecting and labeling dynamic, proximal interactions with objects can be an important prior for deducing object functionality, which can greatly aid autonomous agents by navigating in unstructured environments. We considered the problem of representing and classifying such human-centric interactions, captured through our novel hardware setup combining an RGBD sensor and a VR feedback loop. We introduced and validated (via classification experiments) a descriptor and a matching similarity measure for interactions.

In the future, we will exploit more advanced properties of the motion flow, e.g. velocity vectors of the captured motions in addition to raw point cloud data, to achieve a more nuanced differentiation of interactions. Capturing such directional information reliably for general interaction scenarios remains challenging, especially with a single view sensor setup. We also intend to work on tools for transferring knowledge of interactions and affordance between objects of the same class (e.g. as found in repositories), and for identifying different objects through the space of their possible interactions. We envision a large scale repository of human interactions similar to [6] for robotic interactions - which would allow us to explore our descriptor in the context of deep learning approaches.

5. REFERENCES

[1] Matei Ciocarlie, Kaijen Hsiao, Edward Gil Jones, Sachin Chitta, Radu Bogdan Rusu, and Ioan A. Şucan, *Towards Reliable Grasping and Manipulation in Household Environments*, pp. 241–252, Springer, Berlin, Heidelberg, 2014.

[2] Jacopo Aleotti, Dario Lodi Rizzini, and Stefano Caselli, "Perception and grasping of object parts from active robot exploration," *Journal of Intelligent & Robotic Systems*, vol. 76, no. 3, pp. 401–425, 2014.

[3] Lerrel Pinto, Dhiraj Gandhi, Yuanfeng Han, Yong-Lae Park, and Abhinav Gupta, "The curious robot: Learning visual representations via physical interactions," *CoRR*, vol. abs/1604.01360, 2016.

[4] Antonio Bicchi and Vijay Kumar, "Robotic grasping and contact: A review," in *IEEE ICRA 2000, April 24-28, 2000, San Francisco, CA, USA*, 2000, pp. 348–353.

[5] J. T. Schwartz and M. Sharir, "A survey of motion planning and related geometric algorithms," *Artificial Intelligence*, pp. 157–169, 1988.

[6] Sergey Levine, Peter Pastor, Alex Krizhevsky, and Deirdre Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *CoRR*, vol. abs/1603.02199, 2016.

[7] G. M. Bone, A. Lambert, and M. Edwards, "Automated modeling and robotic grasping of unknown three-dimensional objects," in *IEEE ICRA 2008*, May 2008, pp. 292–298.

[8] Lerrel Pinto and Abhinav Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," *CoRR*, vol. abs/1509.06825, 2015.

[9] Dov Katz, Jacqueline Kenney, and Oliver Brock, "How can robots succed in unstructured environments," in *Workshop on Robot Manipulation: Intelligence in Human Environments at Robotics*, Zurich, Switzerland, June 2008, Science and Systems.

[10] J.K. Aggarwal and M.S. Ryoo, "Human activity analysis: A review," *ACM Comput. Surv.*, vol. 43, no. 3, pp. 16:1–16:43, Apr. 2011.

[11] J. F. Hu, W. S. Zheng, J. Lai, and Jianguo Zhang, "Jointly learning heterogeneous features for rgb-d activity recognition," in *2015 IEEE CVPR*, June 2015, pp. 5344–5352.

[12] Abhishek Gupta, Clemens Eppner, Sergey Levine, and Pieter Abbeel, "Learning dexterous manipulation for a soft robotic hand from human demonstration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, http://arxiv.org/abs/1603.06348.

[13] Jeffrey Mahler, Florian T. Pokorny, Brian Hou, Melrose Roderick, Michael Laskey, Mathieu Aubry, Kai Kohlhoff, Torsten Kroeger, James Kuffner, and Ken Goldberg, "Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards," in *2016 IEEE ICRA*, 2016.

[14] Andreas ten Pas and Robert Platt, *Localizing Handle-Like Grasp Affordances in 3D Point Clouds*, pp. 623–638, Springer International Publishing, Cham, 2016.

[15] O. Oreifej and Z. Liu, "Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences," in *CVPR, 2013 IEEE*, June 2013, pp. 716–723.

[16] Ruizhen Hu, Oliver van Kaick, Bojian Wu, Hui Huang, Ariel Shamir, and Hao Zhang, "Learning how objects function via co-analysis of interactions," *ACM Trans. Graph.*, vol. 35, no. 4, pp. 47:1–47:13, July 2016.

[17] T Schmidt, R Newcombe, and D Fox, "Dart: Dense articulated real-time tracking," *of Robotics: Science* , 2014.

[18] Iason Oikonomidis, Nikolaos Kyriazis, and Antonis A Argyros, "Efficient model-based 3d tracking of hand articulations using kinect.," in *BMVC*, 2011, vol. 1, p. 3.

[19] Andrea Tagliasacchi, Matthias Schröder, Anastasia Tkach, Sofien Bouaziz, Mario Botsch, and Mark Pauly, "Robust articulated-icp for real-time hand tracking," *Comput. Graph. Forum*, vol. 34, no. 5, pp. 101–114, 2015.

[20] Richard A. Newcombe, Dieter Fox, and Steven M. Seitz, "Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time," in *IEEE CVPR*, June 2015.

[21] Zhirong Wu, Shuran Song, Aditya Khosla, Xiaoou Tang, and Jianxiong Xiao, "3d shapenets for 2.5d object recognition and next-best-view prediction," *CoRR*, vol. abs/1406.5670, 2014.

[22] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An Information-Rich 3D Model Repository," *ArXiv e-prints*, Dec. 2015.

[23] Hiroaki Sakoe and Seibi Chiba, "Readings in speech recognition," chapter Dynamic Programming Algorithm Optimization for Spoken Word Recognition, pp. 159–165. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[24] Cédric Villani, *Topics in optimal transportation*, Graduate studies in mathematics. American Mathematical Society, cop., 2003.

[25] Marco Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Advances in Neural Information Processing Systems*, 2013, pp. 2292–2300.

[26] Mariano Jaimez, Mohamed Souiai, Javier González-Jiménez, and Daniel Cremers, "A primal-dual framework for real-time dense rgb-d scene flow," in *IEEE ICRA*, may 2015, pp. 98–104.

[27] S. Garrido-Jurado, R. Muñoz-Salinas, F.J. Madrid-Cuevas, and M.J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280 – 2292, 2014.

[28] S. Garrido-Jurado, R. Muñoz-Salinas, F.J. Madrid-Cuevas, and R. Medina-Carnicer, "Generation of fiducial marker dictionaries using mixed integer linear programming," *Pattern Recognition*, vol. 51, pp. 481 – 491, 2016.