Author:
Ruiz Libreros, Eduardo D

Title:
A geometric approach for fast affordance determination in 3D

A geometric approach for fast affordance determination in 3D

By

EDUARDO DANIEL RUIZ LIBREROS

Department of Computer Science
UNIVERSITY OF BRISTOL

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of DOCTOR OF PHILOSOPHY in the Faculty of Engineering.

MARCH 2019

Word count: 33 681

Abstract

Visual perception for robotics aims to provide robots with the *relevant* information about the environment such that they can interact with it, understand it and accomplish tasks successfully. Thus, agents that need to act on their surroundings can significantly benefit from the perception of their interaction possibilities or affordances. The concept of affordances calls for an approach to visual perception that is free from complex representations, and that is there to help the agent to interact with the world.

This thesis presents an approach for the determination of affordances in visually perceived 3D environments. The introduced method builds on the hypothesis that geometry on its own provides enough information to enable the detection of significant interaction possibilities in the environment. The motivation behind this is that geometric information is intimately related to the physical interactions afforded by objects in the world.

The work presented in this thesis introduces a geometrical representation for the interaction between two entities in 3D space. The nature of the approach provides the possibility to *generically* describe interactions for everyday objects such as a mug or an umbrella, and also for more complex affordances such as *Sitting* or *Riding*. Experiments with numerous synthetic and RGB-D scenes show that the representation enables the prediction of affordance candidate locations in novel environments at fast rates and from a single training example, i.e. one-shot learning. Then, it is shown that the one-shot capability of the proposed approach and the abstraction power of state-of-the-art data-driven methods allow to devise a compact and optimised representation for the detection of multiple affordances in any given location. Experiments and evaluations show that the proposed algorithms achieve high precision rates that outperform alternative methods. The evaluations include human validations via crowdsourcing, which show the meaningfulness of the affordance predictions made with the proposed algorithm.

DEDICATION AND ACKNOWLEDGEMENTS

First, I would like to express my gratitude to my supervisor Prof. Walterio Mayol-Cuevas for his generous guidance and support throughout this process.

I would also like to thank the Mexican government who, through CONACYT, provided the funding to cover the cost of my PhD studies.

I was very fortunate to meet many great people during my life in Bristol. Their advice, discussion and friendship made more enjoyable my life as a postgraduate student in the UK. Thank you all.

I cannot express enough how thankful I am to my mother for her love and encouragement throughout my whole life. Last but not least, I must thank Lorna for all her love and support.

Author's declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: .... DATE: ....

Table of Contents

LIST OF TABLES

TABLE **Page**

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

To perceive an affordance is not to classify an object

James J. Gibson [1]

Whereas recognising objects or entities in a scene is an important competence of intelligent agents, a lot more is needed for visual scene understanding. For cognitive agents (e.g. robots) that need to interact with their surroundings, it can result more useful to reason and learn based on functional properties or affordances. The concept of affordance was coined by James J. Gibson [1] more than five decades ago in the field of ecological psychology. For Gibson, affordances are action opportunities in the environment that are *directly* perceived by the observer. According to this, the goal of vision was to recognise the affordances rather than elements or objects in the scene. Even David Marr, who provided early efforts to computationally address the problem of visual recognition [2], posed the question of *what type of information does vision really convey?*. According to Gibson and Marr, the visual perception problem becomes that of recovering the *valid* properties "offered" by the environment.

Over the last couple of decades, the affordance concept has evolved and has been adopted in multiple fields such as robotics, computer vision, artificial intelligence and psychology; bringing different views or interpretations on what an affordance actually is. To some extent, the general agreement is that affordances are "relations" taking place in the environment: between the environment and an agent, or relations in the environment perceived by an observer.

Being linked with visual perception and perhaps motivated by the top-down view adopted in computer vision research, much of the attention given to the problem of affordances has focused on the recovery of complex representations of the world, internal symbolic relationships or semantic category information, which undermines the idea of *direct* and economical perception of affordances proposed by Gibson. The idea of being able to directly determine affordances has faced many dilemmas, namely the challenging problems of visually recovering the relevant properties of the environment in a robust and accurate manner.

The problems of visually recovering the “valid” properties of the environment that allow to detect affordances are further accentuated in robotics, due to the fact that robots need to be able to work in environments that are cluttered, unstructured and unknown. Developing a system that is able to work under these conditions is a difficult problem; more so when traditional affordance detection approaches often need to recognise objects semantically in the environment or have previously extensively trained for as many cases (examples) as possible in order to generalise to novel scenarios. Robots would benefit from affordance detection approaches that do not rely on object recognition, nor environment’s features costly to estimate; dropping or relaxing such requirements in the perception system can allow robots to have greater generalisation capabilities enabling them to accomplish their task efficiently.

Following Gibson’s idea of direct perception of affordances, and motivated by studies in neuroscience showing that affordance detection does not require semantic reasoning [3–6]; we hypothesise that geometric information or shape provides enough information for an agent to directly perceive the interaction opportunities in its environment. Notably, this has support on research showing that parts of the brain involved in the visual perception of affordances rely on shape, size, and depth properties of objects in the environment [7–9].

Particularly, this thesis is concerned with the detection of affordances between two entities in the environment; more specifically, static interactions between generic pairs of solid entities in 3D space, i.e. pointclouds. Examples of this detection scenario are the prediction of suitable locations for a coat-hanger that needs to be hanged or a glass that needs to be filled (e.g. with water). Moreover, the methods here proposed also allow to consider a model of a human body for the detection of suitable places for sitting or riding a motorcycle.

Briefly speaking, the methods here presented would allow a robot to enter into a never-before-seen environment and detect candidate locations for multiple interactions such as *Hanging* a handbag, *Placing* a bottle, etc. Among the main characteristic of the proposed representation is the fact that it remains agnostic to semantic categories, complex surface features and that is able to generalise to completely novel environment training from a single demonstrated interaction. Furthermore, the method does not rely on the scene or the objects in isolation but in the interactions of both. This is achieved by *hallucinating* the interactions in the scene; thus, the system does not know in advance if a region in the input scenario affords the interaction. Detections are made online at fast rates by investigating, via the proposed descriptor, the likelihood of the interaction taking place at a region of interest. We argue that the aforementioned characteristics align well with Gibson's idea of *direct* perception of affordances.

An example scenario of the problem of affordance determination in 3D environments tackled in this thesis is shown in Figure 1.1. The same Figure shows examples of the type of predictions made by the proposed approach; the detections shown in the bottom image correspond to *Placing* a bottle (orange), a bowl(blue), *Hanging* a coat-hanger (green) and handbag (yellow).

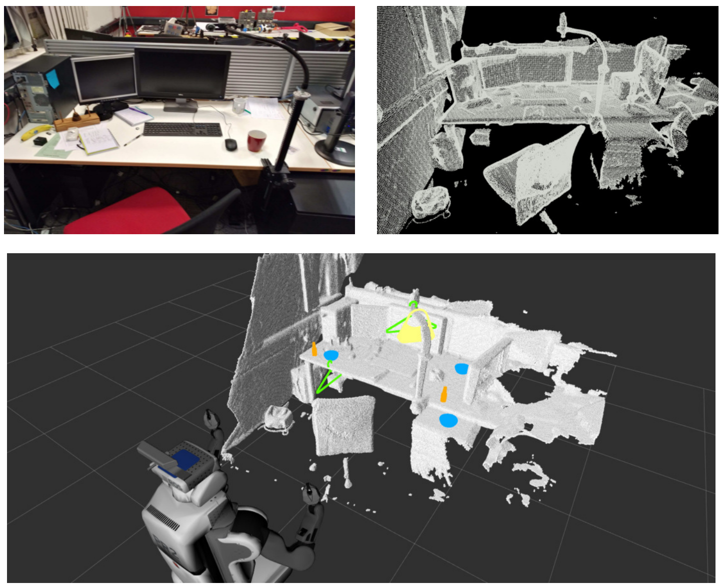


Figure 1.1: The affordance detection approach presented in this thesis enables a robot to go into a completely unknown environment and predict good candidate locations affording to *Place, Hang, Fill, etc.* everyday objects such as bowls, mugs, bottles, handbags, and more. The method allows the agent to give an answer to perceptual questions like: "*Where can I hang a handbag?*", or more generally "*What can I afford to here?*". All this from a single interaction example obtained with different and synthetic 3D models.

1.1 Contributions

In summary, the contributions made with the research reported in this thesis are:

- A 3D geometrical representation for the characterisation of interactions between two rigid entities in space: The Interaction Tensor.
- An affordance descriptor and similarity function that enable to quickly evaluate the likelihood of an interaction taking place in completely novel RGB-D scenarios.
- A scalable one-shot learning algorithm that allows to efficiently query multiple affordances at any given location in novel environments without compromising detection speed or quality.
- A hybrid approach that combines data-driven abstraction power and rich geometric information to learn compact, multiple-affordance representations allowing to predict meaningful interaction opportunities at high frame rates.

1.2 Overview of the thesis

The work presented in this thesis is organised as follows:

Chapter 2 In this chapter can be found a review of previous approaches to the problem of affordance detection and learning. The key concepts and background on affordances in psychology and neuroscience are presented. After reviewing the most relevant works in robotics and computer vision, another source of inspiration is introduced: interactions and functionality in computer graphics. Finally, the key aspects that motivated the approach proposed in the thesis are discussed.

Chapter 3 This chapter presents the core of the proposed representation for affordance detection. Here are introduced the details regarding the characterisation of the interaction between any given pair of objects: the interactions tensor. Examples of the representation are provided for five generic interactions, namely: *Filling, Hanging, Placing, Sitting* and *Riding*. Later, it is shown how this geometrical representation can be employed to devise a descriptor for affordance determination in 3D environments.

Chapter 4 Starts by describing the one-shot learning algorithm that leverages the interaction tensor to detect affordances in novel (i.e. unknown) synthetic scenes and RGB-D scans of real indoor environments. Then, this chapter shows that by efficiently clustering single example interactions, it is possible to obtain a scalable, multiple-affordance representation able to work at high frame rates and producing top quality predictions as validated with human criteria.

Chapter 5 In this chapter is introduced the hybrid approach that allows to devise an optimised multiple-affordance representation. The proposed approach leverages a state-of-the-art data-driven method to efficiently parse the 3D input, learning key locations that inform the iT method of "where to look" while predicting multiple affordances, i.e. keypoint sampling. Evaluations and comparisons with alternative methods are presented, including examples of affordance detection on publicly available datasets.

Chapter 6 This application chapter shows that the proposed approach is suitable for robotic and augmented reality systems due to the fast rates at which predictions are performed. Robotic simulation examples are provided showing how a robotic platform could benefit from fast affordance detection in unknown environments. Then, the affordance lantern concept is introduced, which allows to augment a real indoor environment with dynamically discovered affordances, i.e. affordance-based scene augmentation.

Chapter 7 This final chapter contains the conclusions and final discussion regarding this research. The key contributions and limitations of the current work are summarised, and avenues for future work are proposed.

1.3 Publications

This thesis is primarily based on the work from the following publications:

1. E. Ruiz, and W. Mayol-Cuevas,*Where can I do this? Geometric Affordances from a Single Example with the Interaction Tensor*, in Robotics and Automation (ICRA),2018 IEEE International Conference on, pp. 2192-2199

2. E. Ruiz, and W. Mayol-Cuevas,*What can I do here? Leveraging Deep 3D saliency and geometry for fast and scalable multiple affordance detection*, Manuscript in preparation. arXiv:1812.00889. 2019.

Chapter 2

Literature Review

2.1 Introduction

Affordances relate to the possible actions that are *directly* perceived in the environment by an agent. The concept was coined in 1967 by American psychologist James J. Gibson. Since then, and especially over the last couple of decades, many attempts have been made to develop systems that incorporate such perception idea. One major challenge when investigating affordances in robotics, artificial intelligence or computer vision is the many interpretations that have been made of Gibson's ideas. Even Gibson himself made multiple attempts to develop the concept of affordances. On a first approach [10] he writes:

> " *When the constant properties of constant objects are perceived (the shape, size, colour, texture, composition, motion, animation, and position relative to other objects), the observer can go on to detect their affordances. I have coined this word as a substitute for values, a term which carries an old burden of philosophical meaning. I mean simply what things furnish, for good or ill. What they afford the observer, after all, depends on their properties.*" (p.285)

Which seems to attribute the idea of affordances primarily to the detection of properties of the environment by an observer. Then, in his later work [1], Gibson continues to develop the idea:

> *"The affordances of the environment are what it offers the animal, what it provides or furnishes, either for good or ill. The verb to afford is found in the dictionary, but the noun affordance is not. I have made it up. I mean by it something that refers to both the environment and the animal in a way that no existing term does. It implies the complementarity of the animal and the environment"* (p. 127)
>
> *"...But I now suggest that what we perceive when we look at objects are their affordances, not their qualities. We can discriminate the dimensions of difference if required to do so in an experiment, but what the object affords us is what we normally pay attention to. The special combination of qualities into which an object can be analysed is ordinarily not noticed."*(p.134)

In this work Gibson seems to build on the idea of an animal-environment system, where they compliment each other; therefore affordances do not exist only as properties of the environment (e.g. colour, texture, size, shape) as Gibson first suggested. These contrasting views gave way to a controversial debate regarding the true nature of the affordance concept. In fact, just within the last decade efforts have been made [11–15] to revisit and organise the several works that have tackled the affordance problem in robotics, computer vision and artificial intelligence.

In the following section, a brief overview of the definitions and formalisms of affordances is presented. Following that, evidence from neuroscience studies regarding affordance perception and learning are introduced. Then, the most relevant works on robotics and computer vision are discussed. Finally, other approaches outside these latter fields but also studying agent and environment interactions are presented.

2.2 Affordance Theory

Michael T. Turvey [16] defined affordances as the *dispositional properties* of the environment that only under specific circumstances become evident, e.g. the presence of an agent. He considers affordances central to prospective control, guiding the agent's planning or behaviour by informing about the possible actions to perform. In contrast to Gibson's idea of direct perception, Vera and Simon [17] argue that an affordance is not merely a property of the physical environment but rather an internal semantic mapping (e.g. symbol) encoding functionality in terms of visual display, where the complex visual display is produced by the physical scene that is being observed.

Notably, their view coincides with Gibson's first proposal of affordances being nested in the environment.

The purpose of affordances is to satisfy the needs of an agent. That is the claim of Shaw [18], whose view on affordances is that they provide the harnesses for directing causes, i.e. a source of information for the agent to make decisions, which are complemented by effectiveness. In this view, affordances remain dormant in the environment until complemented by the agent's effectivities. Ultimately, the argument follows the idea that affordances are referenced in the environment.

Contrary to the latter views, Heft [19] sees affordances as dynamic functional relationships between the environment and agent-related factors such as physical attributes (e.g. body size, strength, posture), intentionality and perceptual learning. This point of view sees affordances available percepts that an individual "picks-up" in a historical and reciprocal series of processes. The idea of the agent-environment system is also shared by Stoffregen [20], who defines affordances as emergent properties that do not reside in either the environment nor the agent alone. In contrast to Turvey, Stoffregen argues that if affordances were only in the environment, the agent would need to reason, or infer from other sources of information, the actions available *to him*; a process that would contradict Gibson's *direct* perception of affordances.

This agent-environment system is also considered in the definition of Chemero [21], yet not as properties but as relations between specific aspects of the involved parts. He defines the affordance perception process as the placement of *features* in the environment, meaning that perceiving an affordance is a matter of perceiving that the situation has a certain *feature* that supports a certain kind of action. In his work, Chemero argues that affordances exist without the presence of an agent, however, in order to describe and study the real physical entity, an affordance requires the potential existence of at least one observer that can perceive it Similarly, the definition brought up by Michaels [22], regards affordances as existing independently from perception but subjective to the agent. The agent is needed to actualise or make affordances manifest via its effectivities.

Finally, one of the most popular definitions of affordance comes from industrial design. Here, Norman [23] merges concepts of ecological psychology and ergonomics to define affordances as the properties of objects that make visible how they can be used. Whether perceived or real, these properties should easily convey information about the actions that are possible to perform with the object being designed.

2.2.1 Computational Formalisms

Efforts have also been made to formalise the concept of affordances in the areas of computer science, artificial intelligence and robotics; thou often ignoring the problem of perception. One of such formalisms was provided by Steedman [24], who uses Linear Dynamic Event Calculus to provide a computational interpretation of affordances. In this formalism, actions related to objects are linked to events in the environment; thus, Steedman considers pre-conditions, actions and post-conditions that allow for action planning by forward-chaining. Noticeably, this formalism uses the concept of object entities (e.g. door) but does not make any link with the perception aspect. Examples of an action/event function in Steedman's formalism would be *if x is shut and you push it, it becomes open (and vice versa)* and *if you are in and you go through x then you become out*. The set of such functions for a particular object constitutes its affordances, for instance: if $x \leftarrow door$, $affordances(door) = \langle push, go - through \rangle$.

Remarkably, one of the most used formalisms in robotics is the one proposed by Sahin et al. [25]. Taking inspiration from Chemero, affordances are viewed as relations between an agent and the environment. Using a symbolic notation of the form (*effect,(entity,behaviour)*), this formalism describes the effect of applying certain behaviour to an entity; such representation would allow an agent to build a knowledge base based on its experiences.

One more popular formalism is that of Montesano et al. [26], who proposed a developmental formalisation that uses a probabilistic representation for affordances: Bayesian Networks. In this formalism the nodes of the network represent motor actions $\mathscr{A}$, robot features $\mathscr{F}_r$, object features $\mathscr{F}_o$ and perceived effects $\mathscr{E}$; the edges of the network are the parameters of the conditional probability distributions. Once the structure of the network and the parameters (edges) can be learned/updated by robot exploration or observations, which allows for the computation of e.g. the probability of observing effects given a robot action a for object features f: $p(\mathscr{E}|\mathscr{A} = a, \mathscr{F}_o = f)$.

Finally, one more relevant formalisation is that of the Object-Action Complexes (OACs) [27]. OACs attempt to link low-level sensorimotor knowledge with higher-level reasoning, which allows for planning in cognitive systems. Kruger et al. define OACs as triplets of the form (E, T, M) which refer to motor executions E, a function prediction about the environment's state change after executing E, and a statistical measure M of the success of previous executions. In this formalism, affordances are considered implicitly in the prediction function, since T only maps the *relevant* portions of the environment, e.g. the perceptual invariants of affordances.

2.3 Evidence from Neuroscience

As discussed by Jamone et al. [14], studies in neuroscience supporting the concept of affordance have provided evidence which can be summarized in : i) the fast perception of the object properties that are related to actions, ii) that there is a strong link between actions and perception representations and, iii) that recognition and semantics are not required to perceive the affordances of objects.

Visual information in primate cerebral cortex follows two separate pathways in order to be processed [3, 28–30]. Semantic information about objects such as categorisation and recognition is processed by the ventral pathway [4]; whereas information to control actions such as grasping or reaching is processed comparatively faster [31] by the dorsal pathway [32]. Specifically, the dorsal pathway carries out important sub-processes for affordance representation [33], i.e. the representation of geometric features such as depth, surface and axis [9].

Neurons in the ventro-dorsal substream respond to somatosensory (e.g. sensation of pain, temperature, touch, proprioception) and visual stimuli [34]. More precisely, F5 neurons are involved in the organisation of grasping, manipulation and movements that require space [34]; actions that according to Jeannerod et al. [7] require computations of size and shape of the objects.

Murata et al. [35], conducted experiments with monkeys and tested four conditions: object grasping in light, object grasping in the dark, object fixation, and fixation of a spot of light. In this work, two types of neurons were identified: motor and visuomotor neurons. Their results showed that a high percentage of F5 neurons responded to the presentation of 3D graspable objects; the response was present even in the absence of subsequent movement towards the objects. Later, further distinctions in the F5 visuomotor neurons were made [36]: *canonical* neurons responding to observation of 3D objects and *mirror* neurons responding to action observation. In a more recent work [8], it was detected that F5 visuomotor neurons are involved with the perception of orientation, size and form (e.g. of the object or hand) that are relevant for the visual control of movements.

Perhaps even more interesting are the similarities shared with studies on human subjects. For instance, the detection of motor neuron activations shortly after observing graspable objects [37], which is more pronounced when the objects being observed are within reachable space [38].

As for semantics, although studies showed that both visual pathways interact in visually guided behaviours [39], experiments have demonstrated that *motor* affordances (e.g. action related regions of the brain) are more activated by visual stimuli from the objects' physical appearance rather than semantic category [40], objects' name or symbols (i.e. written words) [41]

2.4 Affordances in Robotics

Works considering affordance perception appeared in the computer vision and robotics communities during the early 1990 decade [42, 43]. Since then and over the last 20 years, these fields have given an important amount of attention to the study of affordances. One main benefit of the affordance concept in these fields is the potential to alleviate the need for complex models of the world, allowing the systems to focus on the most informative or task-relevant regions of the environment.

Horton et al. [11] made an early review of works in robotics and artificial intelligence that incorporated ideas from affordance perception and learning. In Horton's review can be found that roboticists were already considering the ecological design of agents, even before affordances became a popular topic in the area. Robotic systems where *simple* yet efficient perception systems were tightly coupled to mechanisms for planning and acting; for instance, the work of Brooks [44], who took inspiration from ideas of reactive or behaviour-based robotics and developmental robotics.

In fact, a big body of research in affordances for robots comes from the developmental robotics field [12, 14, 45, 46]. This area of robotics takes inspiration from studies in ecological psychology and psychophysics about the developmental principles observed in children in order to design cognitive and behavioural capabilities for robots; for instance, using sensorimotor knowledge, attentional mechanisms as well as factors related to the environment and the robot interactions. Generally speaking, the focus of affordances in developmental robotics is on the representation and learning of robot's actions and their consequences in the environment. The early work of Fitzpatrick and colleagues [47, 48], showed that a robot is able to learn *rollability* and replicate demonstrations by executing exploratory actions (e.g. push and pull) on four different objects.

Another important amount of research has been dedicated to the detection and learning of *grasping* affordances, which is a key skill for robots that need to interact with the world. However, only those works that explicitly work under the affordance concept are briefly reviewed. Lastly, works that do not investigate grasping-only affordances or do not involve a robot actively interacting with the environment are discussed. Example of these systems are approaches that work towards understanding what the environment affords to humans; for instance, in the field of Human-Robot Interaction. Another example of works that are presented in Subsection 2.4.3 is those aiming to understand the affordances of tools by "observing" examples of these tools.

2.4.1 Affordances from robot exploration

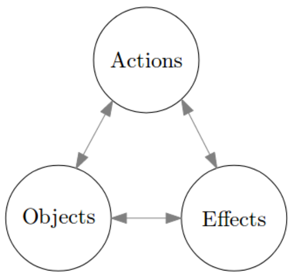
Here are introduced works in robotics research where the *direct* perception of affordances is translated into a direct linkage between sensory data and motor skills. A first example of this is the work of Fritz et al. [49], who used a crane-like robot with the ability to pick objects up in order to learn *lift-ability* affordances based on the objects' appearance (e.g. colour) and pose (i.e. distance and orientation). The system is then extended to use *generic* visual features, i.e. Scale Invariant Feature Transform (SIFT)[50], in a reinforcement learning framework [51].

Another group of works studied the problem of robot navigation, where the concept of *traversability* affordance was used to allow robots to navigate in indoor and outdoor environments. A good amount of approaches [52–56] build on the formalism for affordance learning introduced by Sahin et al. [25], this allowed a mobile robot to navigate indoor environments using basic behaviours (e.g. turns, go-forward). Objects with regular geometries such as boxes, spheres and cylinders are perceived and avoided by computing shape features (e.g. surface normal vectors) and distances from a depth image. The outdoor navigation problem is studied by Kim et al. [57], who represented the world by visual appearance features of the environment (Law's masks [58]) and geometric information (*xyz* coordinates) using a stereo camera system. The model is then extended[59] to include *move-ability* and *support-ability* affordances which are grouped into categories identifiable as ground, table, rollable, pushable, amongst others.

Hermans et al. [60–63] proposed a method to learn *push-able* and *pull-able* affordances for objects in a table-top manipulation task. The robot learns a mapping from the state of the world (e.g. object shape features) to the contact location that best achieves the desired trajectory. The object's shape features capture the contact-point local shape and the object's global shape using

2D projection over the table.

The work of Montesano et al. [26, 64, 65] studied the problem of affordance learning as a problem of structure learning, representing affordances as the (probabilistic) relations between actions, objects, and effects. The object-effect-action relations are shown in Figure 2.1, with this model a robot is able to learn a mapping between basic exploratory actions, such as tap and grasp, and the effects that those actions have on an object.



Inputs	Outputs	Function
(O, A)	E	Effect prediction
(O, E)	A	Action recognition/planning
(A, E)	O	Object recognition/selection

Figure 2.1: Relations between objects(O), actions(A) and effects(E) for affordance learning in works such as [26, 64–69]. Figure taken from [65]

Multiple works [66–69] have built on this model and have used statistical relational learning to encode relations between (afforded) actions and *percepts*. The experiments showed that the robot is able to replicate demonstrations of humans interacting with an object by exploiting the acquired knowledge. Moreover, the model is extended by encoding the effects of single-object actions relative to other objects, which allows the robot to learn two-object *relational* affordances [66], higher-level manipulation actions such as *makeSpace* or *moveAround* [67], and two-arm manipulation [69]. In [68] a list of object properties and afforded-actions is used by a similar model to learn co-occurrence probabilities for occluded object search. The affordance model Montesano et al. is further developed to account for co-occurrence of verbal(speech) descriptions [70].

To a certain extent, these earlier approaches do not focus on the perception problem and assume knowledge about the (simulated) scene, where the objects' shape (e.g. cylinder, prism, etc.), position, size and type (e.g. glass, cup, bowl) is provided by the simulator [68, 69]. Alternatively, the simulations consider a stereo camera system that allows to localise objects in 3D space using colour segmentation and triangulation [66, 67].

Relational affordances have also been studied for learning tool usage [71–79]. In these approaches, the main idea is to learn, also via exploration, the effect that one object (i.e. tool) has over or relative to another. These systems focus on capturing the position and shape of objects in table-top manipulation scenario to plan and achieve a target effect or object configuration, the state of the environment (i.e. objects) is represented by 2D shape features of colour segmented blobs. Features such as area, convexity, circleness, squareness etc. are considered to identify objects in [73, 74, 77, 79]. More complex feature combinations [76] and more tool options [80] are considered by Mar and colleagues, who additionally take into account the way in which the tools are grasped (e.g. rotation around handle). In [72] the system leverages several modules for object segmentation, detection and recognition to represent and track the effects of the tools over other objects. Figure 2.2 shows an example scenario of these approaches.

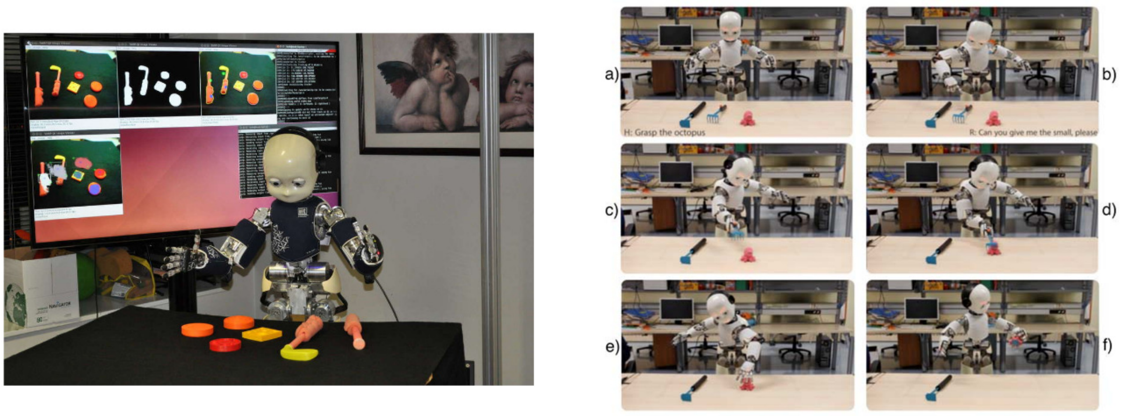
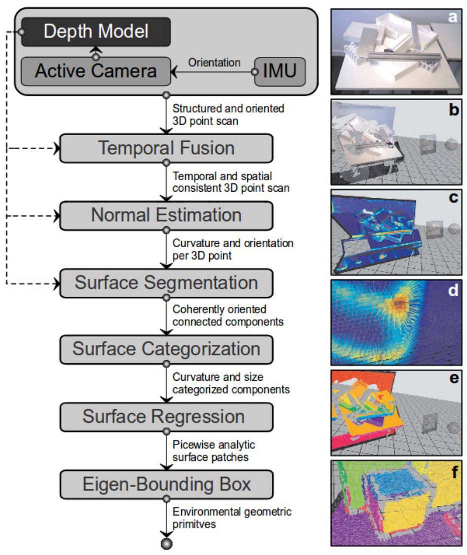


Figure 2.2: Images show a typical scenario for *relational* affordance detection for tool usage learning. Image on the left shows the visual perception system (i.e. object segmentation and tracking). Image on the right shows an interaction example where a robot is asked to grasp an object. The robot has learned that a particular tool allows him to bring the target closer. Images taken from [78] and [72] respectively.

Griffith et al. [81, 82] showed a robot able to learn objects' *contain-ability* affordance. Similar to the previous works, the robot executes a series of *basic* behaviours such as pushing, grasping, dropping, etc. and visually [83], or visually and acoustically [81] identifies the effect of applying such behaviours. The system is able to classify objects as containers or non-containers by (unsupervised) clustering the features observed after interacting with objects.

Similarly, works have used low-level learned behaviours to bootstrap complex affordance learning (e.g. *stackability*) [84–88]. Here, a robot learns rules and object-effect categories (rollable, unstable, hollow, solid, inserted, etc.) that allow him to build a plan in a tower-building task. Information is acquired from an RGB-D camera by computing position, size and 3D shape (e.g. histogram of surface normals) for every object on top of a table.

Whole-body affordances are studied in [89–92] by Kaiser and colleagues, where a humanoid robot generates affordance hypotheses (e.g. *leaning-on, grasping, support*) using pointclouds from an RGB-D camera. In [89], affordance hypotheses are used for planning according to end-effector reachability. In [90], the proposed affordance hypotheses are validated via robot exploration, which allows the robot to learn *push-able,lift-able* affordances. These unimanual affordances are then used to learn higher-level affordances (e.g. prismatic grasp) [91]. The affordance detection system employed by this series of works is based on geometrical attributes of the detected/fitted primitive surfaces, i.e. planar, cylindrical, spherical and circular surfaces. The process that allows to obtain the surface primitives is shown in Figure 2.3. The pipeline involves the temporal fusion of RGB-D images to obtain a "clean" pointcloud representation, normal vector estimation, surface segmentation based on curvature information, the categorisation of surface segments according to curvature and size, and the fusion with inertial sensor data in order to find planar surfaces.


Depth Model
Active Camera
Orientation
IMU
Structured and oriented 3D point scan
Temporal Fusion
Temporal and spatial consistent 3D point scan
Normal Estimation
Curvature and orientation per 3D point
Surface Segmentation
Coherently oriented connected components
Surface Categorization
Curvature and size categorized components
Surface Regression
Picewise analytic surface patches
Eigen-Bounding Box
Environmental geometric primitves
a
b
c
d
e
f

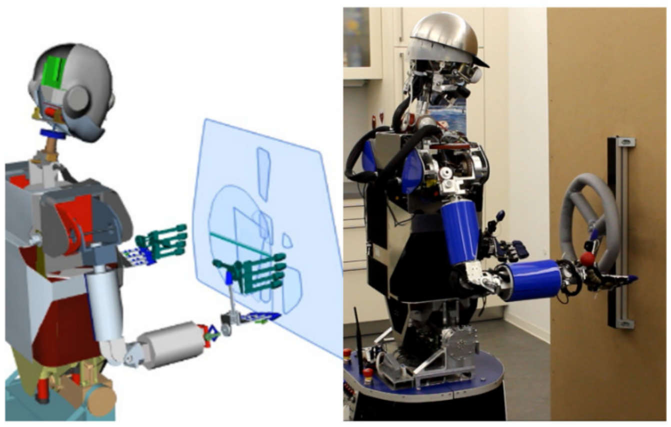


Figure 2.3: Affordance hypotheses for planning according to Kaiser et al. Figure on the left shows the perception pipeline for the affordance hypotheses generation based on geometric primitives (e.g. planar, cylindrical). Figure on the right illustrates a manipulation scenario whit the surface primitives used for affordance prediction. Images taken from [89] and [91] respectively

An affordance type is predicted by considering a set of rules which take into account the area or extension, orientation and reachability of a primitive surface. For instance, a planar surface that extends sufficiently is considered to afford *support*; or a cylindrical surface with or radius between a specific range of values and within certain distance affords *grasp* or *hold*. Using such rule-based system and robot kinematic constraints a set of stability maps can be obtained, these maps encode the available affordances for the robot within its actual workspace. In [91] the rule-based detection of affordances is replaced by sigmoid activation functions which encode how certain the system is about the perception of the affordances. The parameters of these functions include the extension and orientation of the primitive surfaces. Operators that take as input the activations of sigmoid functions and the parameters of the end-effector pose are proposed to compute certainties for manipulation planning. In more recent work [92], the affordance certainty functions are used in so-called *affordance belief functions* which are Dempster-Shafer belief functions [93] in terms of the perceived attributes of primitive surfaces and end-effector poses. With this extension, the proposed system allows for multi-contact whole-body pose sequence planning. Although no information is given regarding the computation times of surface primitives, one they have been computed the authors report an average of 578 ms per scenario to produce hypotheses when only *supportability* affordances are considered.

2.4.2 Grasping affordances

For robots to be capable of manipulating objects in the environment, they need to be able to grasp those objects reliably. Motivated by this, a big amount of research has focused on detecting and learning *grasping* affordances for example for manipulation or pick and place applications. Typically, works studying *grasping* have tackled the problem based on two general approaches: a data-driven and an analytical. Analytical approaches [94] compute grasp poses and finger placements assuming prior information about object class or category, full known geometry or pre-computed physical properties such as material or mass; thus, they are further away from the affordance perception concept. On the other hand, data-driven approaches [95] that study grasp synthesis for unknown or familiar objects are closer to the affordance perception and learning problems.

Earlier works such as Sweeney and Grupen [96] presented a hierarchical, statistical, generative model to represent hand and finger poses learned via teleoperated human demonstrations.

Grasp affordances are defined as the joint distribution over position and orientation of the hand relative to the object, this distribution is conditioned on objects' location in 3D and visual appearance (i.e. image second moment). Stark et al. [97] presented a system that learns finger contact points from observations of humans manipulating objects. The system leverages local geometric features (k-Adjacent segments) and SIFT features to build a codebook of objects' contact points. Though the robot fails to grasp the objects successfully, the experiments show that it is able to predict the hand pose necessary for grasping similar objects correctly.

Learning to grasp based on trial and error was also explored in early approaches by Montesano and Lopes [26], who proposed an algorithm to learn local visual descriptors of good grasping points for a humanoid robot. The system applied a large bank of 2D filters (e.g. Gaussian, Laplacian, Sobel, etc.) to the images of the target object, learning the probabilities of an object-part affording grasp via a Bayesian Network. In [98–100] a method to generate grasp hypothesis densities is developed. The method allows to generalise over partly-similar objects by learning a dictionary of object parts, the perception of these object parts is based on 3D segments encoding colour and edge information; features that are matched against new objects using an alignment error metric (ICP-like).

Song et al. [101] proposed a Bayesian Network model for learning task constraints in grasp selection. The model links the semantic requirements of manipulation tasks (hand-over, pouring, tool-use) to the continuous feature space of the objects and grasp actions. Learning is performed with a simulation-based grasp planner that generates a set of hand-specific, stable grasp hypotheses on a range of objects (25 models in pointclouds), grasps are then labelled by a teacher with the suitable manipulation task(s). The robot is able to infer the intended task of a human demonstration, choose the object that affords this task, and select the best grasp action to fulfil the task requirements. The approach is then extended [102] to include a more challenging simulation: a larger dataset (e.g. more objects), one more task (dish-washing), and different embodiments (e.g. two hand designs). Further extensions [103] allow a real-world setup by considering a markerless, vision-based 3D hand and object tracking system to extract parameters of human hand motion in interaction with objects. In [104] the model is extended to allow a robot to learn from humans demonstration despite the differences in their embodiments.

Kroemer and Peters [105] proposed a framework that represents manipulation tasks in a modular and hierarchical manner. By incorporating notions of the affordances of the objects (e.g.

pushable, rollable, etc.), a system adapts its grasps type depending on the subsequent actions in a plan. In [106], Kroemer and colleagues focus on learning a mapping from object-part geometric features to specific actions such as grasping and pouring. Object-part geometries are encoded as a non-parametric representation of surfaces (e.g. a weighted Gaussian distribution on each of the 3D points), and similarities are computed via a non-parametric surface kernel: the normalized inner product of the two surface distribution functions. With this approach, a robot is able to map object sub-part features to specific motor primitives.

Dan and Allen in [107, 108], learned a mapping from object geometries to semantic grasps (i.e. constrained grasps), this allows to generate grasp poses for new objects (total of 4); which is done by matching the recovered full models against a database of grasp/object pairs using shape context features [109]. In [110] grasping constraints are given by the task of pile clearing, where a robot needs to perform the grasp with the lowest risk of disturbing the pile. Objects are represented with the spatial relationship between 3D facets that are estimated using depth discontinuities and surface normal vectors.

Enveloping grasp affordances and antipodal grasps are studied in by Pas and Platt [111, 112]. A method based on fitting geometrical shapes to pointcloud regions is used to generate grasp hypotheses. By using curvature, normals and quadric surfaces fitted to pointcloud patches[111], and then HOG features on pointcloud 2D projections [112], a robot is able to grasp objects in cluttered scenarios with high rates of success. Parallel and suction grasp affordance proposals are considered by Zeng et al. [113] in a Pick and Place scenario, where affordance hypotheses are seen as a probability map encoding the confidence score of a pixel in the image affording a particular type of grasp. The method, illustrated in Figure 2.4, uses deep convolutional neural networks (CNNs) to learn the visual features that make an image region graspable or suctionable; these CNNs are trained with 1837 pixel-wise annotated images which are rotated 16 times to allow the prediction of grasps at 16 different orientation. Once the object has been grasped the system used a two-stream CNN to classify the objects and place them in the right container.

2.4.3 Affordances from observation

The works described in this subsection also study affordances for robotic applications; however, either the affordance *categories* are provided via labelled data in images or pointclouds, or affordances are learned by observing others (i.e. humans) interacting with the environment.

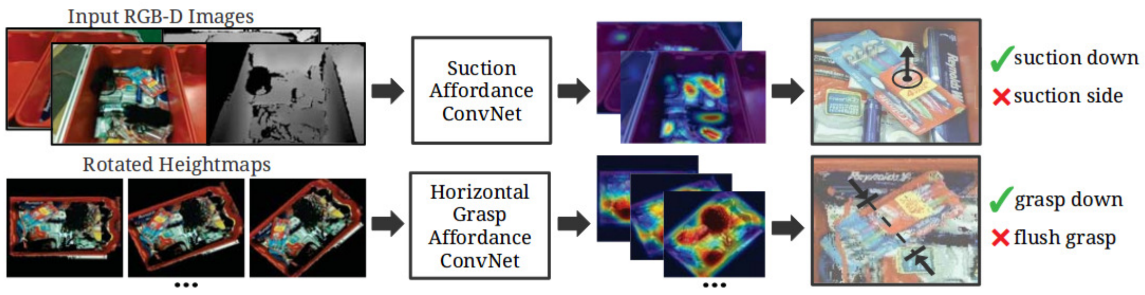


Figure 2.4: Pixel-wise *grasping* affordance prediction based on visual features learned by two CNNs. The output of the system is a confidence score, xyz coordinates of pixel, width and orientation of the end-effector. Image taken from [113]

Generally speaking, these approaches are concerned about perceiving the properties in the environment that make it useful for others. Perceiving affordances in that way allows for a better understanding of humans actions in the environment or to learn what the environment affords to humans. A robotics field that has benefited from this approach to the affordance concept is Human-Robot Interaction.

The work of Saponaro et al. [114] uses the statistical model Montesano [26] to perform human gesture recognition (i.e. tap, push, grasp), which allows a robot to predict human actions in a collaborative object manipulation scenario. The system tracks the 3D hand position on RGB-D video with gesture annotations; whereas the objects are tracked using shape features of colour segmented blobs. Pandey and Alami [115, 116] proposed to encode perspective and effort in an *affordance graph*, where the idea is to model what an agent affords to do with an object (e.g. touch, pick, putOnto and putInto), and what an agent affords to do for another agent (e.g. make accessible, show, give and hide). The approach takes into account reachability, visibility and effort while keeping track of human poses. Objects in the environment are identified by a tag-based stereo vision system.

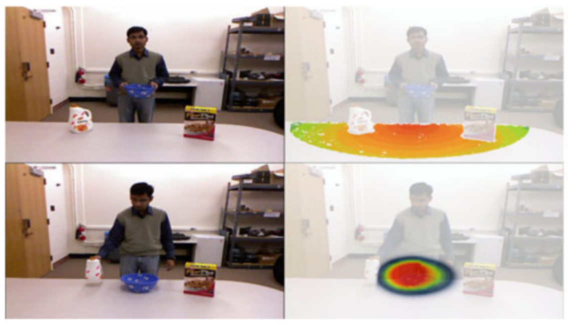
Human-robot interactions are also considered in the work of Chan et al. [117], where a method is presented to determine proper grasp configurations for object handover. The system builds a knowledge base from demonstrations of humans using and handing-over different objects such as knives, screwdrivers, etc.. Affordances representing transport, slide, screw, cut and grasping-point are learned by clustering (k-means) features that encode the relative position and orientation of the observed objects and hand contact points. In order to classify new objects (based on functionality) and determine the grasp configuration, a nearest-neighbour search is performed on the knowledge base.

In [118], *social* affordances are learned by a robot observing humans interacting with other humans on RGB-D video. Affordances are regarded as action possibilities that follow basic social norms (e.g. expected motion or pose); thus, the perception part of the approach relies on human pose detection and tracking. With the proposed spatio-temporal graph model, the robot is able to show human-like behaviours in human-robot interaction scenarios such as *waving, shaking hands*, etc.

Pieropan et al. learn object functional classes (e.g. tool, ingredient, support, container) in [119] and pairwise object relationships in [120]. These categories are learned from observations of humans in a food preparation scenario. The approach generates object hypotheses based on a connected components algorithm [121] that finds segments with colour and depth disparities relative to the table plane. Objects are tracked over time, and their functional categories are learned by tracking their spatio-temporal relationship with the human hands and other objects. In [122], the authors encode the spatio-temporal relationships in the so-called Functional Object-Oriented Network (FOON), which is learned by observing object state changes and human manipulations with the objects on instructional cooking videos. The proposed algorithm builds a graph (i.e. a network) encoding the inter-object relations, motions and manipulation goal (cooking).

Human activities are also studied in by Koppula and colleagues in [123–127], who proposed a graphical model that learns spatio-temporal relationships between objects and humans performing activities with those objects, an example of their approach is shown in Figure 2.5. The system learns so-called *semantic* affordances (e.g. movable, stirrable, pourable, etc.), *spatial* affordances (i.e. object contact points) and *temporal* affordances (i.e. object trajectories) which improves performance on activity recognition[124, 125]. The approach is later developed to allow for anticipation of human activities [123, 126, 127] which showed beneficial in a human-robot interaction scenario. The system is called Anticipatory Temporal Conditional Random Field (ATCRF), and is trained with a manually-annotated RGB-D dataset of humans poses and object bounding boxes.

Another line of research has followed the approach where affordance categories are provided to the system in the form of labelled pointclouds or images. In this way, the problem becomes



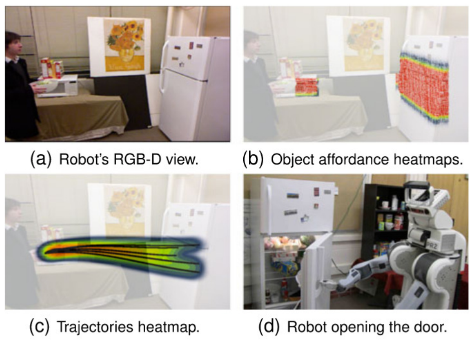

(a) Robot's RGB-D view.
(b) Object affordance heatmaps.
(c) Trajectories heatmap.
(d) Robot opening the door.

Figure 2.5: Image on the left shows affordance heatmaps learned by human observation in the approach proposed by Koppula et al., figure on the top-left shows *placeability* affordance, figure on the bottom-left shows *pourability*. Image on the right shows a scenario example for reactive robot response through anticipation. Images taken from [127]

that of learning a representation or mapping that is able to generalise to previously unseen data. An early example of this is the work of Aldoma et al. [128], who proposed the concept of 0-order affordance. This concept refers to a *hidden* affordance (e.g. rollable, containment, stackable-onto and sittable) that objects are known to have but not in the current pose. Thus, the idea is to first recognise the object and estimate its current 6D pose; then, the transformation that achieves a “stable” pose is computed, which is later used to predict the objects’ affordance. Affordance predictions are shown on 20 real objects (pointclouds) based on the Princeton Shape Benchmark [129]. A manually annotated dataset is generated to test various classifiers (e.g. Random Forest, SVM and Boosting) using combinations of 3D shape descriptors: PFH[130], Spin Images [131], SHOT [132]).

Kim and Sukhatme [133], introduce an algorithm to enhance object segmentation and reduce manipulation uncertainty for a PR2 robot. Geometric features such as planarity, normal vectors, centroid, etc. computed from point cloud segments are used to train a logistic regression that allows predicting *pushable, liftable* and *graspable* affordances. The system is able to use multiple views to improve object segmentation based on the previous geometric features and the object’s colour.

A method to learn three functional classes, e.g. drinking vessel, table and sittable based on physics simulations is presented by Hinkle and Olson[134]. A weighted nearest neighbour classifier is trained on 200 CAD models by counting the number of spheres that remain stable on top of the objects after being dropped from a certain height. Results of the algorithm are shown on pre-segmented real RGB-D scenes. With a similar approach, Yu et al.[135] use a physics-based simulation to learn *containability* affordance on 3D CAD models. Fluid mechanics methods allow the system to predict the best direction to transfer the contained liquid while avoiding spillages. Authors show prediction results on real data using RGB-D scans of objects such as bowls, mugs and spoons.

Tool-part or object-part affordances (i.e. segmentation) have also been studied in robotics. Myers et al. [136] proposed two methods for associating local shape and geometry information from 105 objects to affordances such as grasp, scoop, support, pound and cut. The proposed methods train with shape features such as surface normals and principal curvatures computed from RGB-D images with pixel-wise labels of the corresponding affordances. In [137], tool-part affordances are used to find substitute tools to accomplish a specified task. The method relies on pointcloud geometry (i.e. superquadric fitting) to classify a previously unseen object by matching against a database of manually-defined object-task pairs. The method is then developed [138] to consider a more extensive collection of synthetic objects (e.g. 70) and simulations of the tasks that improve the substitute tool predictions. More recently, methods have been proposed [139] to leverage Markov Random Fields in tool-part affordance segmentation problem.

Finally, just as in many other fields in Robotics research, deep learning methods have been recently introduced to tackle the problem of object-part affordance segmentation. These approaches leverage the ability of deep CNNs to learn features from annotated data in fully-supervised [140, 141] or weakly-supervised [142] manners. Further development of such affordance segmentation approaches has included object detection [143] with larger datasets of RGB images, as well as object detection and recognition using multi-stream deep neural networks [144] to increase the affordance detection performance. Deep CNNs were also used for scene functional-region prediction by Ye et al. [145], where a two-stage CNN is trained on manually-labelled bounding boxes from kitchen scenes of the SUN dataset [146]. The approach leverages object detection and recognition architectures first to generate thousands of region proposals that are then classified according to functional types such as *sittable, turn on/off* and various types of grasp. The network is trained with 250K annotated images, though only one class (i.e. affordance) per region is assumed.

2.5 Affordances in Computer Vision

This section presents an overview of works that approach the problem of affordance detection and learning from a computer vision perspective. In other words, these works primarily use the concept of affordances in order to boost performance for human action/activity recognition, object classification or recognition, improve semantic scene understanding, or build a knowledge base that includes objects functionality attributes which are linked to visual appearance attributes.

Functional regions in objects are studied by Desai and Ramanan [147], who compared several models on the RGB images of the PASCAL VOC dataset [148]. The models compared include 'Blind' methods trained with object spatial priors and functional region priors, Bottom-up methods based on colour, spatial position, texture features and line features, and object-specific Top-down models based on HOG features. The affordances studied have to do with objects that can be grasped, sat-on and looked-at. They showed that models that explicitly consider object shape and structure are worthy of further exploration. Object affordances in human manipulation tasks are studied by Zhu et al.[149], where a spatio-temporal graph representation is proposed to predict substitute tools based on the task to accomplish (e.g. cracking nuts) The model is learned by observing RGB-D video of human performing the task. The method used 3D mesh and material information of tools as well as object, hand and human-pose tracking to learn the spatio-temporal relations among these elements during manipulation tasks.

Object classification and recognition have also been studied using affordance “cues”. Srikantha and Gall [150], present a weakly supervised learning algorithm for object classification in human activities. Affordances are identified as the spatio-temporal relations between objects and human hands/joints. The system is trained and tested in RGB-D and RGB videos where a heavy detection and tracking pipeline (e.g. two tracking algorithms running in parallel) accounts for appearance and shape features. In [151] a multi-stream CNN architecture is proposed to perform object recognition by fusing appearance features with spatio-temporal relations between objects and human hands. A combination of 13 human interactions with 14 objects is considered in their study, which includes affordances such as *grasping, pushing, rotating, hammering, squeezing* amongst others. In [152] an encoder-decoder architecture is used to learn an embedding from 11,505 videos of people interacting with everyday objects. The proposed method allows to predict objects' interaction region and action label for same-object images, the prediction pipeline is shown in Figure 2.6.

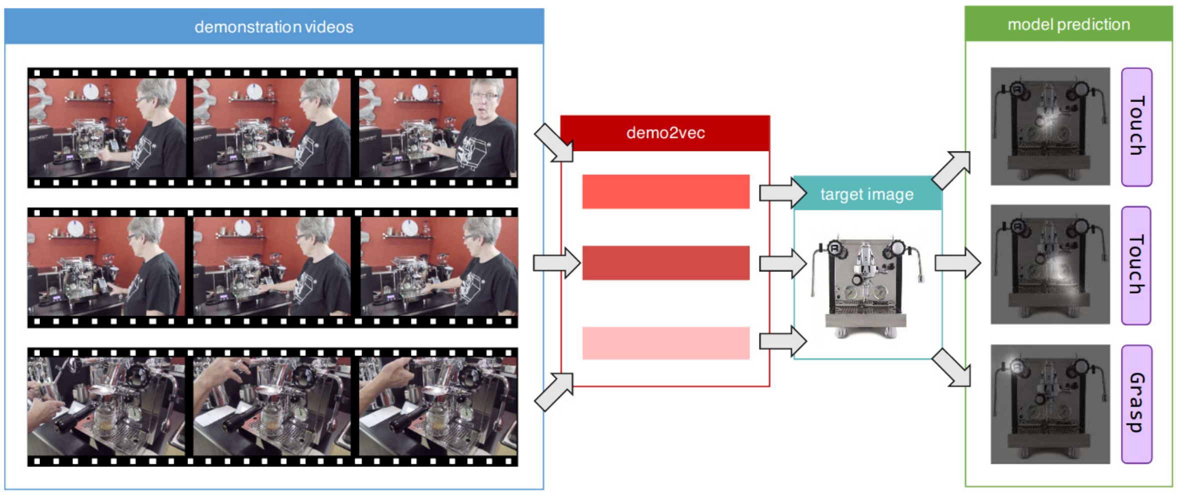


Figure 2.6: In the method proposed by Fang et al., video demonstrations are tracked and embedded into a feature vector (demo2vec). This vector is then used to predict affordances (i.e. region and interaction label) over an image of the same object used for demonstrations. Image taken from [152]

Objects and their relations with humans are studied by Jain et al. [153], who proposed the combination of spatio-temporal graphs and recurrent neural networks to model human-object interactions in RGB video sequences. The model is trained and tested on the CAD-120 dataset [124], where the affordance annotations include objects that are *movable, pourable, containable, placeable*, amongst others (12 in total). The hybrid approach shows better performance on the same dataset when compared with earlier approaches [124].

Many computer vision approaches have studied what objects afford for humans by observing video of humans interacting with objects in their surroundings. In contrast, the work of Grabner et al. [154] proposed to hallucinate humans interacting with objects in a scene to detect *sitting* affordances. The proposed method computes distance fields using 3D triangular meshes of the objects and detecting the degree of intersection between triangles. The performance of *sittable* object detection is further improved by considering an appearance-based chair detector. *Sitting* and *reaching* poses for indoor environments are also studied by Gupta et al.[155], where a method based on the correlation between volumetric representations of the scene and human pose (e.g. cuboids) is used to predict human poses on RGB images. In [156, 157] the problem is reframed to learn scene functional regions and object locations (e.g. semantic segmentation) from images of people interacting with the environment. Human pose hallucinations and scene geometry are used to determine object affordances (i.e. placement/location relative to human) in synthetic indoor scenes [158, 159]. In [160] the system is applied to object labelling for 3D scenes (pointclouds).

The works that address *sitting* affordance are further extended to include comfort and social goals. For instance, Zhu et al. [161], where sitting location preference and pose are learned from RGB-D videos showing how humans sit in different environments; then, the system's affordance hypotheses are further refined and validated using physics-based simulations. In [162], a data-driven method learns a pose probability distribution from 100M RGB images of humans interacting with indoor environments (e.g. sitting, standing). The method capitalises on face and person detectors to train a variational autoencoder[163], which allows generating human poses in previously unseen scenarios.

Social norms are considered for agent interactions with the environment in the work of Chuang et al. [164], who proposed a knowledge base (i.e. dataset) with action-object pairs (e.g. sit-stool, grasp-bag, run-floor) and consequence explanation annotations (e.g. One cannot take other people's property since One could go to jail). This knowledge base is used with a GNN (Graph Neural Networks [165]) to "learn to act properly". The GNN leverages state-of-the-art object segmentation methods to build a graph encoding spatial relations and the social consequences of executing actions on objects.

Locations suitable for human whole-body affordances have been studied by Piyathilaka and Kodagoda [166, 167], who introduced the concept of affordance map to encode locations in a 3D scene where a human could stand or sit. The method is based on collisions and distance between a human skeleton (15 joints) and a voxel representation of the scene. In [168] a joint distribution is learned over function (semantic label), geometry (cuboids) and appearance features, this allows to detect objects that afford *sitting, storing* and *sleeping*. The work is further developed to learn affordance maps that consider RGB-D video of humans interacting with the environment [169] and simulated human trajectories in indoor environments [170]. An example of the affordance predictions by Qi et al. is shown in Figure 2.7.

Using RGB-D images of indoor scenes, Roy and Todorovic [171] perform pixel-wise segmentation for *human* affordances. The approach uses multiple CNNs to extract mid-level features or "cues" namely depth, surface normals and semantic segmentation. These cues are then used as input to another CNN which classifies the segmented regions as walkable, sittable, lyable,

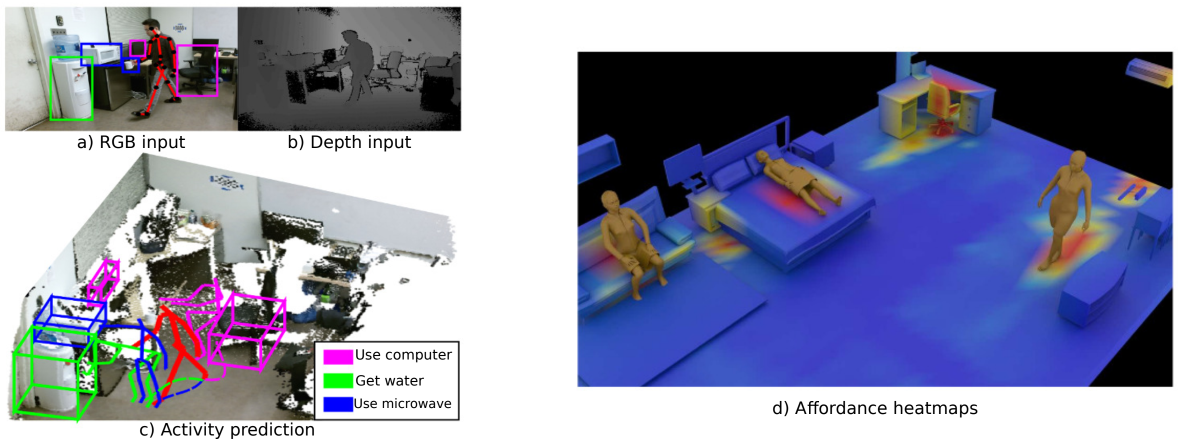


Figure 2.7: Images on the left (a-c) show human affordance predictions learned by observing people on RGB-D video. Affordances are regarded as the most likely human pose in a specific activity. Image on the left(d) shows affordance maps learned by simulating human activities in synthetic indoor scenarios. These maps are obtained by accumulating human positions across different activities. Images taken from [169] and [170] respectively.

reachable or movable. The system is trained and evaluated with the NYUv2 [172] dataset with annotations of the features mentioned above in addition to planes and surface heights. Figure 2.8 depicts a diagram summarising the approach for *human* affordance segmentation.

Lastly, there have been efforts to incorporate affordance information for vision systems in the form of a knowledge base that allows making higher-level inferences. An example of this is the work of Zhu et al. [173], who build a knowledge base representation for object affordances via Markov Logic Network. The model learns relationships between objects' affordances (a total of 14 e.g. grasp, push, ride, sit on, etc.), visual attributes (texture, shape, material), physical attributes (size, weight) and semantics. The method allows to learn rules such as "objects that look metal are less likely to be feed-able". Chao et al., [174] propose to link object classes with actions classes. Three methods are proposed to fill a matrix representing the relations between 20 objects and 957 action classes: text mining (e.g. noun-verb frequency on Google Syntactic N-Grams), visual mining (e.g. visual consistency on Google Image search) and collaborative filtering (WordNet similarities). Patterson and Hays [175] provide an attribute database with affordance annotations (e.g. *playing, cooking* for objects appearing in the SUN[146] dataset. Varadarajan and Vincze introduced the Affordance Network (AfNet), a database with knowledge ontologies of over 250 common household objects. The database is formed by a set of visual features that the authors divide into *structural* affordances and *material* affordances. One set accounts for the features

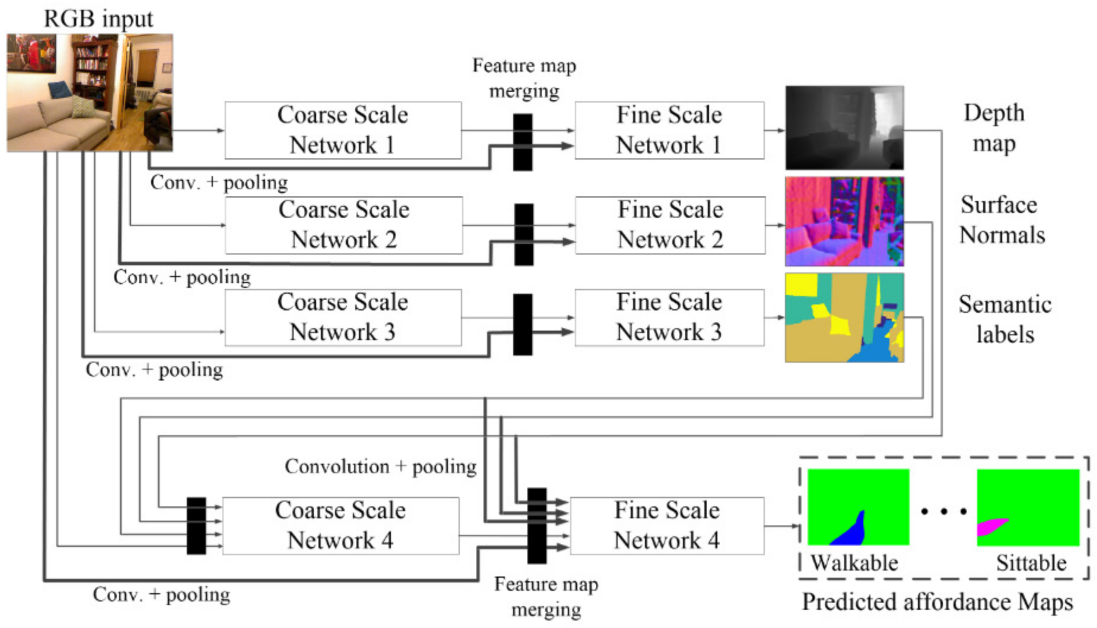


Figure 2.8: Multi-scale CNN approach for *human* affordance prediction in indoor scenes by Roy and Todorovic. Various CNNs are employed to learn and merge cues such as depth, normals, semantics, etc. at different levels. The prediction results are presented as pixel-wise labels of affordances such as *sittable, walkable* and *lyable*. Image taken from [171]

defining the structure of objects, the other accounts for the visual features that describe the material of the objects.

2.6 Other interactions

Interactions between entities (e.g. objects or agents) in the environment have also been actively investigated in the field of computer graphics. Many works in this research area have been concerned with proving tools and methods that allow the design and simulation of artefacts in a scene. In recent years, an increasing number of approaches that consider functional information about the objects interacting in a scene have been proposed [176–178]. While many of these approaches assume precise information about locations, pose or detailed geometry of the objects or entities in 3D space, they serve as a valuable source of inspiration for methods that study the problem of affordances.

Works in computer graphics typically investigate shape and object functionality based on the geometry of the artifacts in a scene, there have been approaches that predict the functionality class of an entity (e.g. 3D object model) by studying how an object reacts to rigid motions [179] or the spatial configuration of the object parts [180]; however, given that these approaches do not consider interactions in their methods they are further away from the concept of affordance. Works have also tackled object-part functionality; these are analogous to methods in robotics that learn tool-part affordances [136–138]. In other words, given a collection of labelled object-segments, they use geometric features to learn and predict functional regions in novel objects. An example of such approaches is [181], where part context (e.g. relation with other parts) is considered in addition to geometric features in order to build a graph representation of the parts' functionality.

On the other hand, works that do consider interactions have proposed a number of rules, features and representations to encode the spatial relationship of the elements in the scene. Earlier work used spatial arrangements to enable users to *quickly* lay out complex scenes with multiple objects using interior design guidelines [182], or placement constraints (e.g. proximity, support), pseudo-physics and semantics [183]. Fisher et al. introduced methods to learn, from multiple scene arrangement examples, the spatial relationships of furniture based on a co-occurrence model [184] or based on a graph encoding semantic information [185].

Richer representations, allowing to capture more complex interactions accurately, can also be found in the computer graphics literature. An important example of these approaches is the work of Zhao et al. [186], where a method for retrieval and indexing of 3D scenes is proposed. The method is based on topological and geometric features of the Bisector Surface between two interacting entities, i.e. the Interaction Bisector Surface (IBS). Authors proposed a similarity measure which uses the IBS mesh data via the computation of Betti numbers [187], Point Feature Histograms (PFH), angles between normal vectors and the vertical as well as the distance between the objects. Inspired by works using graph kernels [185], a hierarchy (i.e. graph) is built on the basis of pairwise similarities in local and extended regions. Then, given a target input scene and a scene database, the method allows to retrieve another scene with a similar object arrangement. The IBS was later used with Interaction Regions [188] to further characterize the interactions between the objects' fine-grained geometry.

In the approach of Zhao et al. [189], IBS features combined with the so-called space coverage feature (SCF) allow to synthesize scenes using a template matching algorithm with a large database of 3D models. Figure 2.9 visualises the general idea behind their approach; details are presented below.

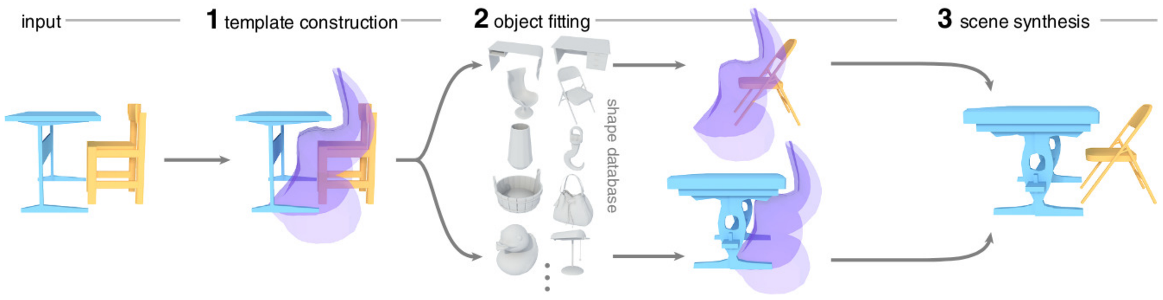


Figure 2.9: Example-based approach of Zhao et al. to synthesise scenes with similar interactions based on the Interaction Bisector Surface. Image taken from [189]

Using a template interaction, the IBS is computed among all objects in the scene. Features are computed over 200 points on the IBS similarly to [186]. Sampling is based on a weight assigned to every triangle T on the IBS mesh; specifically, weights are defined by $W(T) = W_{\text{area}}(T) \times W_{\text{scene-distance}}(T) \times W_{\text{angle}}(T)$, where $W_{\text{area}}(T)$ is the area of triangle T and where

$$W_{\text{angle}}(T) = \begin{cases} 1 & if\, \alpha < 45^\circ \\ 0 & otherwise \end{cases}$$

$$W_{\text{scene-distance}} = (1 - \frac{2 \times d}{d_{\text{diag}}})^n$$

where, α is the angle between the normal vector of triangle T and a vector pointing towards the object, d is the distance between the centre of triangle T and the object, d_{diag} is the diagonal of the bounding box of the whole scene. n set to 20 empirically. Then, a set of features is computed over the points sampled on the IBS: $f_{\text{dis}}, f_{\text{dir}}, f_{\text{scf}}$. These features are: the shortest distance from a point on the IBS to the object, the direction at which the object is located relative the point on IBS, and the space coverage feature, respectively. The space coverage feature is computed by first fitting a sphere centred in every IBS sampled point; then, $n \times n$ points are uniformly sampled in this sphere and rays are cast from the centre of the sphere towards each one of the sampled points. For every sphere, a volume diameter function is computed with

$$F_{vdf}(i,j) = \left\{ \frac{d^{min} + e}{d(i,j) + e} \quad | \quad 0 \le i,j \le n-1 \right\}$$

where n is set to 30 empirically, i,j are ray indices along the sphere's longitude and latitude directions, d^{min} is the minimum distance among all rays, e is an offset whose value is set to the mean of all distances. F_{vdf} is normalised to [0,1] and used to compute the space coverage feature (SFC), which is the spectrum power of the volume diameter function expressed in terms of the spherical harmonics at 5 levels. Formally, $SFC(F_{vdf}) = \{a_0, a_1, ..., a_n\}$ where

$$a_l = | \sum_{|m \leq l|} a_{l,m} Y_l^m |_2 = \sqrt{\sum_{|m \leq l|} (a_{l,m})^2}$$

where $a_{l,m}$ are the spherical harmonic coefficients at frequency level l.

Using these features (distance, direction and SCF) a similarity function is used to evaluate how well a novel object fits the template. The similarity function is given by:

$$S_{\text{final}} = (1 - d_{dist})(1 - d_{dir})(1 - d_{scf})$$

where

$$d_{\text{dis}} = \frac{1}{N} \sum_{j=1}^{N_i} \frac{\left| f_{\text{dis}}^j - f'^j_{\text{dis}} \right|}{\alpha_{\text{dis}}}$$

$$d_{\text{dir}} = \frac{1}{N} \sum_{j=1}^{N_i} \mathbb{1}\left(\angle (f_{\text{dir}}^j, f'^j_{\text{dir}}) \leq \alpha_{\text{dir}} \right)$$

$$d_{\text{scf}} = \frac{1}{N} \sum_{j=1}^{N_i} \frac{\left| f_{\text{scf}}^j - f'^j_{\text{scf}} \right|_2}{\alpha_{\text{scf}}}$$

where $N_i = 200$, i.e. the number of points sampled over the IBS, $(f_{\text{dis}}, f_{\text{dir}}, f_{\text{scf}})$ are the features of point p_j on the template, $\left(f'_{\text{dis}}, f'_{\text{dir}}, f'_{\text{scf}}\right)$ are the features of the novel object. $\mathbb{1}(\mathscr{X})$ is the indicator function which returns 1 if $\mathscr{X}$ is true and 0 otherwise, $\alpha_{dir} = \frac{\pi}{2}$ is a manually defined threshold within which orientations are considered similar, and $\alpha_{dis}, \alpha_{scf}$ are normalisation parameters to bound d_{dis} and d_{scf} in the range [0,1]. In order to decide whether a novel object would generate a similar interaction, space coverage features are computed in regions of interest (ROIs) surrounding the novel object. These ROIs are generated by sliding a ”window“ across the empty space around the novel object and comparing the features against the template's mean feature. This process allows to pre-select good ROIs that are later used to effectively match against the template using geometric hashing [190]. The match is further refined by treating the distance feature as a distance field and updating according to the gradient. Finally, additional criteria such as contact, collision and height constraints are taken into account to synthesise a new scene. Although some of the computations (e.g. features) are performed offline, the process is iteratively repeated for every object that needs to be added to the scene.

Dynamic functionalities have also been studied, where entities are tracked over time in order to capture the abstract dynamic characteristics of the interactions. An example of such approaches is Interaction Landscapes [191], which represents the interacting objects as sets of particles of which motion is tracked by treating the surrounding space as sensor regions. These sensor regions detect particles passing by; thus, the representation of the interaction is obtained in the form of vector fields that encode the position of the particles over time. The distance between two interaction landscapes is computed by comparing histograms that encode the attributes of the vector fields inspired by fluid dynamics (e.g. vorticity, dilatation, shear strain rate).

Another type of dynamic interactions studied in computer graphics are agent-environment interactions. In SceneGrok [192], a method to predict *action maps* in indoor scenes is presented. The approach trains a classifier for various human activities (e.g. using-laptop, watching-tv) that are observed in RGB-D video. The method reconstructs a mesh representation of the scene, tracks the human skeleton and the mesh segments in contact with skeleton joints. Clustering geometric features of the mesh segments in contact with human joints allows to learn a segment dictionary; this dictionary is then used to predict the likelihood of an action for a previously unseen scene segment. In an agent-centric approach, Kim et al. [193] proposed a method to predict the pose that a human would need to adopt in order to use an object (e.g. bicycle, chair). In PiGraphs [194], human poses are used to learn to arrange objects according to their relationship with human usage. Interestingly, features on the IBS have also been used to represent agent-object interactions recently; tracking such features over time allows for the retrieval and classification of motion trajectories [195].

2.7 Summary

In this chapter have been review the different definitions and interpretations of the affordance concept, which included from the creation of the concept by J.J. Gibson to the computational formalisms commonly used in robotic applications. Neuroscience studies that have provided evidence supporting the affordance theory have also been presented, evidence that has served as inspiration for the big body of research in affordance learning. Then, an overview of the previous works that have addressed the problem of affordance perception and learning was presented; which included approaches in robotics, computer vision as well related work in computer graphics.

One of the most contentious aspects of the affordance concept is the role of *direct* perception. Earlier works in robotics [43] interpreted this idea as making use of low-level visual information (e.g. optic flow) for robot control (e.g. navigation). Other approaches in robotics, such as those in the developmental robotics field, have treated the *direct* perception of affordances as learning a mapping between percepts and actions available to a robot. These approaches are inspired by learning stages observed in human infants, but also take inspiration in visuomotor neurons discovered in neuroscience studies. The use of such *direct* link between perception and action has allowed robotics systems to build a representation suitable for planning, multiple object manipulation scenarios as well as learning tool usage.

Another important amount of effort has been made regarding *grasping* affordances and *human* affordances (e.g. *sitting*). The former is a fundamental skill for robots that need to interact with the world; the latter is a very useful ability for robots that do not need to manipulate objects but rather assist humans in their daily life. One of the outstanding characteristics of many of these approaches is their ability to leverage local geometric cues without the need to recognise the object categories or labels.

In contrast, a substantial number of approaches have tackled the problem of learning object (or tool) affordances from labelled RGB or RGB-D data. In fact, this has been an increasingly popular approach within the last couple of years, primarily motivated by the availability of large collections of annotated data or knowledge bases. The *direct* perception of affordances has been regarded as the possibility of learning a representation directly from the data via data-driven algorithms such as those based on neural networks, e.g. deep learning.

In computer vision research, many of the current methods that incorporate the concept of affordances have focused on the relation of objects relative to human hands or pose (i.e. human context). In this sense, affordance cues have shown to improve on activity recognition and prediction, object recognition and semantic scene understanding. An alternative that can also be found in the computer vision literature is the detection and tracking of "blobs" or entities in video sequences without performing recognition. These approaches have achieved remarkable results in the discovery of functional categories by observing the spatial relationships among "blobs" (i.e. objects) through time.

Another area that has been actively studying interactions amongst agents, objects and the environment is computer graphics. Works in this area leverage geometrical information of synthetic objects and their spatial configurations to perform functional analysis of virtual scenes. These approaches take advantage of the relations between entities in the scene to characterise and more accurately synthesise novel environments.

2.8 Criticism

The affordance concept has undoubtedly proven to be beneficial for multiple tasks in robotics and computer vision. However, as reviewed by Horton [11], the vague definition of the concept that Gibson provided led to many interpretations and debate regarding the true nature of the term. To a certain extent, roboticists seem to agree that affordances should be a relation between two (or more) interacting entities. Sahin et al. [25] enriched such definition by proposing that these relations can be viewed from three perspectives: the agent, the environment and an outside observer.

Many approaches in robotics learn these relations as a symbolic representation that enable a robotic system to plan its actions. Thus, affordances become agent internal representations, which is practical for the purposes of building plans but clearly goes against the ecological approach proposed by Gibson. Moreover, while many of these approaches allow for human-inspired learning stages, the applications are limited to a small set of objects and affordances. It is not clear how the models would apply for novel objects and novel realistic environments. A similar dilemma is faced by methods that solely focus on one type of interaction, i.e. *grasping* or *human* affordances. Methods addressing these affordances have achieved remarkable results when facing novel realistic scenarios, yet the question remains open about the generalisation of the approaches for other types of interaction or scenarios that do not require manipulation.

Within the last couple of years a notorious trend has been observed for learning object or tool affordances, that is exploitation of large collections of labelled imagery. One important challenge faced by these methods is that they usually need to detect (and even recognise) object instances in the environment. As reviewed by [15], these methods inherit many problems faced by computer vision, problems such as clutter and occlusions, viewpoint and scale variations, intra-class variations and more importantly the multiple-object multiple-label nature of affordances. Whereas deep neural networks have proved to be a powerful tool in this area, generalisation and scalability remain as important challenges. For instance, the models would need new annotated data and an extensive retraining process in order to learn new affordances. Besides, the manually annotated datasets used for training build on the assumption that objects in the environment have a pre-defined set of affordances; making uncertain how an agent would discover new affordances, i.e. the absence of annotated data.

We argue that in order to truly perceive affordances in the environment in a way that is most useful for cognitive agents, there is a need for methods that are agnostic to object categories and free from complex feature representations; methods of a generic nature that allow for the simple yet robust description of multiple affordances. Furthermore, affordance detection methods for robotic systems need to allow for fast computations and be able to generalise to novel scenarios without lengthy and costly training phases. We hypothesise that geometry on its own provides enough information to robustly and generically characterise affordances in a way suitable for robotic perception.

Developing affordance perception systems based only in appearance features with the intention of categorising entities in the world, provides little hope for the generalization capabilities of cognitive autonomous agents. We argue that basing the perception of affordances in 3D geometrical information is a far more promising alternative. A strong case for the geometric characterisation of affordances can be found in the affordance research literature reviewed previously, where has been shown that shape features and spatial relationships have constantly been used as cues for the study of affordances and functional understanding of the environment. The advantage of geometric features over alternative representations, such as texture or colour, is that geometry provides a stronger generalisation power since the geometry of everyday objects strongly dictates the physical interactions that are possible with the environment.

The research presented in this thesis shows that methods based entirely on geometric information are capable of predicting high quality and meaningful affordance locations for realistic environments. In contrast to other works considering geometric information for affordance perception such as [89–92], the approach proposed in this thesis does not build on higher-level geometric primitives nor complex features computed on the environment. Moreover, the general purpose nature of the representation here proposed allows to characterise affordances for *simple* objects such as a mug but also enables the representation of more complex interactions like a human riding a motorcycle. Contrary to methods in computer graphics such as [186, 188, 189, 195], the approach introduced in this thesis takes into account visually perceived information, does not require highly detailed geometries and is straightforward to compute.

CHAPTER 3

THE INTERACTION TENSOR

3.1 Introduction

In the previous chapter has been shown that many approaches in computer vision and robotics have taken advantage of geometric (i.e. shape) descriptions and spatial relationships to characterize interactions or identify functional categories in the environment successfully. Motivated by this and recent approaches in computer graphics for shape functionality, scene indexing and synthesis [186, 189], this chapter presents the affordance representation central to this thesis: The Interaction Tensor.

Starting from a brief review of relevant computational geometry concepts such as Bisector Surface, Voronoi diagrams and convex hull, an overview is given of the key characteristics and computation details of the Bisector Surface between 3D object, namely the Interactions Bisector Surface (IBS). The introduction to computational geometry fundamentals is followed by details on how a simple yet robust affordance representation can be devised from the proposed Interaction Tensor. In short, this is achieved by encoding in the representation information about the expected or most relevant regions that allow the interactions to take place. Details are provided regarding the computation of an affordance descriptor that allows for the real-time detection of interaction opportunities in 3D environments. The speed that the method allows, in addition to its straight-forward computation, align well with Gibson's economical and direct perception of affordances.

3.2 Computational geometry fundamentals

The affordance representation proposed in this thesis takes inspiration from the concept of Interaction Bisector Surface (IBS). This geometrical entity was conceived and has been recently exploited within the computer graphics community, its definition and computation require computational geometry concepts and tools that, for completeness, are briefly introduced in the following subsections

3.2.1 Voronoi diagram

The Voronoi diagram for a set P of n points (also known as generating points) in Euclidean space E^d is the subdivision of the space into n regions (cells) such that all locations within any one region $V(p_i)$ are closer to the generating point p_i than to any other point in P. In two dimensions each Voronoi cell is a convex polygon, in three dimensions a Voronoi cell is a convex polyhedron formed by convex faces (polygons). Formally the Voronoi cell for point p_i is given by:

$$V(p_i) = \{x \quad | \quad \|x - p_i\| \leq \|x - p_j\|, \forall j \neq i\}$$

The lines (or polygons in 3D) delimiting a Voronoi cell are called **ridges** or edges, these are perpendicular bisectors for every pair of points in P. The intersection of the boundary (edges) of three adjacent cells is called a Voronoi vertex. The Voronoi diagram (red polygons in Figure 3.1-b) is formed by set of Voronoi cells for every generating point in P: $V(P) = \{V(p_1), V(p_2), ..., V(p_n)\}$. The Voronoi diagram has a geometric dual structure called the **Delaunay Triangulation**, which partitions the space into triangles such that no point is inside the circumcircle of any triangle i.e. the circumcircle is empty. The relationship between the Delaunay Triangulation and the Voronoi Diagram follows that every ridge in the Voronoi diagram has a corresponding edge in the Delaunay triangulation, and every point in the Delaunay triangulation corresponds with a Voronoi cell. Each triangle in the Delaunay triangulation is associated with one vertex of the Voronoi diagram; the vertex is located at the centre of the circumcircle. These geometric entities are illustrated in Figure 3.1-d.

Multiple algorithms exist for computing the Voronoi diagram; the one used in this thesis is based on the computation of the convex hull [196]. Briefly speaking, the idea proposed by Brown consists in transforming points in d-dimensional Euclidean space into $d+1$ space, constructing the convex hull of the transformed points, and then transform back into d-space. This process can be performed efficiently with the Quickhull algorithm [197] whose implementation is readily available in many programming languages.

The convex hull of a set of points is the smallest convex set that contains all the points [198]. The Quickhull algorithm follows a divide and conquer approach, based on the intuition that most of the points lie in the interior of the hull. Then, the idea is to discard the points that do not lie in this hull as quickly as possible; which can be achieved by iteratively selecting the furthest point from the current convex hull. Figure 3.2 illustrates the idea behind the computation of the Voronoi diagram using the convex hull. Notice that the output of the Quickhull algorithm is a simplicial complex (e.g. triangulation of points).

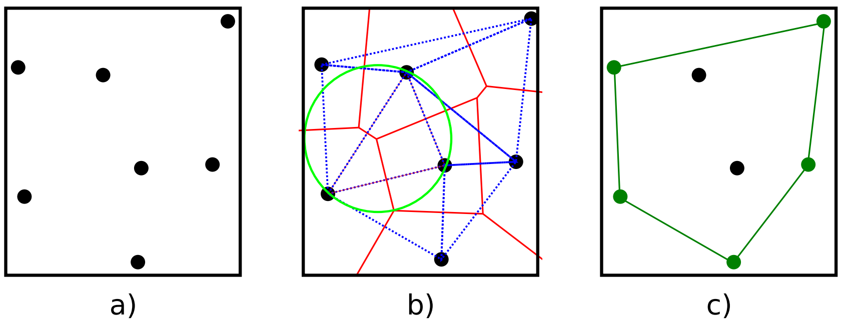


Figure 3.1: Illustration of basic computational geometry fundamentals. a) Set of points in Euclidean 2-space, b) Voronoi diagram with cell ridges highlighted in red, blue triangles are the Delaunay triangulation, and green circle exemplifies an empty circumcircle, c) Convex hull highlighted in green.

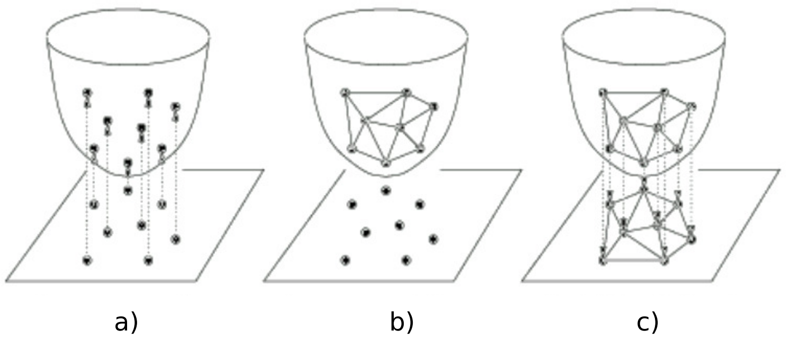


Figure 3.2: Voronoi diagram computation via the convex hull. a) Points are projected onto $(d+1)-space$, b) The convex hull is computed for the transformed points, c) Hull faces are projected back into d-space. The projected regions correspond to Delaunay triangulation; the Voronoi diagram is computed by its duality relationship with the Delaunay triangulation. Adapted from [199]

3.2.2 Bisector surface

The Bisector Surface for any two geometric objects O_1 and O_2 in Euclidean 3-space E^3 is defined as the locus of equidistant points from O_1 and O_2. Generally speaking, O_1 and O_2 can be points, curves, surfaces or solid object instances. If $O_1 = O_2$ is a solid object, the bisector surface is usually called self-bisector, medial axis or skeleton. As a matter of fact, the medial axis (and its transformation) have been widely used in computer vision [200, 201], pattern recognition [202, 203], path planning [204, 205] and mesh generation [206]. Much of the success or popularity of the medial axis is due its ability to abstract the object's shape by producing a discrete graph structure; albeit when computed on a single object it is very sensitive to changes in the object's (or shape) boundary. This issue, however, does not represent a problem when the bisector is computed between two different objects in proximity (i.e. IBS) [186].

Generally, computing the bisector surface is a complex task since it requires to solve a system of non-linear equations [207]; however, it can be approximated by computing the Voronoi diagram for objects O_1 and O_2 in space. Figure 3.3 shows in blue a few examples of bisector surfaces for basic geometrical objects in Euclidean 3-space E^3.

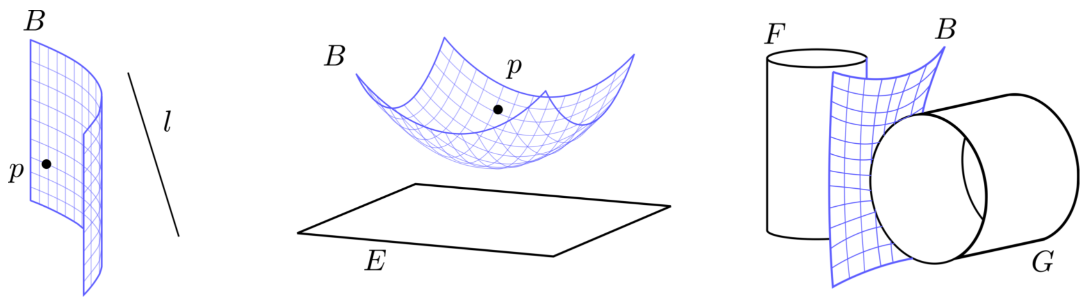


Figure 3.3: Examples of basic bisector surfaces (**B**). From left to right: bisector surface between a point(p) a line(l), bisector surface between a point(p) and a plane (E), bisector surface between two solids (F and G). Adapted from [207].

3.2.3 Interaction Bisector Surface

Introduced by Zhao et al. [186], the Interaction Bisector Surface (IBS) is a generalisation of the Bisector Surface when computed between two or more 3D models in a scene. Formally, given N point sets $P_1, P_2, ..., P_N$ in space where $P_i = \{p^i_1, p^i_2, ...p^i_n\}$, an Interaction Bisector Surface divides the space into N regions with the two properties: 1) Points from the same point set reside only in one region and 2) If a point $q \notin \{P_1 \cup P_2 \cup ...P_N\}$ resides in the same region as P_i, then the

Hausdorff distance between set $\{q\}$ and P_i is shorter than the Hausdorff distance between set $\{q\}$ and P_j, where P_j is any other point set.

Being a generalisation of the Bisector Surface, the IBS can be approximated by computing the Voronoi diagram between objects. In this sense, the IBS is the set of points equidistant from two sets (i.e. objects). After computing the Voronoi diagram over all the points in space, the resulting simplicial complex is processed in order to detect ridges generated by points belonging to different objects, which are the ridges effectively splitting the space into two regions. An illustration of the IBS and its computation between two objects, e.g. a cup and a sink, is shown in Figure 3.4

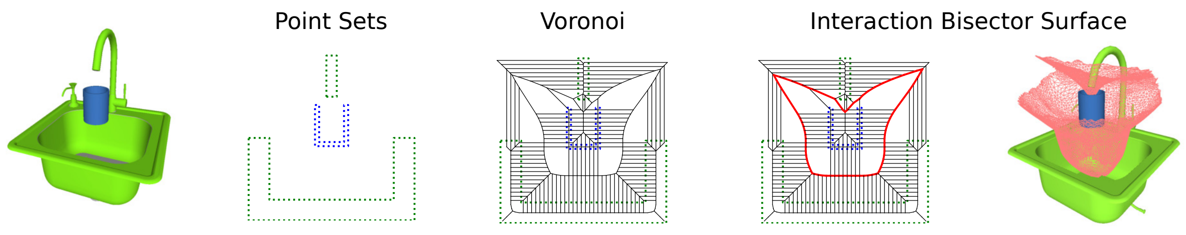


Figure 3.4: IBS between a cup and a sink. From left to right: target objects, 2D point set representation of the objects, 2D Voronoi diagram for all points, IBS (ridges diving object regions) is shown in red, Full 3D IBS surface in red.

Recent approaches for shape functionality and scene synthesis in computer graphics research [186, 188, 189, 195] have made use of the IBS concept; this is due to its powerful and robust ability to describe the relationship of synthetic object models in proximity. Figure 3.5 shows two IBS examples of similar interactions with different objects in a 2D scenario. This figure serves to illustrate the robustness of the IBS to geometric changes in the target scene; where, despite variations occurring in specific locations, the IBS manages to capture the overall common geometrical properties.

As discussed in the previous chapter, in order to leverage the robust features captured by the IBS, current approaches compute geometrical and topological features of the surface. In order to do so, one needs to first compute the IBS before any matching, prediction or training. Depending on the complexity of the scene, objects and point sets, computing the Voronoi diagram for 3D data is bounded by $\mathcal{O}(nlogn)$ and $\mathcal{O}(n^3)$, where n is the number of points for both objects. When the objects are either very large, very complex or there are many of them in a given scene, the computation of IBS and features on its surface can be very expensive and time-consuming. For this reason, computer graphics approaches typically pre-compute the IBS and features for every

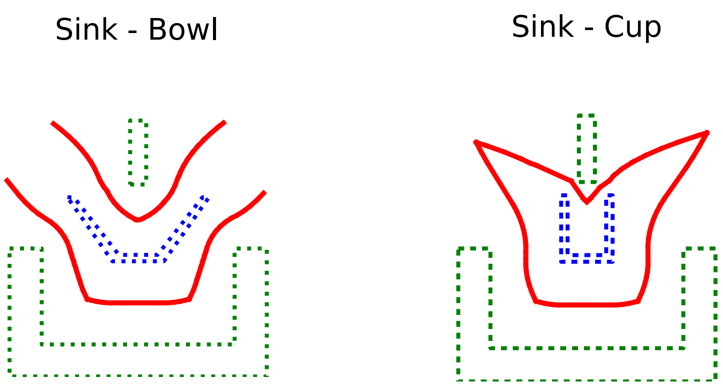


Figure 3.5: Two IBS examples in a 2D scenario for interactions between a sink and two different objects: a bowl and a cup

possible object pair in a database; thus allowing to more quickly synthesise similar interactions by only querying the pre-computed features. The following section introduces the approach proposed in this thesis. This approach extends the IBS representation to a vector field that, among other benefits, allows to detect similar interactions with significantly less computations. This simple yet robust representation combined with an appropriate matching function makes the proposed method suitable for applications in robotics and computer vision.

3.3 The Interaction Tensor

The Interaction Tensor (**iT**) is a vector field representation able to characterise the static interactions between 2 generic entities (e.g. objects) in 3D space. This proposed representation builds on the IBS concept and extends its robustness by three main factors:

1. Proposing a representation suitable for visually generated data, e.g. pointclouds
2. Encoding the locations in the interacting entities that contributed to the computation of their bisector, i.e. provenance vectors
3. Introducing a descriptor that allows for real-time prediction of affordance candidate locations on RGB-D data, i.e. affordance keypoints

First, adopting a pointcloud representation brings the possibility to work not only with synthetic models but also with data generated from robotics sensors such as, e.g. RGB-D cameras. This gives the possibility of dropping the need for fine-grained geometries and detailed mesh information in order to reconstruct or compute the IBS surface. Second, embedding the provenance of points on the IBS enables to approximate its computation at prediction time; which is done by investigating if in a novel scene exist the regions needed to obtain the same (or a similar) surface. Third, the introduction of affordance keypoints allows reducing further the computations needed to approximate an **iT** at prediction time, since they take into account only the key locations in the target interaction. All of this favours the application for the proposed method in robotics and computer vision problems, avoiding the need to compute the bisector or complex surface features in order to make affordance predictions, and enable to predict geometrically-meaningful affordance locations in never before seen RGB-D scenarios from a single *training* example.

3.4 Computing the Interaction Tensor

The first step to compute the tensor characterising an interaction of interest consists in placing two objects relative to each other simulating the affordance under investigation. For instance, a bottle placed on top of a table to study *Placing* affordance. Due to the increased availability of online collections of synthetic object models, the interaction simulations are carried out using such publicly available[1] CAD models for the simulations of interactions. The process of performing these simulations with synthetic models is referred to as *training* example (or *training* for short).

The two objects involved in the interactions are referred to as **query-object** and **scene-object** (or scene) respectively. For instance, in a *Filling* a mug interaction, the mug would be a query object. Conversely, a scene-object is the second part of the interaction; this can be a whole object but also just part of the scene or furniture (i.e. world) that allows the affordance to take place. A tap and sink would act as scene-object for the *Filling* a mug example.

The placement of objects relative to each other is guided by selecting points or areas that should be close together in the interacting objects . As an example, in the interaction of *Placing* a bottle on a table, a point on the bottom (base) of the bottle and a point on the surface of the table are selected. Then, the transformation required to *connect* these points is computed and applied; resulting in the two objects being brought together simulating the intended interaction. The software developed for these purposes allows to apply rotations along any axis if the interaction under study requires it.

[1]https://3dwarehouse.sketchup.com/?hl=en

Once both objects have been placed in their corresponding poses, the next step is to generate a dense pointcloud representation. The pointclouds are obtain by uniformly (randomly) sampling points on every face (i.e. triangle) of the CAD model, where the number of points sampled on each triangle is proportional to its area. A point **d** sampled on the triangle defined by vertices A, B and C, with 3D coordinates **a**, **b** and **c** is generated by: $d = (1-\sqrt{r_1})\mathbf{a} + \sqrt{r_1}(1-r_2)\mathbf{b} + \sqrt{r_1}r_2\mathbf{c}$ .Where r_1 and r_2 are random numbers sampled from a uniform distribution between $[0,1]$. As a result of this sampling technique, more points will be generated on larger triangles in the mesh; however, the location of these points is left at random. This sampling technique has the advantage of producing more "realistic" pointclouds to those produced with alternative approaches such as e.g. Poisson disk sampling, since the randomness involved in the computation makes the distance between neighbouring points not to be uniform or constant over the whole pointcloud.

One important requirement for the computation of the IBS, as observed in [186], is that the pointcloud densities (i.e. number of samples) must be high enough to avoid undesired anomalies in the IBS surface or penetrations of the IBS into the objects. Deciding on the number of points is not trivial since it will depend on the objects' dimension and complexity. An alternative is to adopt a post-processing step, where noisy or anomalous data is removed from the resulting **iT**. The is the approach followed for the computation of the **iT**, as it is more convenient given the diversity in dimension across all affordances studied in this work.

Once the dense pointclouds have been obtained, the next step is to compute the Voronoi diagram on the pointcloud comprised by the two objects, ridges shared by points from different objects are computed in order to obtain the IBS of the interaction. Figure 3.6 illustrates this process in a simplified 2D scenario for *Placing* a bowl on top of a table.

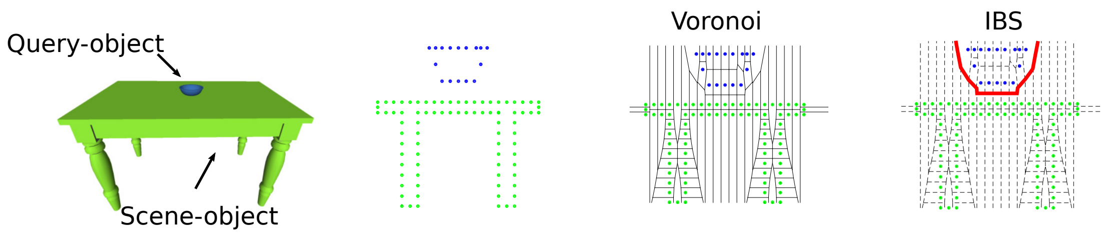


Figure 3.6: IBS of two interacting objects in a 2D scenario. From left to right: objects are placed simulating the interactions, pointcloud representation of the interaction, Voronoi diagram computed over the pointcloud of the two interacting objects, ridges shared by points from different objects form the IBS.

This process is carried out keeping track of the locations in the scene-object that gave origin to every point on the IBS, i.e. the centres of Voronoi cells in the scene. These locations are encoded in the **iT** in what has been named **provenance vectors**. A provenance vector is a vector going from a point on the IBS to its closest Voronoi cell in the scene-object. Figure 3.7 illustrates how provenance vectors are computed using the simplified 2D scenario.

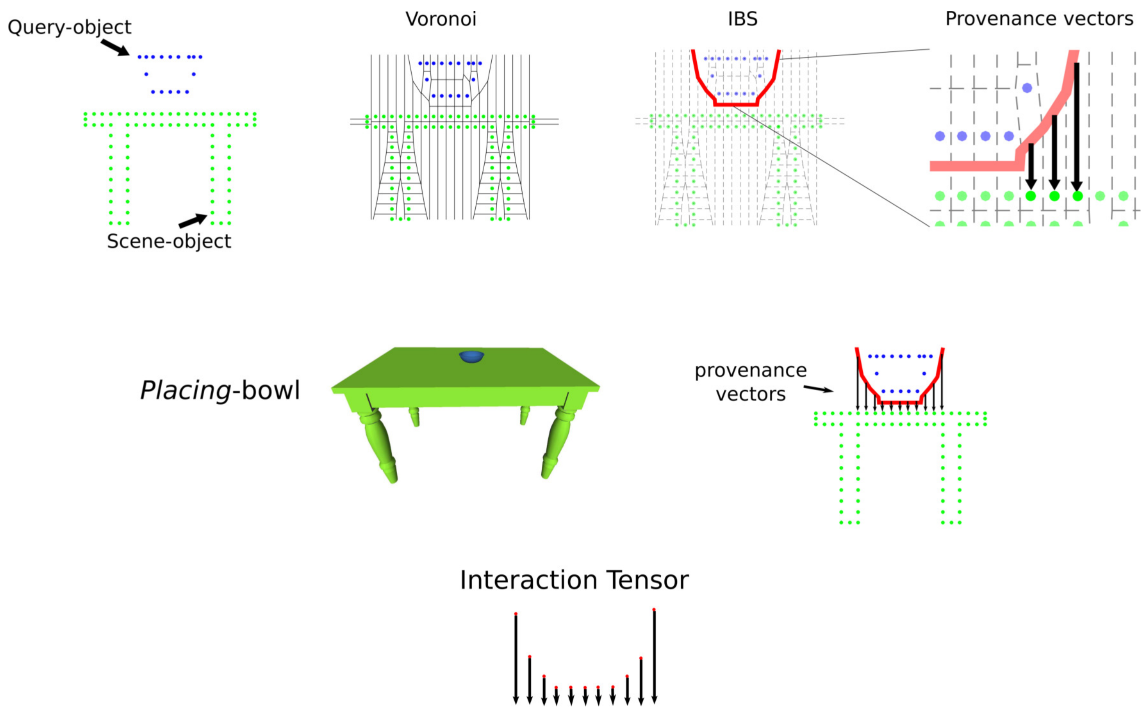


Figure 3.7: Provenance vectors and interaction tensor a 2D scenario. A provenance vector goes from points on the IBS to its closest Voronoi cell centroid in the scene-object. Together all provenance vectors and points on the IBS comprise an Interaction Tensor.

The **iT** descriptor of the investigated affordance is comprised of points on the IBS and their associated provenance vectors. Formally, given the interaction bisector surface B, and an object O in the scene formed by points $O = \{o_1, ..., o_i\}$, the vector field characterising the interaction is defined as

$$\mathbf{iT}(B) = P\hat{i} + Q\hat{j} + R\hat{k} \tag{3.1}$$

where

$$P = \hat{G}(B)_{\hat{i}} - B_{\hat{i}}$$

$$Q = \hat{G}(B)_{\hat{j}} - B_{\hat{j}}$$

$$R = \hat{G}(B)_{\hat{k}} - B_{\hat{k}}$$

with

$$\hat{G}(B) = \arg\min_{i} \|o_i - B\|_2$$

In principle, the Voronoi diagram (thus the IBS and **iT**) extends towards infinity; in practice, the representation is trimmed to fit a sphere of radius equal to the diagonal of the query-object bounding box.

The interaction tensor inherits from the IBS the discriminative power in characterising the relationships between sets of objects. It preserves key geometrical features while being robust to changes in the geometry of the interacting objects. The top row in Figure 3.8 shows examples of the **iT** for *Filling* affordances using query-objects with changing geometries. In contrast, the bottom row in the same figure shows **iT** examples generated with the same query-object (coat hanger) but scene-objects (coat racks) with varying geometries.

It should be said that in Figure 3.8 and all remainder figures with 3D tensors in this thesis a few considerations have been made for ease of display and better understanding the vector fields. These considerations include rescaling width and length of the vectors, as well as translating them by one unit towards the tail (as in *expanding* the field). Altogether this is aimed at making more evident the similarity between tensors.

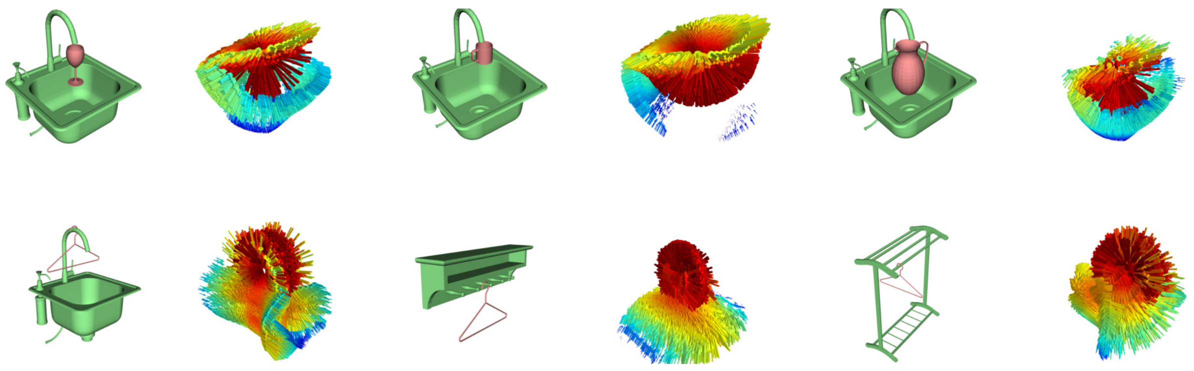


Figure 3.8: The **iT** preserves key geometrical features despite variations in the involved geometries. The top row shows **iT** examples for *Filling* affordance using different query-objects: glass, mug and pitcher. The bottom row shows **iTs** for *Hanging* a coat-hanger with various scene-object geometries.

The point being made with Figure 3.8 is demonstrating that despite geometrical changes in the interacting objects the **iT** retains the overall shape or geometrical features characterising the interaction. Additionally, as shall be made evident in the following chapter, the inclusion of provenance vectors allows to more quickly predict the likelihood of the affordance in a novel scenario. The affordance descriptor based on the **iT**, with an appropriate similarity or matching function (Chapter 4), opens the possibility to detect similar interactions in novel scenarios without the need for fine-grained geometries or mesh information.

3.4.1 Outlier removal

Depending on the density of the pointclouds used to compute the interaction tensors, there will be noise or irregularities present in the data. Although this problem can be avoided by having very dense pointclouds, it is preferable to solve the problem via an outlier removal post-process. This is because the number of points needed to meet the "high" density requirement will depend on the object's dimension. Larger affordances such as *Sitting* for a human would require many more points than smaller interactions such as Filling. The proposed post-processing consists in keeping the number of points sampled on the objects bounded to 100K and implement a RANSAC[208] -like approach for the removal of outliers.

The algorithm removes spurious data by iterating through the points on the IBS and comparing the provenance vectors of the current sample with those in the vicinity. After a few iterations, the points whose provenance vector is further than 1.5 standard deviations from the mean vector in their vicinity are removed from the tensor. The radius of the vicinity is proportional to the size of query-object, i.e. 10% of bounding box length. Figure 3.9 shows the removal of outliers related to errors in the computation of Voronoi diagram (e.g. quick hull algorithm errors). Figure 3.10 show examples of outliers related to penetrations that appear when lower density pointclouds are used to compute the IBS.

3.4.2 Weighted Interaction Tensor

Points on the IBS are defined by *provenance vectors* from the scene-object, this information is used to assign a weight to every location on the interaction tensor, $W = \{w_b, \forall b \in B\}$. The weight related to any given location in the interaction tensor is computed from the magnitude of its corresponding *provenance vector*. This weight or distance represents how relevant is every point for the interaction taking place between the objects. The idea behind this weight assignment is that locations where objects are closer together have higher importance, as opposed to those regions where the interacting objects are further apart. An example of such scenario can be

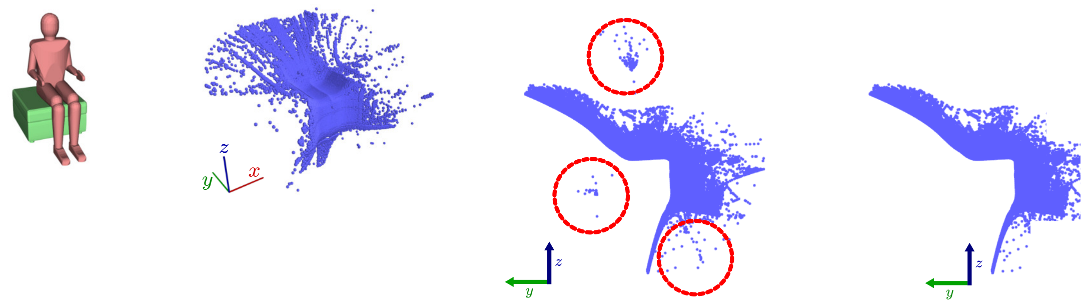


Figure 3.9: Outlier removal for an **iT** of *Sitting-human* affordance (only points on the IBS are shown). These noisy data points are associated with numerical errors in the quick hull algorithm. From left to right: interaction example and its associated IBS. Detected outliers. Cleaner data points on the IBS.

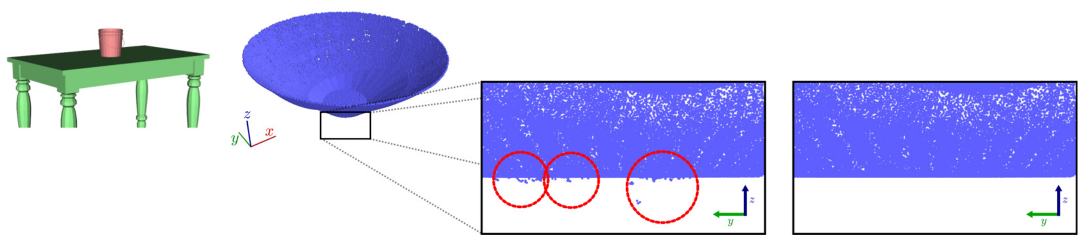


Figure 3.10: Removal of outliers for an **iT** of *Placing-bucket* affordance. These outliers are caused by lower density pointclouds in the interaction example. From left to right: interaction example and its associated IBS. Detected outliers. Clean data after removal of object penetrations.

seen in Figure 3.8, which shows that the region of the **iT** where the coat-hanger's hook *wraps* around the "hanging racks" is assigned higher weights. At the beginning of the research various weighting criteria were considered; the alternatives included were:

- **Distance to test-point** Weight is equal to the distance between the point on the IBS and the reference point in the scene-object. The reference point is the point selected in the scene-object to guide the interaction simulation during *training* (e.g. middle point under bottle in *Placing*-bottle affordance)

- **Distance to scene-object** Weight is equal to the distance from the point on the IBS to its nearest point in the scene-object. This is equivalent to the magnitude of the *provenance* vector as introduced earlier. Also, notice that this distance is the same as calculated from points on the IBS to their nearest neighbour on the query-object.

- **Distance to both objects** Weight is equal to the distance from the point on the IBS to its nearest neighbour in the scene-object, plus the distance to its nearest neighbour in the query-object.

- **Distance to centroid** Weight is equal to the distance from the points on the IBS to the centroid of the query-object.

Figure 3.11 shows these alternative weighting methods for different interactions. Whereas for *Placing* and *Filling* affordances the difference is minimal, more complex interactions such as *Riding* show higher variance for the high-weight concentrations. For instance, considering the distance to the query-object centroid would regard the chest of the biker (human) as the most relevant area of the interaction. Alternatively, distance to both objects would focus a lot on the hands and not so much on the saddle. Another important difference is observed in *Sitting*, where the distance to scene-object (i.e. provenance) is the only alternative that takes into account the region near the legs of the human. This was corroborated by testing the weighting alternatives in novel scenes, where the most notable difference was obtained with *Riding* and *Sitting*. The alternatives where the human legs are not regarded as important (lower weight) predicted candidate locations for *Sitting* where the legs would be inside furniture of other objects in the scene. These predictions would not be possible to achieve in a real scenario.

Finally, Figure 3.12 shows **iT** examples for the affordances considered for this research: *Placing*, *Hanging*, *Filling*, *Sitting* and *Riding*. These tensors illustrate the provenance-based weighting depicted by the colouring in the vectors. High weights are coloured in red while lower weight locations are rendered in blue.

3.5 Affordance descriptor

The **iT** is a high dimensional and rich representation for object interactions, employing it directly as a descriptor for affordance detection would require costly computational resources, and it would also deviate from Gibson's concept of *direct* and economical perception. In order to reduce computational costs and improve the generalisation capabilities of the descriptor, the dimensionality of the representation is reduced by drawing N samples from the **iT**. This subset comprises what has been named *affordance keypoints* $\mathbf{X} = \{X_1, X_2, ..., X_n\}$ where $X_n = \langle b_i, \mathbf{iT}(b_i) \rangle$. In other words, each *affordance keypoint* is formed by a 6-dimensional feature vector which consists of the Cartesian coordinates (x, y, z) of the data point b_i on the bisector surface, and its associated provenance vector $\vec{p}$ to the scene-object. The scalar $|\vec{p}|$ encodes the importance

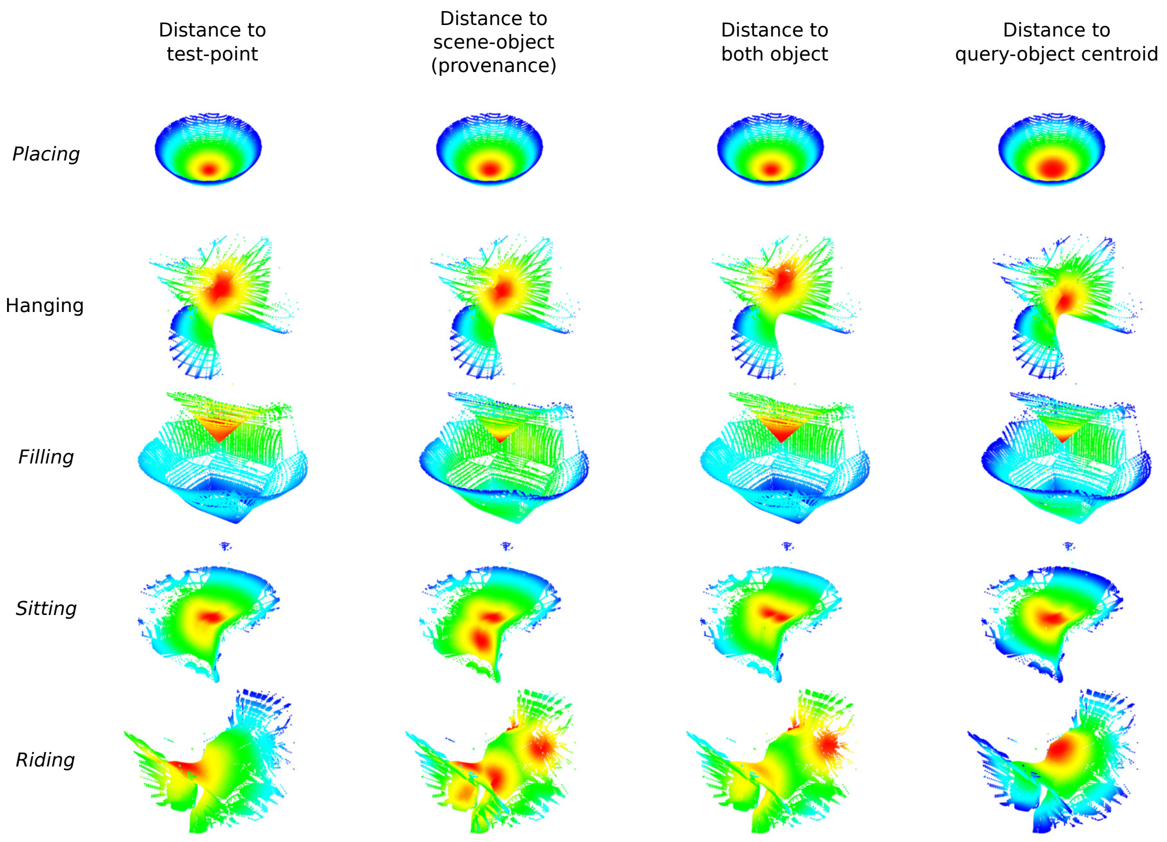


Figure 3.11: Weighting methods considered for the interaction tensor. Weights are shown as colours of the points on the IBS: higher weights are depicted in red. For some affordances, e.g. *Placing*, no major differences are observed; whereas for *Riding* and *Sitting* affordance the differences are notorious and relevant.

(weight) of a keypoint in the interaction between objects; since in principle the shorter the vector, the more significant the interaction is between them. Figure 3.13 depicts the method to compute *affordance keypoints* forming the descriptor for *placing* a bowl on a table in a 2D case.

Each affordance descriptor $X_{\text{affordance}}$ has $N \times 6$ dimensions, where N is the number of keypoints sampled from the **iT**. For this work two sampling strategies are investigated: uniform sampling and weight-driven sampling. These are described in the following subsections.

3.5.1 Uniform sampling

This strategy consists in sampling a target number of points N which, as the number suggests, are uniformly distributed over a given **iT**. This sample set is obtained as follows. First, a voxel grid is fitted to the pointcloud represented by the IBS. This grid is based on an OcTree structure

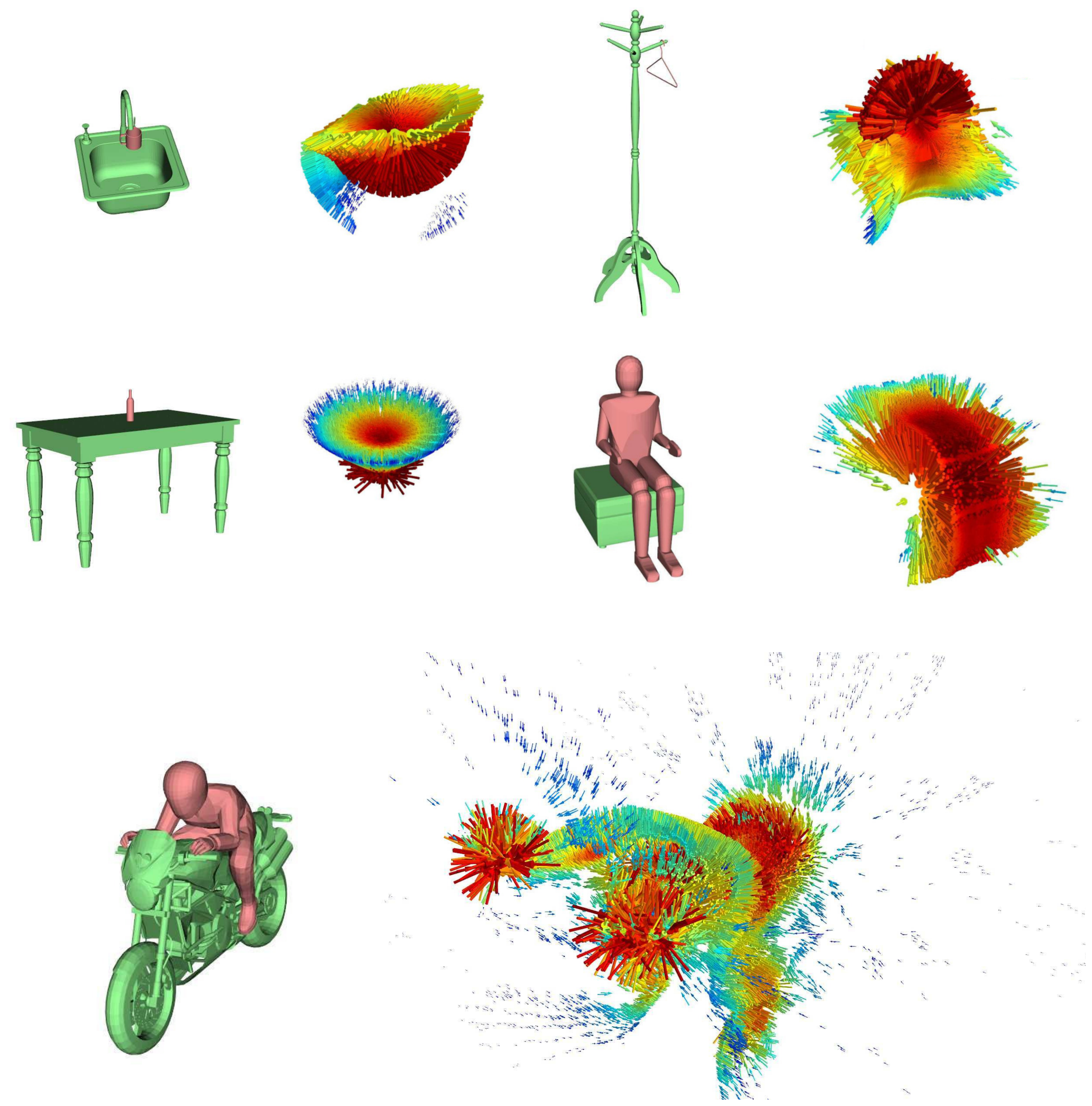


Figure 3.12: 5 **iT** examples of affordances studied in this research. Top row: *Filling*-mug, *Hanging*-coat-hanger. Middle row: *Placing*-bottle, *Sitting*-human. Bottom row: *Riding*-biker.

which splits the pointcloud into equally-sized parts iteratively until the number of voxels is equal to the sample size N. Then, the closest point in the IBS is computed for every voxel centroid in the final grid. Finally, the affordance descriptor is formed by the considering as affordance keypoints the IBS points (from the previous step) and their associated provenance vectors. Figure 3.14 shows the process to obtain a uniformly sampled keypoints.

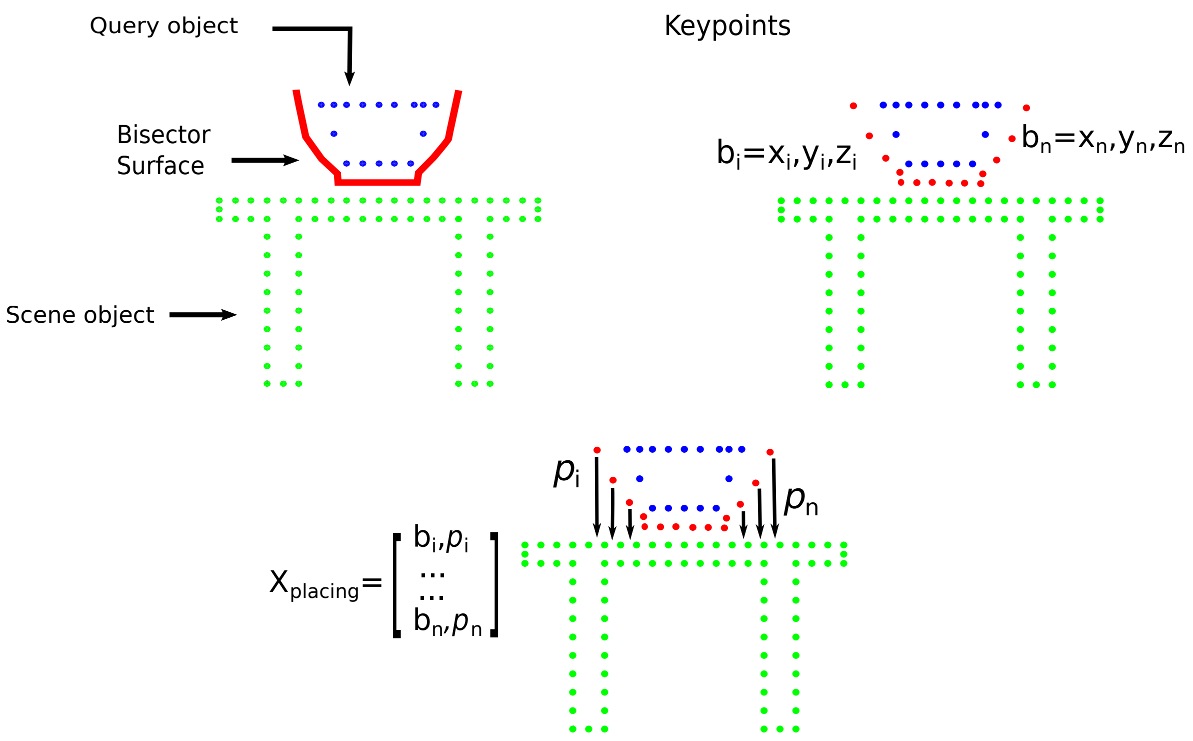


Figure 3.13: Affordance descriptor for *placing* a bowl on a table in a 2D scenario. A set of points is sampled from the bisector surface. An affordance keypoint is obtained by computing the interaction tensor over these sampled points. These keypoints lead to the interaction tensor descriptor $\mathbf{X}_{\text{affordance}}$.

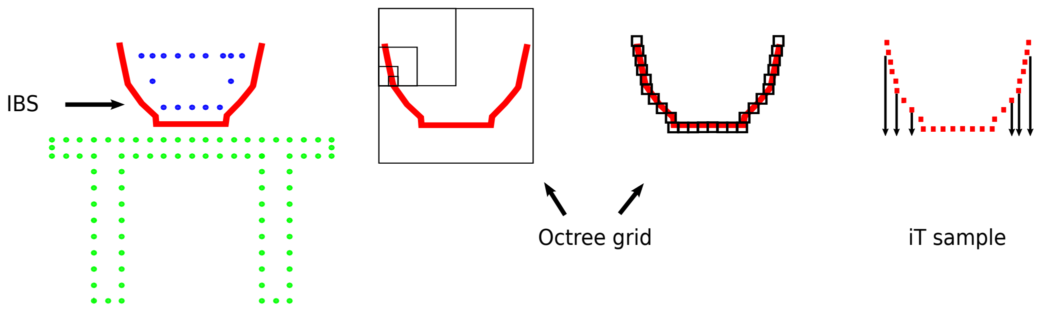


Figure 3.14: Uniform sampling based on an Octree fitted to IBS pointcloud. The octree structure iteratively splits the pointcloud (i.e. grid) until the target number of samples is achieved.

3.5.2 Weight-driven sampling

The second sampling strategy samples from a distribution where probabilities are inversely proportional to weights in the **iT**:

$$P(b_i \mid w_1, w_2, \dots, w_n) = \frac{w_i}{\sum_{k=1}^{n} w_k} \tag{3.2}$$

where

$$w_i = 1 - \frac{|\vec{p}_i|}{|\vec{p}_{max}|}$$

Weights are given by the magnitude of the provenance vectors in any given **iT** as a proportion of the largest vector $\vec{p}_{max}$ in the current tensor. The idea behind this sampling method is to have a more meaningful representation (i.e. higher keypoint density) in locations considered to be highly relevant for the interaction. These typically are locations where objects come closer together or touch, for instance in Figure 3.12 the saddle, handlebar grips and footrest of the motorcycle.

Figure 3.15 shows **iT** descriptors obtained with the two sampling methods that have been just described. As expected, the weight-driven sample focuses the keypoints in the regions closer to the tap spout; whereas uniform sampling focuses on maximal coverage of the original tensor.

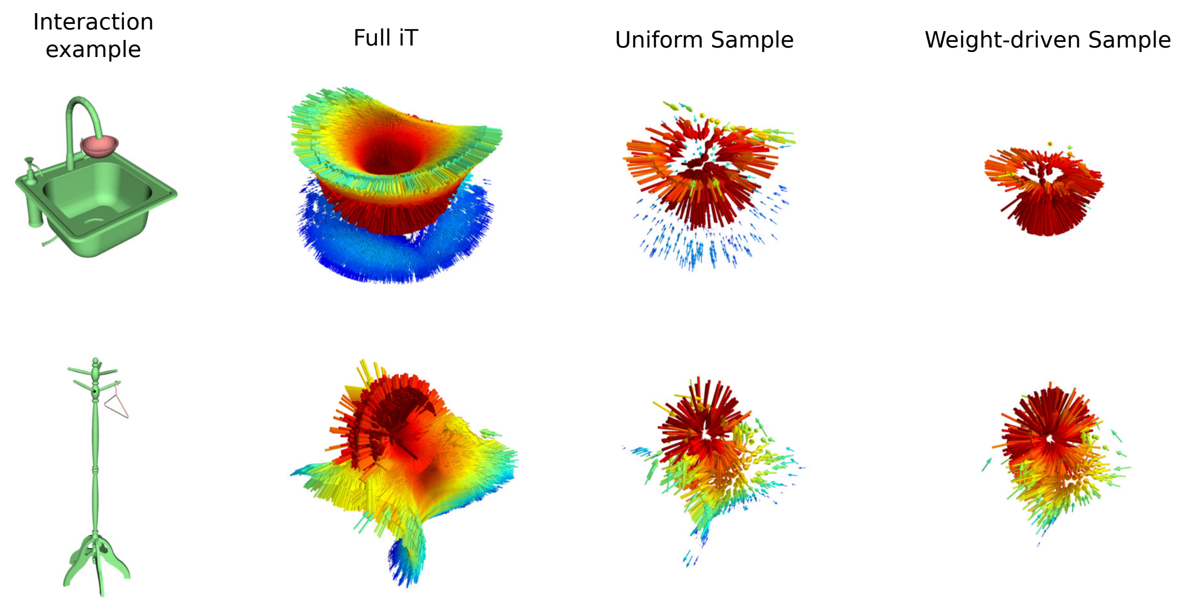


Figure 3.15: Affordance descriptors achieved with different sampling methods ($N = 512$). Columns on the left show the interaction and its full tensors as a reference.

3.5.3 Training pose

In addition to computing an **iT** and sampling from a given interaction example, some auxiliary information such a the query-object and descriptor poses is also needed. The complete process of extracting such information, computing interaction tensors and descriptors is referred to as *training* in the Chapters 3 and 4; more specifically, this is the *training* stage for the one-shot affordance prediction approach presented in this thesis. On the other hand. the process of predicting affordance locations with the one-shot approach (Chapter 4) is referred to as *testing*. The single example of the interaction, for every interaction in this research, is computed using a single orientation of the query-object and assuming that the pose is stable under standard gravity vector ($z \downarrow$).

Descriptor pose

As shall be detailed in the following chapter (Chapter 4), in order to test and predict affordances in a never before seen environment the affordance descriptor should be aligned relative to a test-point in the scene. To accomplish that, some degree of translation invariance is needed to allow that multiple points can be tested regardless of their location w.r.t. world frame. This translation invariance is feasible sine after *training* an interaction, the pose of the affordance descriptor is stored in a local frame relative to the scene-object used to generate the **iT**. More specifically, the *reference* point selected in the scene-object and used to guide the simulation (i.e. interaction example). The information of this point in the scene is used to compute and apply a transformation to every affordance keypoint such that the affordance descriptor is expressed in Cartesian coordinates (points on the IBS and provenance vectors) relative to the reference point in the scene.

Query-object pose

A similar approach is followed to store the pose of every query-object. First, origin in the object's local frame is translated such that it is coincident with the reference point(in the query-object) used to compute the interaction example. This is an important step for two reasons:

- Many CAD models available online will have coordinate systems relative to a local frame defined by either the CAD software or a human designer. In order to study similar interaction across different objects, a canonical (consistent) pose for every object is needed.

- Some objects afford more than one interaction, and their local frame should change accordingly. For instance, a pitcher for *Placing* should have its coordinates relative to the base, whereas the same pitcher for *Filling* should have coordinates relative to the top; furthermore, for a *Hanging* affordance, the origin of the local frame should be near the handle.

Figure 3.16 exemplifies scenarios where having an appropriate local frame plays an important role in the objects' training pose. Whereas affordance prediction does not need the query-object, the benefit of the pose convention is shown to better illustrate the predicted interactions (Chapters 4 and 5 and during scene augmentation in Chapter 6.

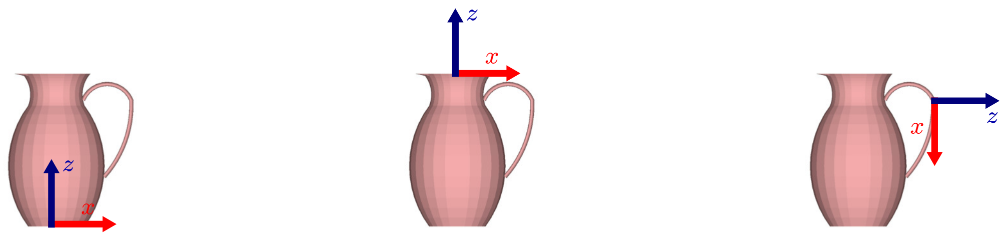


Figure 3.16: Objects need to have their coordinates (local frame) according to the affordance under investigation. This local frame is set from the moment of *training* (i.e. interaction example computation). From left to right are shown local frames in a pitcher for three affordances: *Placing, Filling, Hanging*.

3.6 AffordanceSim

In order to compute interaction tensors, descriptors and all other relevant data, software tools were developed based on C/C++. The set of libraries that were developed for these purposes are referred as AffordanceSim, and these include the software needed for *training* (i.e. produce interaction tensor examples) and testing (i.e. perform affordance predictions) based on the iT algorithm. The vanilla version of AffordanceSim was made publicly available[2] under the MIT License in September 2018. Details regarding the software requirements, instructions of compilation and usage, as well as functionality, can be found in Appendix A.1.

[2]`https://github.com/eduard626/interaction-tensor`

3.7 Conclusion

This chapter introduced the Interaction Tensor (**iT**) representation to characterise static interactions between any two objects in 3D space. Based on the robust description capabilities of the IBS, the newly proposed representation allows to devise an affordance descriptor in a straight-forward manner, i.e. by drawing samples from the **iT**. As shall be illustrated in the next chapter, with the inclusion of *provenance* vectors and a suitable similarity measure, the proposed representation allows to: 1) more efficiently detect similar affordance candidate locations in novel scenarios; in addition to 2) not requiring fine-grained geometrical information in the novel scene. In contrast to approaches in computer graphics [186, 188, 189], the proposed method 1) uses more flexible representation (i.e. pointclouds) instead of mesh data and 2) does not need to recompute the **iT** for predictions. All of this brings the possibility of more easily employ the **iT** method in RGB-D data for robotics and computer vision tasks.

Chapter 4

Affordance Detection

4.1 Introduction

Being able to determine affordances in an unknown environment is a key competence for acting-perceiving agents. In principle, it can alleviate the computational approach to visual perception from the focus on objects and their somewhat arbitrary labels, which would need to be learned extensively. The work proposed in this thesis focuses on a subclass of affordances between rigid objects, specifically the interactions between objects (or a human) and the environment. The approach proposed to do so is based on the Interaction Tensor (**iT**) introduced in the previous chapter; where it has been shown that an affordance descriptor can be obtained by sampling affordance *keypoints* from an **iT**.

First, the current chapter demonstrates how the affordance descriptor computed from the **iT** representation allows to predict affordance candidate locations or hypotheses in a novel scene. The proposed similarity function allows to quickly detect affordance locations that give answer to perceptual questions such as "*Where can I afford to place a bottle?*","*Where can I hang a handbag*?" , "*Where can I fill a mug?*", etc. This testing scenario is referred to as **affordance query**. Later, this chapter introduces an algorithm that enables to efficiently increment the number of affordance-object pairs queried in any given location without heavily comprising the detection rate. Results and evaluations of the experiments carried out with the proposed methods are shown accordingly, including comparisons with alternative approaches.

4.2 One-shot affordance detection

The **iT** representation inherits from the IBS the robust and powerful ability to capture the geometric characteristics of the interaction between any pair of objects. Sampling from this representation allows to build an affordance descriptor which is compact but still retains the geometrics information captured by the **iT**. In this section is introduced the similarity function and algorithm that allows to quickly determine affordance hypotheses in a novel scenario from a single *training* example, i.e. one-shot learning.

Previous methods employing IBS for functionality analysis have relied on the computation of local shape features on its surface which are then fed to a machine learning algorithm or similarity function in order to retrieve or synthesise matching interactions. Thus, for any new pair of objects, they need first to compute the IBS, then compute shape features at various locations on the IBS, and (typically) build histograms representing the global shape features [186, 188, 189, 191]. All of this turns into a very time-consuming process, which also needs some form of mesh information in order to compute the proposed local features. As seen in Chapter 3, computing an IBS requires dense pointclouds or a post-processing step in order to remove noisy or spurious data, process the adds extra computations to the "traditional" detection pipeline.

The alternative proposed in this thesis is to approximate the **iT** descriptor at prediction time (i.e. *testing*) via a Nearest Neighbour (NN) search. It is worth reminding that the *provenance* vectors in the **iT** account for regions in the scene that contributed to the computation of the IBS surface; the proposed algorithm uses this information to investigate whether those regions exist in a novel scenario, these regions would allow computing the same or a similar **iT** (if required). In this sense, the NN-search is used to investigate if the point in the scene required to compute a point on the IBS exists; or more precisely, if the point in the scene is where is expected to be.

Notice that the **iT** approximation process does not need the *training* objects (query-object nor the scene-object) to make a prediction, only the descriptor $X_{\text{affordance}}$ (X_{a} for short) of the interaction between such objects suffices for the NN-search approximation. In other words, after *training* an affordance neither the query-object or the scene-object are required for affordance prediction. The query-object is only needed to visualise more clearly the predicted interaction.

In order to estimate the likelihood of an affordance in a new test point t_i, the descriptor X_a needs to be aligned relative to the new location. From training, the pose of X_{a} relative to a reference point in the scene is known; therefore, testing in t_i is done by simply applying the translation corresponding to that point T_{t_i}. It is also important to notice that from the *training* example only one orientation is considered; thus, the descriptor can be rotated around the gravity vector $\vec{z}$, through transformation R_o, in order to test the affordance at different orientations.

In summary, the pose of the affordance descriptor given a test-point p_i is given by

$$
\begin{aligned}
X'_a &= R_o T_{p_i} X_\mathrm{a} \\
&= R_o \begin{bmatrix} 1 & 0 & 0 & p_{i_x} \\ 0 & 1 & 0 & p_{i_y} \\ 0 & 0 & 1 & p_{i_z} \\ 0 & 0 & 0 & 1 \end{bmatrix} X_\mathrm{a} \\
&= R_o \begin{bmatrix} 1 & 0 & 0 & p_{i_x} \\ 0 & 1 & 0 & p_{i_y} \\ 0 & 0 & 1 & p_{i_z} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_{a_x} \\ X_{a_y} \\ X_{a_z} \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & t_{i_x} \\ 0 & 1 & 0 & t_{i_y} \\ 0 & 0 & 1 & t_{i_z} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_{a_x} \\ X_{a_y} \\ X_{a_z} \\ 1 \end{bmatrix}
\end{aligned}
\tag{4.1}
$$

Where θ is the angle at which the interaction should be tested. If $\theta = 0$ then it is exactly the orientation used for the *training* example. In this work, eight orientations evenly distributed in $[0, 2\pi)$ were considered.

Once the descriptor is aligned relative to the test point, the NN-search is performed to investigate how likely it is to afford the interaction in this location. First, the 1-NN in the scene is computed for every *keypoint* in the descriptor; this allows to calculate *test vectors* $\vec{v}_t$. These vectors are the approximation of *provenance vectors* at test time. Then, a score or likelihood can be computed by comparing both sets of vectors via Equation 4.2:

$$
s_i = \sum_{i=1}^{N} \frac{1}{\sqrt{2\pi w_i^2}} e^{-\frac{\Delta_i^2}{2w_i^2}} \tag{4.2}
$$

where

$$
\Delta_i = \frac{\|\vec{v}_{ti} - \vec{p}_i\|}{\|\vec{p}_i\|}
$$

Where N is the number of affordance keypoints in the descriptor, Δ is the magnitude of the difference between vectors expressed as a proportion of the expected provenance vector $\vec{p}_i$. Important is to notice the inclusion of keypoint's weight w_i, which helps to relax the matching criteria (i.e. magnitude of the difference) for regions of the interaction that are not very relevant. In contrast, those regions of the interaction that are more relevant (i.e. higher weight) have a stricter matching. Figure 4.1 illustrates the idea behind the similarity/matching of vectors for *Placing* affordance prediction.

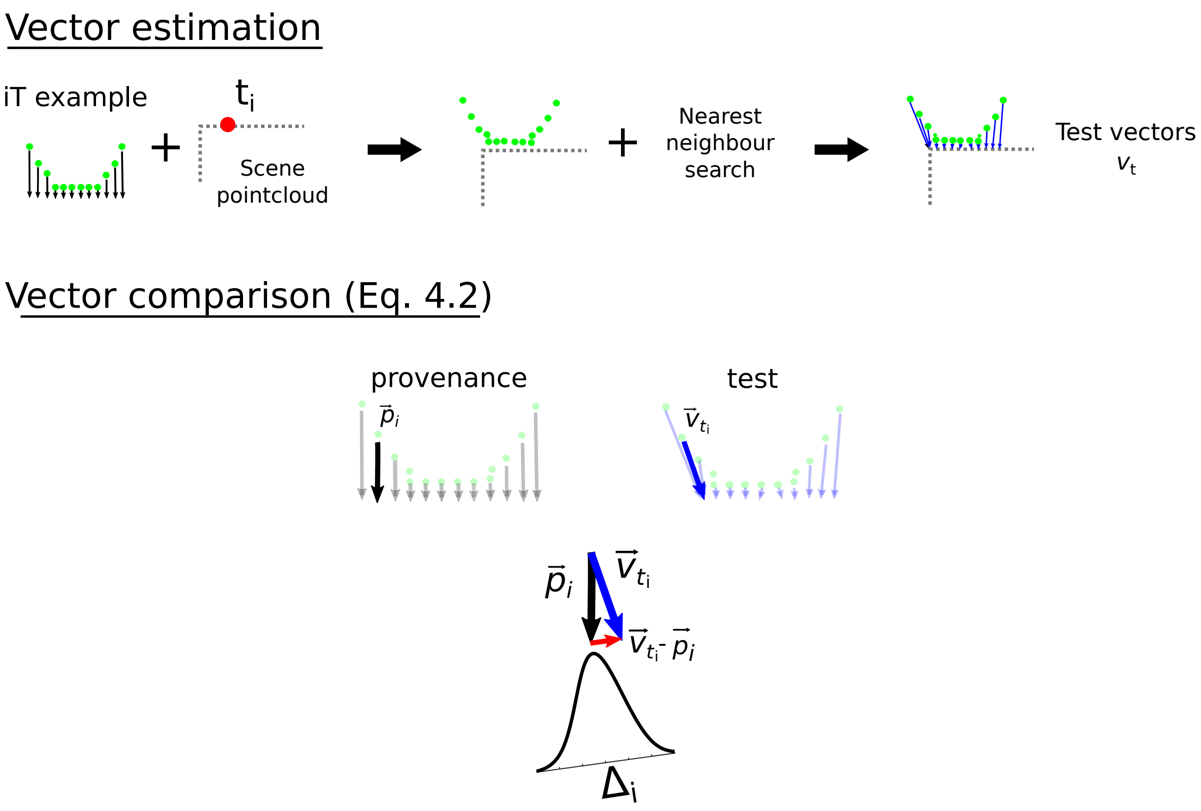


Figure 4.1: Affordance predictions are carried out by approximating the **iT** descriptor at a test-point in the scene. Test vectors are computed via nearest neighbour search for every keypoint in the descriptor. The proposed similarity function compares provenance vectors and test vectors to produce a score based on pairwise vector differences.

More specifically, affordance location predictions are performed following Algorithm 1. First, test points $\{t_1, t_2, \ldots, t_i\}$ are uniformly sampled all over the input scene. Then, the voxel surrounding test point t_i is extracted, the size of this voxel is given by radius r_o which is equal to the diagonal of the bounding box of the query-object associated to the affordance being queried. This smaller scene pointcloud (voxel) is used to perform the NN-search needed to estimate test vectors;

which are then used to compute a score via the similarity function shown above in Equation 4.2. Figure 4.2 depicts the affordance detection algorithm from a given input scene and an affordance of interest(e.g. *Filling*-mug).

Algorithm 1 Affordance query

```
1: for all test points $T$ in scene do
2:     Extract voxel of radius $d_o$ around $t_i$
3:     for all orientations $\theta$ do
4:         Estimate test vectors using NN-search
5:         Compute score $s_i$ at $\theta_i$ using (4.2)
6:         if $s_i \geq S_{\text{prediction}}$ then
7:             Predict good location at $(t_i,\theta_i)$ with
8:             probability $s_i$
```

It is worth mentioning that the search performed in the scene or the way test-points are selected does not follow a particular criterion, not other than sampling from the input scene uniformly. In a way sampling uniformly ensures covering or testing locations along the whole scene. Whereas this could be seen as an exhaustive process, it should be kept in mind that no prior assumption is made regarding scene or objects appearance nor surface complex features. For instance, alternative approaches would compute curvature features or normal vector information in the scene to speed up the discrimination process; however, this would bias the detection towards parts of the scene that “are known” to afford an interaction. In other words, the approach proposed in this thesis is agnostic to scene shape features, and instead computes the likelihood of the interaction by *hallucinating* the interaction; it is until the interaction is carried out (by hallucinating) that the affordability of the interaction is discovered.

It should be mentioned that surface normal vectors in the scene were considered to guide or prune the search in early stages of this research; however, estimating scene normal vectors is a research problem in itself. The idea of such approach was to associate each affordance with an expected normal vector orientation in the scene, for instance, a normal pointing upwards for *Placing* affordances which should come from the surface supporting the object, or a normal pointing downwards for *Filling* affordances (e.g. in taps/faucets). The problem with such approach is noisy or wrong normal estimates. In general, the absence of a mathematical principle to solve for the sign of the normal, causes its orientation to be ambiguous, and often times not consistently oriented over an entire point cloud. This causes that promising locations are not explored if the normal vector in that surface is not correctly oriented (or as expected).

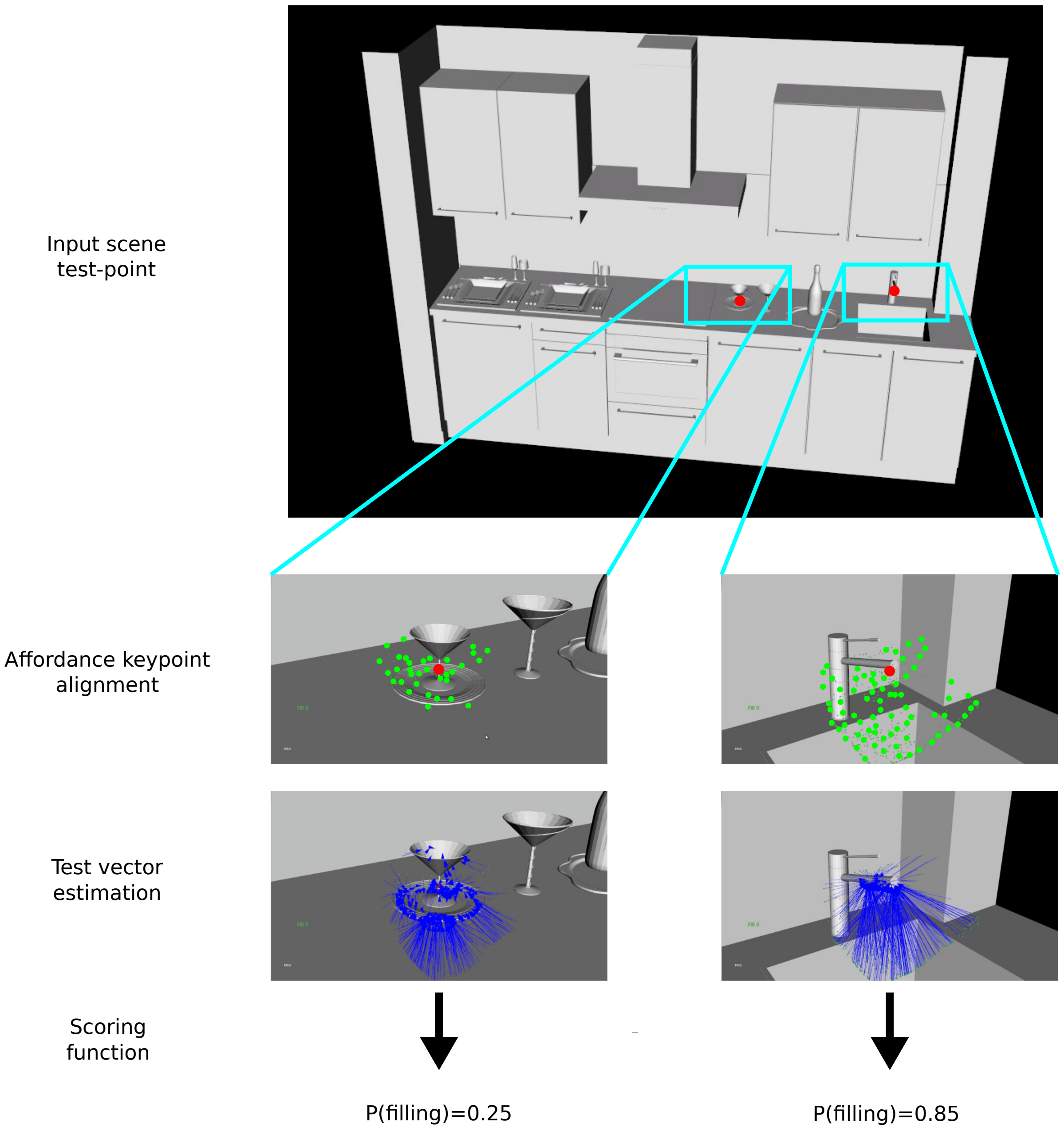


Figure 4.2: Affordance detection example for *Filling*-mug. Test-points are sampled from the scene (red). Affordance keypoints (green) are aligned relative to the test-point. Test vectors(blue) are estimated using a NN-search. Test vectors and *provenance* vectors are compared to produce a score.

4.3 Affordance query

In the following subsections are presented the experiments carried out in order to perform affordance detection (affordance query). The experiments comprise predicting affordance candidate locations over a set of 20 publicly available synthetic scenes[1] which include living-rooms, kitchens, offices as well as motorcycles. Examples of these scenes are shown in Figure 4.3; the full dataset can be found in Appendix A.2. As mentioned earlier, eight orientations are tested for every point in the input scenes. Pointclouds are generated from the synthetic scenes following the same approach to produce dense pointclouds for *training*, i.e. uniformly randomly sampling points on the surface of the CAD models.

For these experiments, eight affordance-object pairs were considered: *Filling*-mug, *Filling*-cup, *Placing*-bottle, *Placing*-bowl, *Hanging*-hanger, *Hanging*-handbag, *Sitting*-human, *Riding*-biker, from which $N = 512$ keypoints are sampled; this value was obtained empirically and proved to be a good balance between quality and speed of the predictions. A method to learn the optimal sampling for affordance prediction is presented later in Chapter 5.

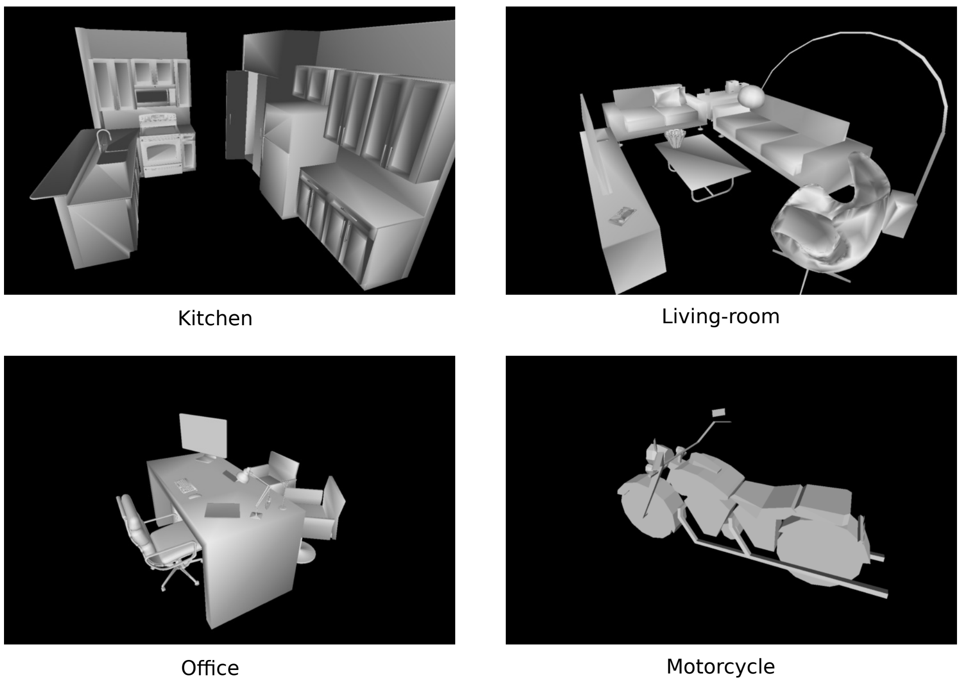


Figure 4.3: Examples of scenes used to query affordances. The scene dataset includes: living-rooms(5), offices(5), kitchens(5) and motorcycles(5).

4.3.1 Sampling methods

The two sampling methods used to generate an affordance descriptor were assessed in a first set of experiments. As a reminder, these sampling methods are: 1) Weight-driven and 2) Uniform

[1]`https://3dwarehouse.sketchup.com`

sampling. For these experiments, predictions are made while the individual scoring of provenance vectors and timings are tracked. This allowed to generate Figure 4.4, which shows the scores of the top 20% affordance predictions made with both methods.

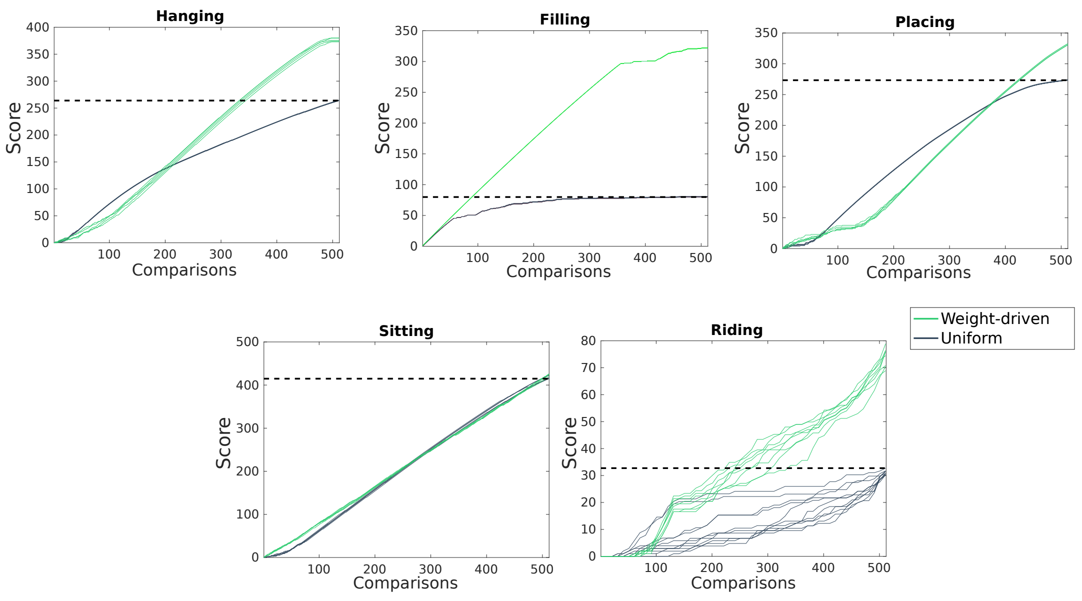


Figure 4.4: Plots show the performance of two keypoint sampling methods. For most affordances weight-driven sampling achieves the prediction score threshold faster than uniform sampling (fewer comparisons made at test time). For some affordances the difference can be subtle, whereas in some others such as *filling* affordance, the difference goes to 80%.

In these plots can be seen that when weight-driven sampling is used to compute scores, the algorithm reaches the prediction score threshold faster than uniform sampling in four out of 5 affordances. This because by making sure that higher-weight keypoints are presented, the algorithm becomes confident enough to make a prediction sooner than with uniform sampling. High-weight keypoints are compared first by the algorithm; hence, the score threshold is achieved performing fewer computations. On the other hand, uniform sampling achieves similar performances using on average 40% more comparisons (i.e. 200 keypoints more); which in most cases means all the keypoints in the descriptor. It stands out the plot from *Riding* which shows a "noisy" behaviour on both sampling methods. This, however, is related to the nature of the interaction. All the other interactions typically have one concentration or cluster of high-weight keypoints, which usually is where object come closer or contact each other. In the case of *Riding*, there are 5 of such high-weight clusters: motorcycle handle grips(2), footrests(2) and saddle; depending on the scene being tested some of those locations might not be there which can cause a jittery behaviour in the plot. For instance, if a complete motorcycle is found in the scene the plot would be smooth; but if only the "saddle" is present in the scene (like in Figure 4.5(d)), the graph will not show the same response.

Figure 4.4 also shows the thresholds used to predict a good location for the affordances. Different values were considered depending on the target interaction, whereas this is not ideal, it should be reminded that the main interest here was to find out the performance of different sampling methods over the **iT**. Later in this chapter will be introduced the single value of this threshold that produces the best predictions across all affordances.

Notice that, due to the fact that a search is performed on test-points located all over the scene, it is crucial to quickly asses the affordability of the interactions in any given location. There lies the advantage of using a weight-driven sampling approach to compute affordance descriptors. If the computing resources are limited (e.g. no GPU available) most calculations are done in a sequential manner, which further emphasises the importance of implementing a fast method. Even when parallelisation is introduced, the advantage of the same method is important since the computations are made using ordered batches of affordance keypoints and vectors. If at an early stage of the prediction (i.e. first batches) there are not favourable scores, the computations for the current test-point can be stopped; or one could only to pursue those that are more promising. Whereas this behaviour has not been implemented yet, it is a relevant consideration when testing on multiple locations all over the input scene.

4.3.2 Baseline comparison

In a second round of experiments, the proposed **iT** method is compared against two baseline algorithms: 1) IBS-only and 2) Naive matching. The former uses the query-object to compute the IBS at every test-point in the scene, producing a score by comparing the two pointclouds: *training* IBS and current IBS. This comparison is made by computing the best transformation that aligns both pointclouds via Iterative Closest Point (ICP) [209]. The second baseline is a *Naive* matching algorithm, which computes the pairwise distances between the query-object and the scene pointcloud. The idea behind this approach is to resemble template matching in traditional computer vision methods or methods that use distances between objects instead of a representations of the interaction between them. For fairness, both of the baseline algorithms sample points uniformly from all over the scene similarly as the affordance query algorithm (i.e.

uniformly all over the scene).

In addition to being slower or more computationally intensive, the method based only on the IBS as a descriptor is much more strict by trying to find only interaction opportunities closely similar to the *training* example. One first advantage of the **iT** approach is that, by considering a weighted vector field, there is a more relaxed matching criterion in parts of the interaction that are not critical to the affordance; this allows the detection of affordance candidate locations despite variations in the scene geometry. In order to achieve a performance similar to the **iT** descriptor, it is necessary to relax the matching threshold for IBS-only method; however, this increases the number of bad predictions (i.e. false positives). Figure 4.5 shows an example of such circumstances for *hanging* a coat-hanger on a rack.

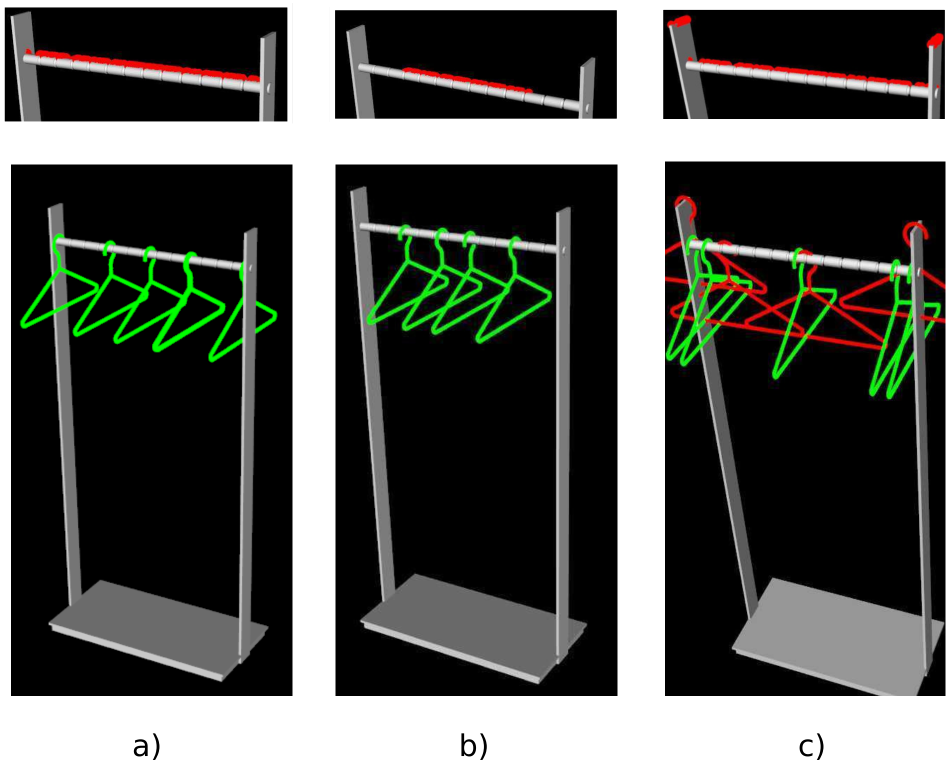
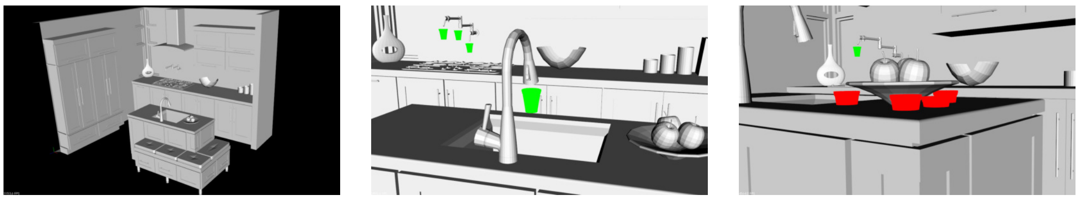
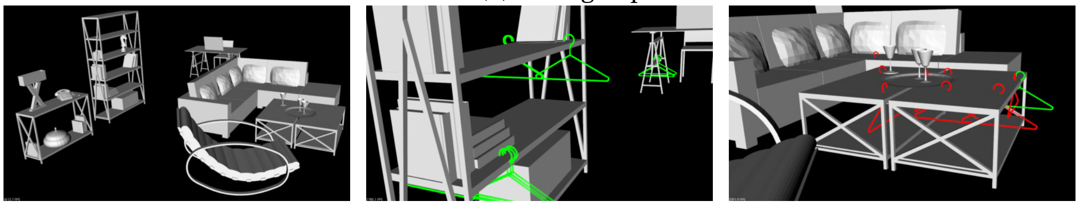


Figure 4.5: The **iT** descriptor (a) allows more flexibility in the prediction of affordance location candidates. The IBS-only method (b) predicts affordance in locations more closely similar to the *training* example (centre of the hanging rack). Similar performance with IBS-only method can be achieved by relaxing the matching threshold(c), but this is at the expense of increasing the number of false positives (red coat hangers).

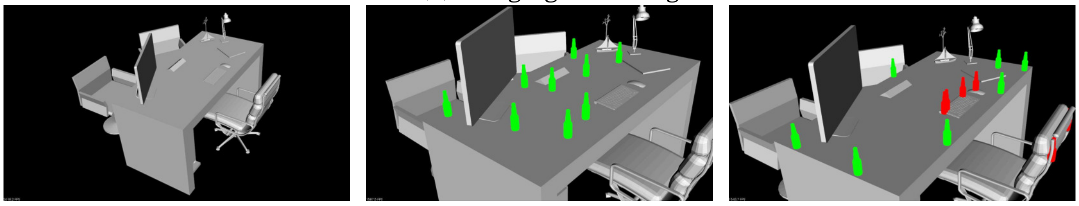
The second alternative method computes the best match by computing the pairwise distance between the two entities, query-object and scene, at the current test-point but without any explicit representation of the interaction between the objects; therefore the goal is to find the best possible "alignment" at test time by minimizing the distance error between example and current interaction. Figure 4.6 shows results contrasting the *Naive* algorithm and the **iT** approach. One important thing to notice about the *Naive* approach is that it does find some expected locations; yet it also predicts as good the locations with object penetrations, occlusions or intersections. This kind of predictions would not be useful or achievable in reality. For instance, 4.6(d) and 4.6(e) show *Naive* predictions for *sitting* and *riding* where the legs or parts of the body (query-object) are inside furniture. Similar cases are observed in Figure 4.6(a) - 4.6(c), where the predicted locations would make the query-object collide or to be inside other objects in the scene.



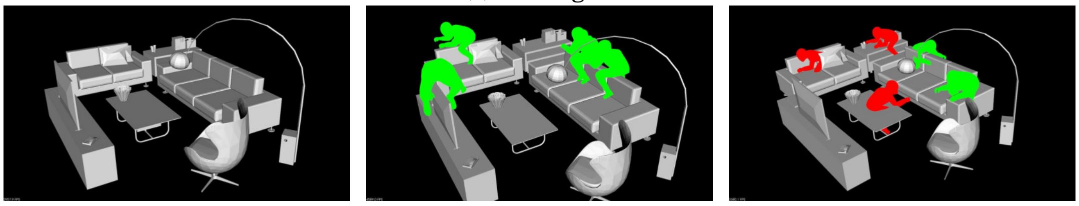
(a) *Filling* cup



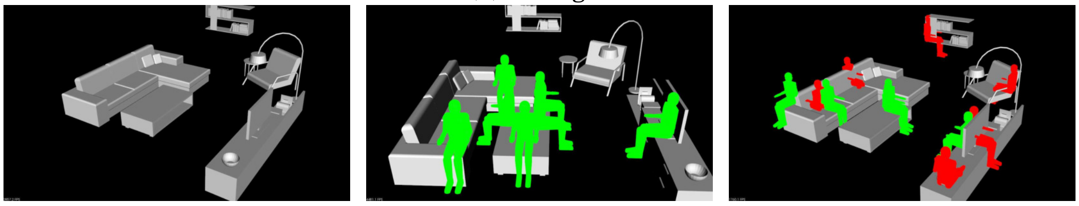
(b) *Hanging* coat-hanger



(c) *Placing* bottle



(d) *Riding* biker



(e) *Sitting* human

Figure 4.6: Results on the middle column show predicted positions using the **iT** descriptor. Results in the column on the right show predictions made with the baseline *Naive* algorithm. *Naive* algorithm predicts good locations with equal probability as bad or unachievable configurations (red).

During these experiments, two remarkable and unexpected predictions were observed. First, the **iT** algorithm suggests edges of flat surfaces as good candidates for *Hanging* a coat-hanger. Take into account that the *training* example consisted of an actual hanging rack, as shall be shown next, hanging a coat hanger from the edge of a shelf is regarded as possible according to human criteria. A second noteworthy outcome was the prediction of *Riding* affordances in living-rooms, especially in sofas. The algorithm is able to detect the saddle-like structure of the furniture and predicts it a good candidate for *Riding*. All of this highlights the generalisation power of the proposed method. Finally, Figure 4.7 shows example predictions for this latter affordance made in actual motorcycle scenes. Notice the important changes in the motorcycles' geometry.

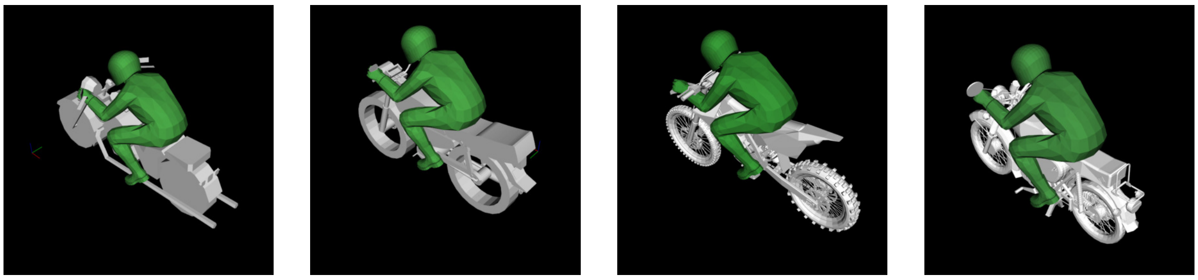


Figure 4.7: Predicted *Riding* affordance in never seen before motorcycles.

Affordance validation

The affordance predictions are further evaluated through Amazon Mechanical Turk, where the performance of the **iT** method according to human criteria was investigated. Due to the fact that no prior assumptions are made regarding objects in the scene affording the interactions, the predictions consists of locations that an agent would choose to accomplish an action. As an example, one can afford to place a bowl on a chair as much as one can sit on the kitchen's table. These are arguably valid placings but need to be validated by an "agent", in this case, a human.

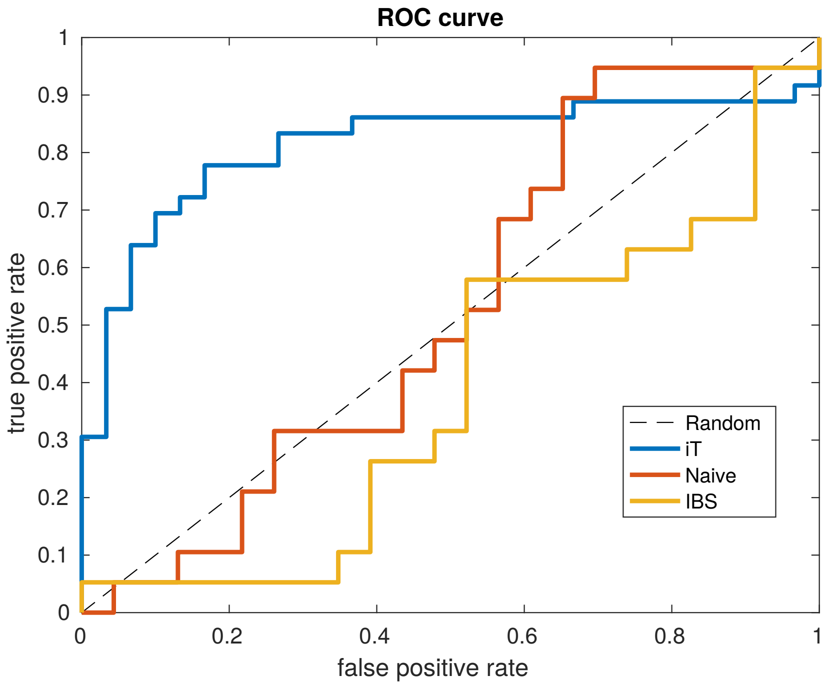
Human "annotators" were asked to select good locations for each one of the five *generic* affordances considered in these experiments: *Filling, Hanging, Placing, Sitting* and *Riding*. People were presented with six different options (2 candidate locations in 3 views each) at a time. From these options, they had to choose the ones that according to them were good locations for the interaction to take place. A total of 60 persons were involved in this validation of affordance locations; each person provided 10 annotations per affordance. Performance metrics were computed for the **iT** approach as well as for the baseline methods using the consensus of human annotations as ground truth. Results of this evaluation are shown in Table 4.1, which shows that on average the approach based on the **iT** achieves an accuracy of 84.90% and f-score of .825; outperforming the baseline methods in nearly all the affordance predictions. In other words, the **iT** method consistently predicts top geometric affordance locations in previously unseen scenarios that agree with human criteria approximately 85 per cent of the time.

	iT		Naive		IBS	
	Accuracy	F-score	Accuracy	F-score	Accuracy	F-score
Filling	96.00	**96.30**	87.50	88.89	87.50	90.91
Hanging	85.71	**82.35**	75.00	80.00	37.50	54.55
Placing	80.30	**81.16**	66.67	66.67	66.67	40.00
Sitting	72.00	63.16	77.78	50.00	88.89	**66.67**
Riding	90.48	**90.00**	75.00	75.00	37.50	54.55
Average	**84.90**	**82.59**	76.39	71.48	63.61	61.34

Table 4.1: Affordance prediction performance evaluated according to human annotators criteria (in terms percentage).

It is worth noticing that in the case of *Riding*, which could be considered the most complex interaction, **iT** outperforms the baseline methods with a more significant difference over IBS. As discussed previously, the baseline IBS algorithm mainly detects affordance at locations with scene geometries very close to the example. When there is no motorbike-like geometry (e.g. living-rooms), it struggles to predict such affordance. A similar situation occurs for *Placing* affordances, which remains challenging for the baseline algorithms. This is mainly due to the scene geometry at specific places; for instance, baseline algorithms will not *place* the query-object if the area is not completely clear (flat clear surface). Notably, all the algorithms have a high performance with *Filling*, which can be explained by the fact that locations affording such interactions have a more distinctive geometry (e.g. taps and sinks); these geometries are found very seldom (one in most kitchen scenes) which makes easier to detect the *Filling* affordance correctly. Another remarkable result can be observed for *Hanging* affordances; where, according to humans, hanging a coat-hanger on edges of flat surfaces is regarded as possible. Interestingly, **iT** and Naive approaches predict these type of interaction successfully. Traditional methods based on object appearance would fail to detect these cases.

For now, predictions have been made base on thresholds established individually per affordance. Ideally, there should be a single value for the detection threshold that yields the best performance in the general affordance (i.e. all affordances) prediction; for this purpose, the ”human labelled“ data is used to obtain the value of this parameter. The results from this process are shown in Figure 4.8, where is shown that the best performance across all five affordances is obtained with a threshold value of 0.52. In other words, a prediction made by the **iT** algorithm with a score above 0.52 is valid 82% of the time according to human criteria, regardless of the affordance being queried. It is interesting to see that in some points the baselines perform worse than random. When considering the every-affordance case, the overall performance of the baselines is penalised or heavily influenced by the low performance on complex affordances such as *Hanging* or *Riding*.



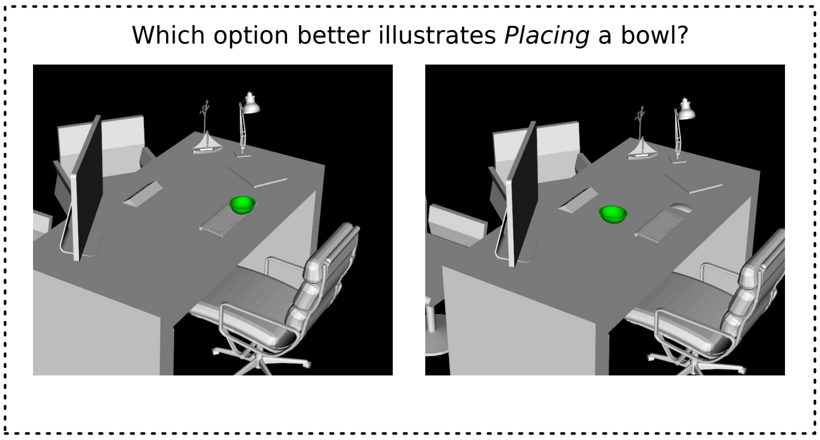


Figure 4.8: ROC plot on top shows the performance achieved when considering the every-affordance case. The plot shows **iT** considerably outperforming the baseline methods with an accuracy of 80.30% and a precision of 84.85%, with a prediction threshold of 0.52. Images on the bottom illustrate the validation task faced by human evaluators.

4.3.3 Detection on RGB-D data

Experiments were also carried out using pointclouds captured with an RGB-D camera (Asus Xtion sensor) and using a publicly available dense mapping system [210]. Additionally, publicly available data containing indoor scans [211] and five motorcycle scans [212] were considered for these affordance detection experiments. Using the same pipeline explained before, affordances are queried in a total of 20 scenes using the very same *training* example from the synthetic data. The only pre-processing step carried out to the input scenes is ground plane calibration. Figure 4.9 shows affordance heat-maps for these scenes and examples of the predicted locations.

Notably, the proposed method performs favourably with this type of data; which is attributed to the fact that the similarity function used to compared vectors can handle noisy estimates by fitting a Gaussian model to the expected difference (error) in the vectors. In this way, the method does not need further modifications in order to account for the irregular or possible noisy pointcloud from the mapping system. This again highlights the robustness of the proposed method and that fine-grained scene geometry is not essential for its performance. As an example, Figure 4.10 illustrates a close-up to the kitchen pointcloud shown in 4.9, notably the tap geometry is very noisy or inaccurate yet the **iT** method manages to predict a meaningful affordance successfully.

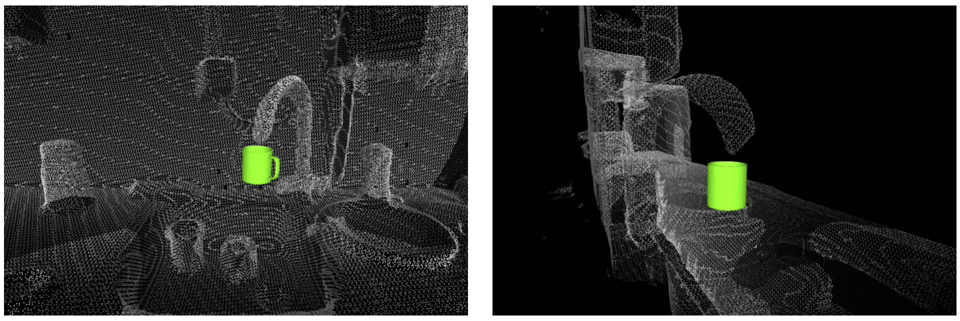


Figure 4.10: Filling a mug detection in the RGB-D scanned kitchen of Figure 4.9. Notice the noisy pointcloud for the tap, despite not being a very accurate or fine-grained reconstruction the **iT** method successfully predicts the affordance location.

4.4 Multiple-affordance detection

Up to this moment, different affordances candidate locations can be detected with the help of their specific descriptor individually . That is, if the task is to detect *Placing*-bowl affordance, the input to the system is the descriptor of this interaction and the scene. If the task at hand is to detect

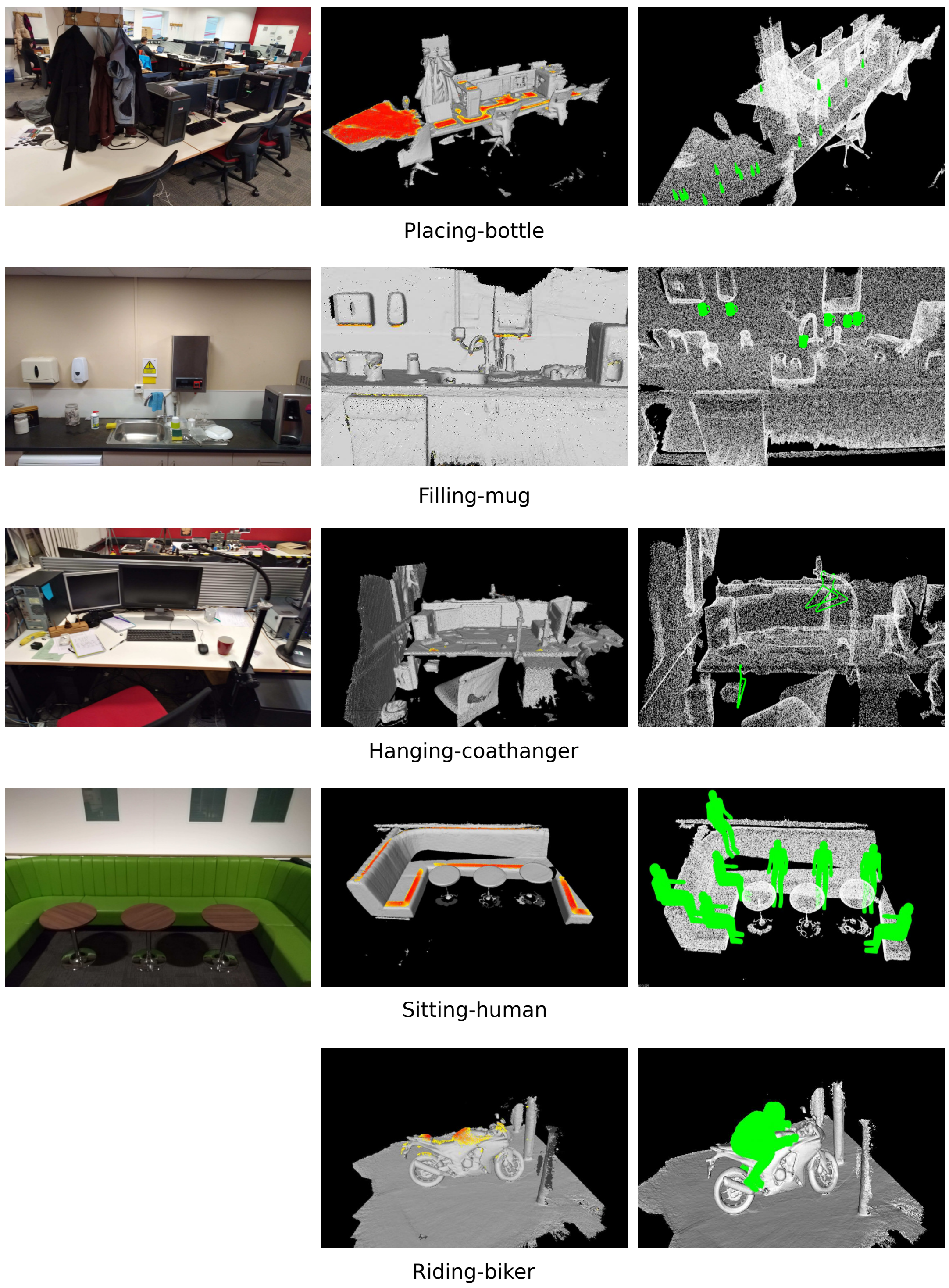


Figure 4.9: Affordance heatmaps with predicted locations in RGB-D scenes. From left to right: *placing* a bottle in office environment, *filling* a mug in a kitchen, *hanging* coat hanger in an office desk, *sitting* in a reading room and *riding* a motorcycle.

Sitting affordances, the input to the detection algorithm should change. This section presents a scalable algorithm to enable multiple-affordance detections with a single representation (i.e. one descriptor). The proposed algorithm follows the same one-shot learning approach as before in that it uses a single example from every affordance to devise the multi-affordance descriptor. With such representation and algorithm, one can give answer to questions such as “*What can I affford to do here?*" on multiple point locations of an input scene without the need to individually test affordances. The approach proposed next allows to increase the number of affordance-object pairs queried simultaneously at test time without heavily compromising detection rates.

Briefly speaking, the algorithm for multiple-affordance detection agglomerates several affordance descriptors and performs a grid-based clustering to select a reduced number of keypoints (centroids) required to make predictions at test-time. This is aimed for parallelisation and efficient evaluation. For this work a total of 92 affordance-object pairs are considered, they include CAD models of multiple household items from a wide range of geometries and dimensions, and its inspired by standard robotic manipulation datasets such as [213]. Human models for *Riding* and *Sitting* are also included in order to test "human" affordances. The specific affordance-object pairs are in Table 4.2 with information regarding their spatial dimensions (e.g. bounding box diagonal). It should be stated that only objects with bounding box diagonal was larger 10cm long were considered for this experiments; this decision was taken in the knowledge that standard RGB-D sensors would fail to recover pointclouds for such dimensions, for instance, a screw, a washer or a coin.

It is important to note that each sample interaction is an affordance on its own right, as it has been established before, this is an intimate relation between object and scene. It should also be noticed that some objects afford more than one interaction, e.g. *Fill-Pitcher* and *Hang-Pitcher*. Additionally, it is also possible to consider some top-level clustering with conventional generic labels for affordances such as *Placing, Hanging, Filling*, etc, for which performance metrics are presented as well.

4.4.1 iT agglomeration

Naively, one could detect several affordances by testing individual descriptors one after the other. However, this quickly becomes a problem, for instance in order to detect 92 affordances in a single location one would need to search for $n = 92x512x8$ nearest neighbours (92 affordances, 512

Affordance	Object	Dimension [metres]	Affordance	Object	Dimension [metres]	Affordance	Object	Dimension [metres]	Affordance	Object	Dimension [metres]
Filling	bin	0.40	Placing	bowl	0.20	Placing	hat	0.35	Placing	plate	0.25
Filling	bowl	0.20	Placing	brick	0.25	Placing	headphone-stand	0.20	Placing	pot	0.40
Filling	bucket	0.50	Placing	bucket	0.50	Placing	headphones	0.28	Placing	printer	0.61
Filling	cup	0.17	Placing	cap	0.35	Placing	keyboard	0.50	Placing	pudding-box	0.17
Filling	glass	0.21	Placing	casserole	0.41	Placing	knife	0.17	Placing	saucepan	0.41
Filling	mug	0.15	Placing	cell-phone	0.14	Placing	lamp	0.48	Placing	scissors	0.15
Filling	pitcher	0.35	Placing	chair	1.00	Placing	laptop	0.54	Placing	screwdriver	0.20
Filling	saucepan	0.41	Placing	clamps	0.18	Placing	lemon	0.08	Placing	spam-can	0.15
Hanging	cap	0.35	Placing	coat-hanger	0.47	Placing	magazine	0.39	Placing	spatula	0.35
Hanging	coat-hanger	0.47	Placing	coffee-can	0.20	Placing	marker	0.13	Placing	sponge	0.13
Hanging	handbag	0.40	Placing	coffee-maker	0.33	Placing	mug	0.15	Placing	spoon	0.17
Hanging	hat	0.35	Placing	crackers-box	0.25	Placing	mustard-bottle	0.19	Placing	stool	1.00
Hanging	headphones	0.28	Placing	credit-card	0.10	Placing	nail	0.10	Placing	tablet	0.29
Hanging	mug	0.15	Placing	cup	0.17	Placing	notebook	0.27	Placing	tape	0.12
Hanging	pitcher	0.35	Placing	detergent	0.40	Placing	orange	0.10	Placing	tea-pot	0.33
Hanging	saucepan	0.41	Placing	drill	0.32	Placing	padlock	0.11	Placing	toaster	0.36
Hanging	umbrella	0.90	Placing	flower	0.25	Placing	pc-case	0.69	Placing	tool-box	0.41
Placing	apple	0.12	Placing	football	0.22	Placing	pc-monitor	0.66	Placing	tv-remote	0.17
Placing	banana	0.18	Placing	fork	0.17	Placing	pc-mouse	0.13	Placing	umbrella	0.90
Placing	big-box	1.00	Placing	frying-pan	0.43	Placing	pear	0.12	Placing	wine-bottle	0.32
Placing	bin	0.40	Placing	glass	0.21	Placing	pencil	0.17	Placing	wrench	0.23
Placing	bleach	0.31	Placing	hammer	0.45	Placing	pitcher	0.35	Riding	biker	1.32
Placing	book	0.20	Placing	handbag	0.40	Placing	plant	0.50	Sitting	human	1.61

Table 4.2: Query-object collection used for multiple-affordance experiments. A total of 92 affordance-objects are shown with information regarding their size in meters (bounding box diagonal).

keypoints, 8 orientations); which would imply in the best case scenario $\mathcal{O}(logn)$. Alternatively, and is presented below, one could take advantage of the overlapping pattern found when many affordance descriptors are agglomerated in a single location.

The general idea behind the proposed method, i.e. the **iT** agglomeration, is to leverage the overlap or pattern that appears when affordance keypoints from multiple descriptors are stacked (or agglomerated) on top of each other. By taking advantage of this overlap among keypoints one can efficiently group or cluster neighbouring keypoints, which allows to drastically reduce the dimension of the points in a descriptor that encodes multiple interactions.

As a starting point and similarly to the work presented earlier, interaction tensors are computed and affordance descriptors are obtained by sampling $N = 512$ keypoints from their associated **iT**. Appendix A.3 shows all the 92 interactions tensors and the object models used to generate them.

For this time, instead of rotating the descriptor at test-time to detect affordances at different orientations, every individual descriptor is rotated 8 times around the gravity vector from the beginning. This produces a representation that already accounts for the interaction at 8 orientations from the outset. In this manner, every affordance is represented by a 4096 keypoints (points on the IBS and provenance vectors, i.e. 4096x6). In addition to allow for the exploitation of overlapping patterns, this new 8-orientations representation enables to leverage hardware parallelisation during tests and perform a more efficient clustering during *training*.

Once all the descriptors have been computed, they are agglomerated in a single pointcloud on which clustering is applied following Algorithm 2. First, a grid of uniform-size cells is fitted to the pointcloud, making sure that every single *affordance keypoint* is inside a cell. Then, the centroids of non-empty cells are used as seed-points for clustering. For every one of these cells, the closest keypoint to the cell centroid is kept on a per-affordance manner. For instance, one cell could contain 100 keypoints, all coming from the descriptor of *Placing-bowl*; after the **iT** clustering process is carried out this cell will only contain the one keypoint closest to the cell's centroid. Finally, the centroid locations are updated by considering the keypoints within each cell. Additionally, information regarding the number of keypoints and their associated *provenance vectors* is stored for every cell. Figure 4.11 depicts the cell-updating process for the **iT** clustering algorithm (steps 6-9 of Algorithm 2)

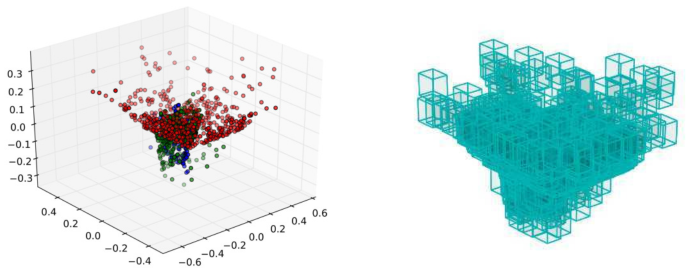
Algorithm 2 iT clustering

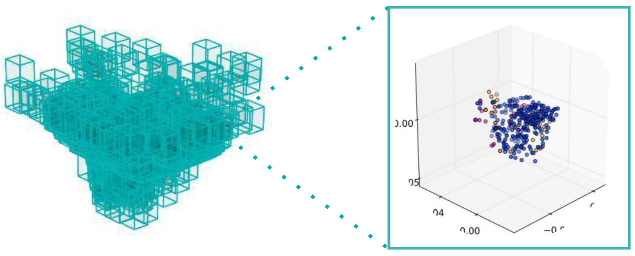
```
Input: Affordance keypoints $X = \{x_1,...,x_i\}$, cell size $e$
Output: Cluster centroids $C = \{c_1,...,c_j\}$
1: Initialize C with centroids evenly distributed in $[x_{\min}, x_{\max}]$ according to $e$.
2: Assign $x_i$ to cluster $argmin_j \|x_i - C_j\|_2 \; \forall x_i \in X$
3: Remove empty clusters
4: Initialize update sets $Y_1,...,Y_j$ to empty
5: for all Clusters $C$ do
6:     for all Affordances $A = \{a_1,...,a_k\}_{\neq}$ in $C_j$ do
7:         Recover all $x$ from affordance $a_k$
8:         Assign $argmin_i \|x_i^k - C_j\|_2$ to $Y_j$
9: Update centroids: $c_j \leftarrow \frac{1}{|Y_j|} \sum_{y \in Y_j} y$
```

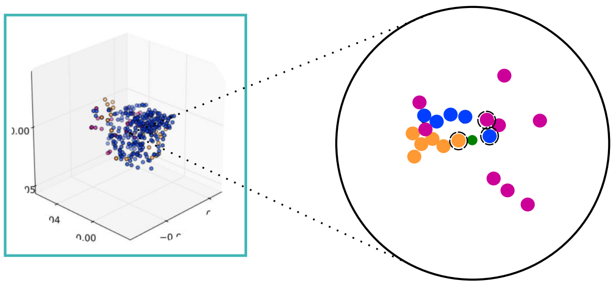
The clustering process leads to a reduced number of 3D points (cell centroids) that represent a large number of affordance keypoints. This reduced number of new keypoints and their associated *provenance vectors* are used to compute and predict affordance candidate locations at test time in a similar manner as before. As discussed earlier, the focus is now in answering the question of "*What can be afforded here?*" or "*What can I do here?*" in any given location of an input scene. In order to answer that for up to 92 affordances ($k = 92$), an approach similar to single affordance queries (Algorithm 1) is followed, that means:

1. Uniformly randomly sample a test-point in the input scene

2. Apply the transformation (translation) to the descriptor to align it relative to the test-point


a)


b)


c)

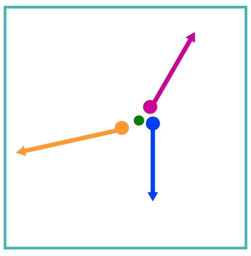

d)

Figure 4.11: Agglomeration of affordance descriptors and grid-based clustering of an example agglomeration of 3 affordances. a) single-affordance keypoints are agglomerated (affordances shown as different colours) and a uniform-size cell grid is fitted, b) one cell can potentially contain many keypoints from multiple affordances, c) only closest keypoint (per-affordance) to the cell centroid (green) is taken into account during the update process, d) an updated cell with the provenance vectors associated to the keypoints kept after clustering.

3. Perform a 1-NN search for every keypoint in the agglomerative descriptor using the voxel surrounding the test-point

4. Estimate *test-vectors* and compare against *provenance vectors* to produce a score using the similarity function shown in Equation 4.3 below

$$s^k = \frac{1}{N^k}\sum_{i=1}^{N^k}\frac{1}{\sqrt{2\pi(w_i^k)^2}}e^{-\frac{(\Delta_i^k)^2}{2(w_i^k)^2}}, \tag{4.3}$$

with

$$\Delta_i^k = \frac{\|\vec{v}_{tj} - \vec{p}_i^k\|}{\|\vec{p}_i^k\|},$$

where w_i^k is the weight of i-th keypoint of affordance k, the value of the weight is computed from the *provenance vector* associated with that keypoint. Δ_i^k is the difference between test vector $\vec{v}_{tj}$ (estimated using the j-th cell-centroid) and provenance vector p_i^k. Figure 4.12 depicts the steps followed at test-time in order to predict affordance candidate locations.

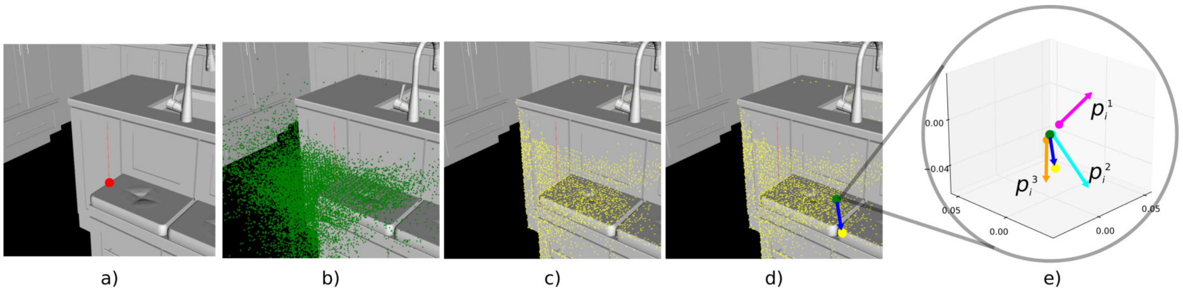


Figure 4.12: Illustration of affordance prediction at test-time. a) a test-point is sampled from the input scene (red), b) the agglomerative representation (green) is aligned relative to this test-point, c) The closest scene point (yellow) for every centroid in the agglomeration, d) an example test-vector (blue) from a cell centroid to its closest scene point, e) test-vector is compared against the stored provenance vectors p_i^k associated with affordance keypoints in that cell. In this particular cell, three scores are obtained.

Notably, the approach is very similar to the previous method (i.e. single affordance query), where a NN-search forms part of the main pipeline in order to estimate test-vectors. In order to make such approach more efficient, the NN-search and the scoring (i.e. vector comparison) are implemented using parallel computing libraries (e.g. CUDA and PCL::GPU [214, 215]). First, the nearest neighbour search is optimized by building an Octree structure with the scene pointcloud. The access to this data structure is shared across all computing threads inside the GPU; therefore, a single call of the NN-search function is able to provide up to 2048 nearest neighbours (e.g. Nvidia Titan X). Once the 1-NN for every cell centroid is computed, the data is shared with the scoring module (i.e. vector comparison) which also runs in the GPU. In this module, one GPU thread computes Equation 4.3 for one cell; meaning that 2048 scores can be computed in a single run this module.

Remarkably, the method results straight-forward to compute thanks to the compact, robust and efficient affordance descriptor presented in Chapter 3. In the following subsections are presented the evaluations and experiments carried out to test the agglomerative multiple-affordance representation that has been just described. These experiments and evaluations include the predictions rates, the validation of the affordance candidate locations and the evaluation of the parameters involved in the method, e.g. cluster size.

4.4.2 Evaluation

Cluster size and detection rates

One important parameter of this approach for multiple-affordance prediction is the cell size employed to cluster *affordance keypoints*. This is somewhat a compromise of parallelisation capability and framerate operation. One first consideration of this work explored non-uniform spatial representations of the keypoint agglomeration, representations such as those in e.g. Octrees. However, the diversity in dimension and sparsity of the affordances considered in this research made very challenging the selection for the right positioning of centroids, which did not perform as well as sparse yet uniform-sized cells.

In principle, the smaller the cell gets, the closer it is to the original descriptor since in the limit every cell will contain only one keypoint; in practice, the smallest cell size to study during experiments is obtained following Algorithm 2 with a single-affordance descriptor (i.e. 4096 keypoints). First, a cell size of 2 cm^3 is used, then the cell size is gradually reduced until the number of clusters remains stable. Figure 4.13 shows that at a cell size of 0.5 cm^3 the number of clusters remains roughly the same. This figure also shows the size in meters of the diagonal of agglomeration's bounding box as more affordance descriptors are added, the 92 descriptor agglomeration occupies a volume of approximately 2.7 cubic metres.

The affordance query algorithm for multiple affordances is executed on the scene dataset introduced previously, which is comprised of 20 RGB-D scans of indoor environments and 15

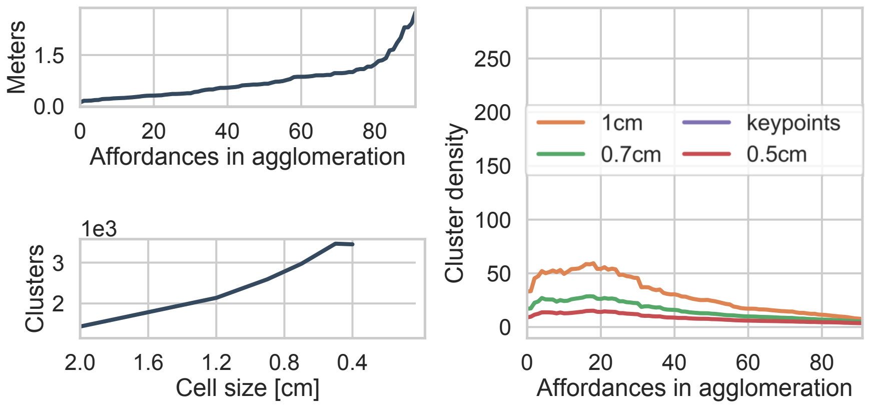


Figure 4.13: Plot on the top-left shows the increment in the size of the agglomerative representation (bounding box) as the number of affordances contained in it grows. The plot of the bottom-left shows the number of clusters produced by considering various cell sizes in single affordance agglomeration. The plot on the right shows the per-cluster density (number affordance keypoints) for various cell sizes.

synthetic scenes from publicly available CAD models. As a reminder, these scenes consists of living-rooms, kitchens, offices and motorcycles. Figure 4.14 shows the dimensionality of the multiple-affordance representation and the average prediction rates according to the cell size. Looking at this figure, it stands out the large reduction that is achieved with the proposed

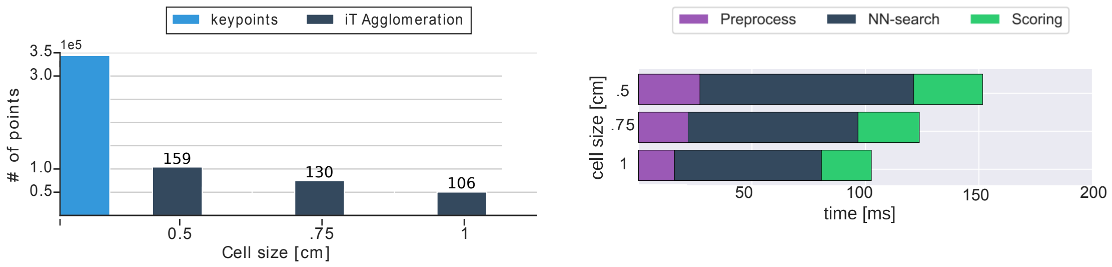


Figure 4.14: Bar plot shows the dimensionality reduction achieved with the agglomerative method for different cell sizes. The number of keypoints required to make predictions is reduced up to 6 times. Numbers above each bar show prediction time (milliseconds) per test-point of the input scene.

approach, which is nearly six times smaller (344K vs 60K keypoints). The prediction rates on the same figure show that using grids with a cell size of 1 cm^3 allows the detection of up to 92 affordances at 10 different locations per iteration on the input scene. This is significantly faster (7x improvement) than predicting affordances by trying descriptors individually at test time one after the other. Because the prediction algorithm performs a NN-search in order to estimate test-vectors and compare them against provenance vectors, the complexity of such operation depends heavily on the dimension of the multiple-affordance representation (i.e. the number of centroids/keypoints). More points in the representation require more computations; thus, reducing the representation allows for faster evaluations at test-time. Even with such a reduction in dimensionality the approach, as shall be shown later, is able to produce top quality affordance predictions.

In an effort to further emphasise the scalability of the **iT** agglomeration method, Figure 4.15 shows the computation times observed during affordance predictions of 92 affordance-object pairs. The green curve in this figure corresponds to the computation time measured by progressively incrementing the number of affordances represented by an agglomerative representation of 0.5cm^3 cells. That is, the time shown in the far right corresponds to an agglomerative descriptor of 92 affordances, whereas the first value on the left corresponds to an agglomerative descriptor formed by agglomerating keypoints of 1 affordance (*Sitting*-human). Observe that the time grows sub-linearly on the number of keypoints added to the agglomeration. This figure also shows the time required to predict affordances by testing individually one after the other, which requires approximately 644 ms per test-point in the input scene.

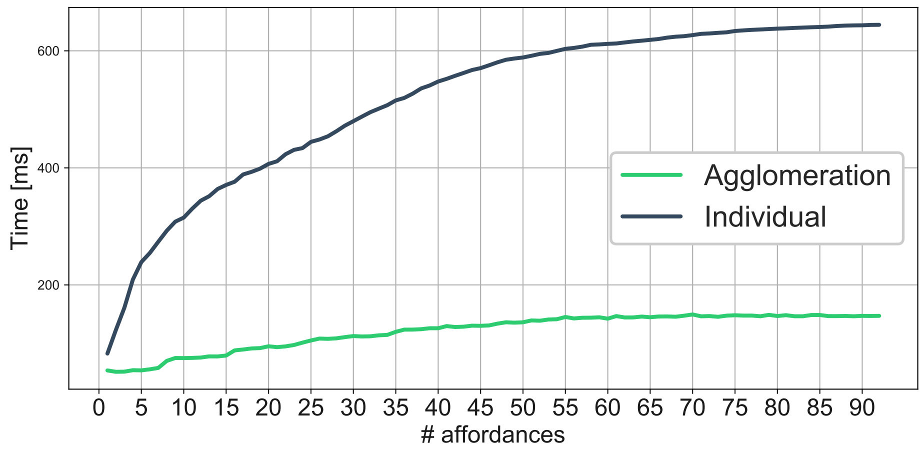


Figure 4.15: Plot in green shows the computation time (milliseconds) observed by testing an agglomerative descriptor of different sizes (number of affordances) on one thousand randomly sampled test-points. In dark blue, the plot shows the time measured when predictions are performed by testing individual affordance descriptors one after the other.

iT agglomeration vs Single affordance prediction

Previously, the good performance of the **iT** method for detecting affordance candidate locations on an individual basis, i.e. querying one affordance at the time, was validated via crowdsourcing. This information is leveraged in order to asses the performance of the agglomerative approach. In this sense, the affordance predictions made with the agglomerative representation were compared against those predictions produced in a single-affordance scenario. This was done with the intuition that achieving a good performance with this baseline would reflect in the predictions of meaningful affordance location according to human criteria.

First, Algorithm 1 (single affordance query) is executed in order to produce affordance predictions for every affordance-object pair in Table 4.2; these predictions are then treated as "ground-truth" in order to compute performance metrics for the multiple-affordance approach based on the agglomeration of **iT**s. Notice, however, that this dataset does not strictly constitute a set of target locations to be classified or discovered; however, the good performance achieved by the method in single affordance predictions motivates to investigate if the newly proposed multiple-affordance representation is able to produce predictions of a similar nature (i.e. quality). Later on, this is further corroborated via human validation.

Figure 4.16 presents the performance achieved for all interactions under investigation. In this figure can be seen that 1cm-cells perform better for *Sitting* and *Placing* of medium-to-big objects, the exceptions being *Placing* pc-case. All cell sizes seemed to struggle with *Placing* coat-hanger, fork, tv-remote and pc-mouse. Also, notice that for all *Hanging* and *Filling* affordances predictions the precision is roughly the same for the three agglomerative representations. Another interesting result is that *Placing* smaller objects such as a knife, scissors, plate, spoon and pencil are predicted more reliably with the smallest cell size. This is explained by the fact that **iT***s* of smaller objects are comprised of shorter *provenance* vectors; these vectors are not well represented when the cell size is increased. As suspected, smaller objects require a more fine-grained representation, such as the one achieved with 0.5 cm cells.

It should also be noted that Figure 4.16 shows only 84 interactions. During tests, it was found that *Placing* thin and flat objects, such as a magazine, a credit card, a mobile phone or a tablet, required many short (less than 1 cm long) and vertical *provenance vectors* (e.g. under the objects). This type of vector is hard to match due to the density of the pointclouds used for testing.

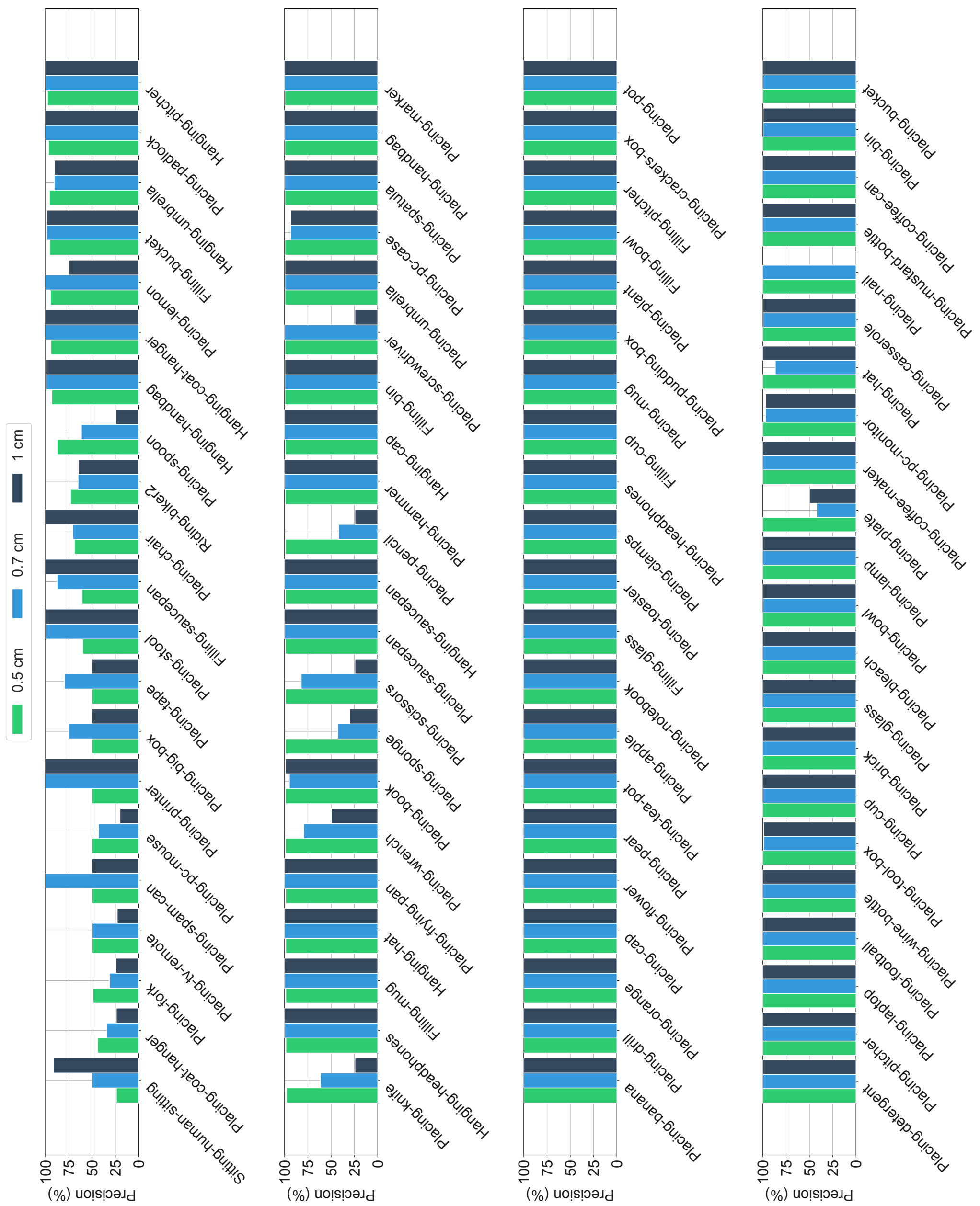


Figure 4.16: Precision achieved with agglomerative representations produced by cell sizes of different sizes.

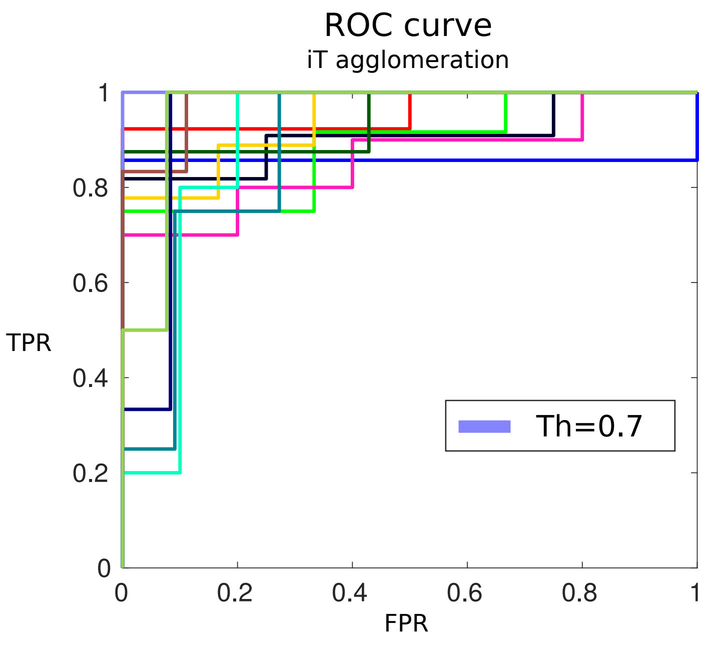
Table 4.3 shows performance metrics for top-level generic affordances and the average performance of the agglomerative approach for various cell sizes. It can be seen that the agglomerative approach overall performs best for *Filling* and *Hanging* affordances, where every single prediction made with the agglomerative representations was also a good location for the single affordance baseline. It is also worth noticing that *Riding*, which is regarded as the more complex interaction in this study, has the best performance with a cell size of 0.5 cm^3. In contrast, the predictions for *Placing* affordances, which are arguably the least complex interactions, are better when a larger cell size is employed. This can be explained by the fact that *Placing* an object relies on vectors located under the objects; these vectors are very small (i.e. millimetric) when the cell size gets smaller. Vectors estimated a test-time rarely present such magnitudes due to the density of the scene pointcloud. In other words, agglomerative representations of larger cell sizes comprise larger *provenance vectors* that are more easily matched during test-time.

	0.5cm			0.75cm			1cm		
	Precision	**Recall**	**F-1 score**	**Precision**	**Recall**	**F-1 score**	**Precision**	**Recall**	**F-1 score**
Filling	94.28	49.44	0.6261	98.19	27.55	0.4224	99.74	5.6663	0.1063
Hanging	97.08	18.11	0.2792	98.69	10.17	0.1724	98.69	2.2375	0.0418
Placing	92.34	59.48	0.6853	90.48	32.44	0.4613	84.13	5.5334	0.0954
Riding	73.24	60.70	0.6646	65.00	47.26	0.5473	64.30	33.51	0.4406
Sitting	23.85	16.76	0.1968	50.00	14.95	0.2302	91.57	13.15	0.2300
Average	76.16	40.90	0.4904	80.47	26.48	0.3667	87.68	12.02	0.1828

Table 4.3: Average performance for different cell sizes.

Optimal detections

As discussed earlier, affordances predictions on their own are elusive to ground-truth without subjective judgement or evaluation of the likelihood of an interaction. In this work, affordance location predictions are made by setting a threshold to the output (score) of the algorithm. Amazon Mechanical Turk is used to determine the threshold that produces the best results. Here, people are asked to evaluate the predictions made with the agglomeration algorithm based on the smallest cell size. A total of 2.4K example predictions representing different scores were shown to 42 humans evaluators (turkers). These subjects had to select a "winner" from two possible options showing the same affordance-object pair resulting from different scores. A "true" ranking based on human evaluation is computed by fitting a Bradley-Terry model [216] to the pairwise comparisons, with this ranking the performance of the **iT** agglomeration algorithm was assessed and the optimal threshold that resulted in the optimal detections was obtained. Figure 4.17 shows the family of classifiers induced by setting different threshold values at the score of the **iT** agglomeration and algorithm.



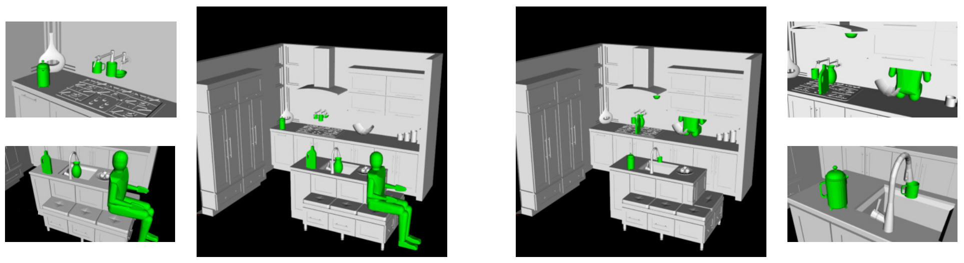


Figure 4.17: Mechanical turk evaluation. ROC plot shows the family of classifiers generated by setting different thresholds bands to the prediction score; the best result is obtained with a value of 0.7. Images on the bottom exemplify the type of judgement humans had to make: choosing the image that best depicts the interactions (query-objects in green).

The ROC plot shows that the method achieves a good performance according to human criteria when considering predictions made with a score above 0.7. In other words, the affordance predictions with a score above this threshold are deemed as good candidates according to humans all the time. This contrasts with the value found in earlier experiments for predictions made with individual descriptors. The increase in the threshold value can be explained by quantisation error introduced by the agglomeration. This error would cause false positives to appear if the threshold was set to 0.52 as was previously found. However, by considering predictions on the top end (i.e. score >=0.7), the method detects meaningful affordance locations.

4.4.3 Predictions on RGB-D data

This subsection presents qualitative results of multiple-affordance predictions performed on a publicly available RGB-D scans dataset, i.e. ScanNet [217]. This dataset is comprised of more than 1500 scans from which 150 are randomly sampled to illustrate affordance predictions. In addition to these, the five motorcycle RGB-D scans introduced in section 4.3 are also included to show example predictions for *Riding* affordances. Overall, the scenes used to query multiple affordances include living-rooms, kitchens, offices and lounges. It should be kept in mind that these predictions are made using agglomeration of single example iTs (i.e. one-shot) and that the input scenes are never before seen environments. Figure 4.18 shows examples of top predictions made by the agglomerative approach. These images were generated offline by considering top score predictions and checking *placements* of query objects that are free of collision. The reason for this post-processing generation of images is that the prediction algorithm has the potential to detect many interactions in a single location, the job of deciding on-line ”what happens where? “ is regarded as a different problem. However, recall that these type of affordance predictions are made at a fast rate (e.g. 150 ms per test-point).

4.5 Conclusion

This chapter presented the proposed approach for one-shot affordance detection in novel scenarios. The affordance descriptor and the similarity function allow to detect up to 84 affordances in just over one second at any point location of the input scene. The affordance hypotheses generated with the proposed method have been validated using crowdsourced human evaluations, which showed that on average the methods agree with human criteria 92% of the time. Evaluations also showed that the algorithm based on the **iT** representation outperforms alternative approaches, such as those in computer graphics that inspired the proposed method. By taking advantage of a state-of-the-art publicly available dense mapping system, multiple affordance detections can be produced in real indoor environments.

In the following chapter, it is shown that the combination of the **iT** approach with a data-

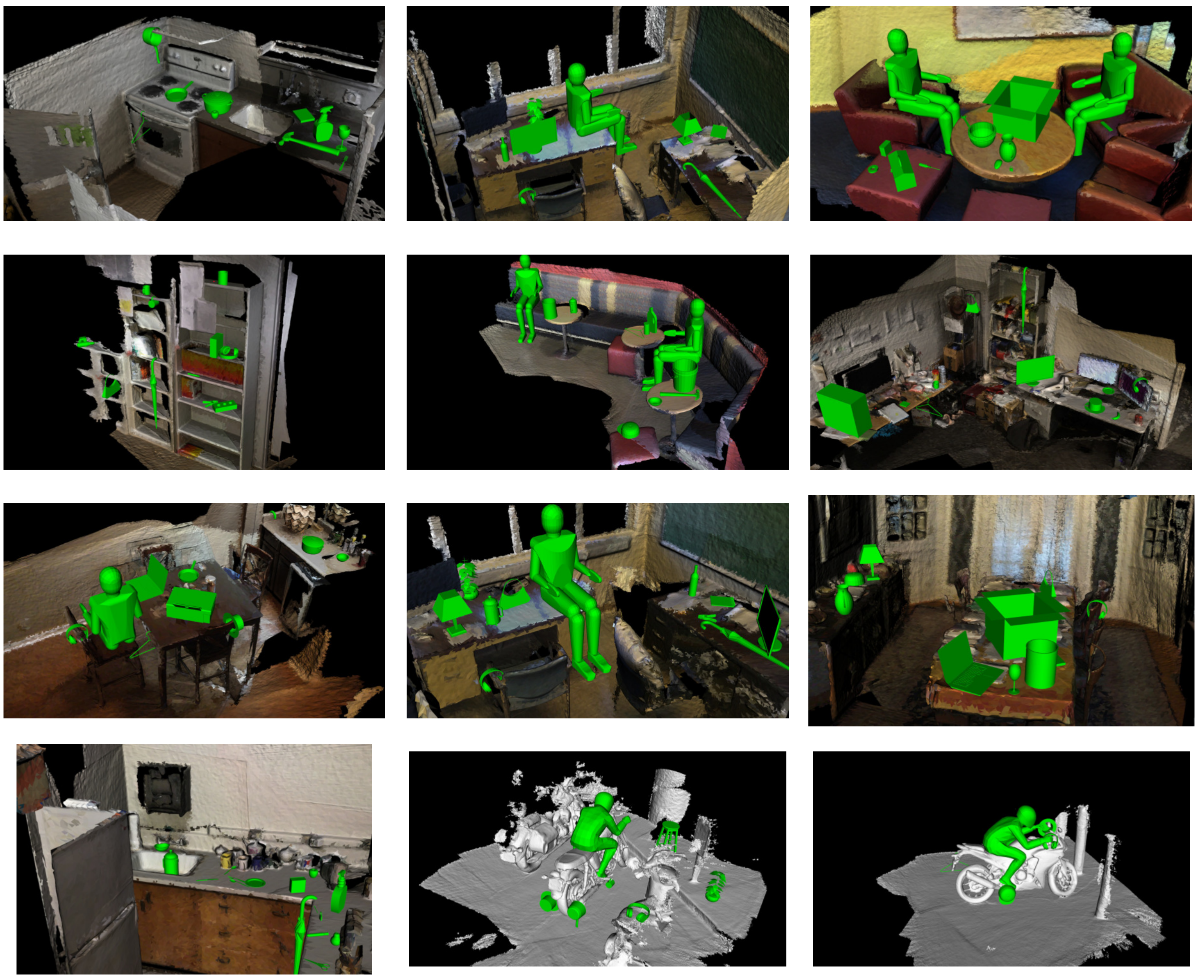


Figure 4.18: Multiple-affordance detection on never -before seen environments.

driven method allows to: 1) overcome the limitations produced by implementing a one-shot learning approach , and 2) optimise the representation to allow faster and top-quality affordance detections.

CHAPTER 5

LEARNING AFFORDANCE DESCRIPTORS

5.1 Introduction

The one-shot **iT** approach presented in this thesis is inspired by Gibson's call for *direct* and economical affordance perception methods. This, in addition to the lack of 3D data for the type of affordances investigated in this thesis, lead to compromises such as empirically-defined parameters in the proposed method, namely the location and number of affordance keypoints in the descriptor. The current chapter presents a method to optimise a multiple-affordance representation for 3D data by leveraging the abstraction power of current deep learning techniques, specifically a state-of-the-art deep neural network that consumes 3D data. First, this chapter explores and evaluate the performance of one deep learning architecture when applied to the affordances that this thesis investigates. The selection, evaluation and comparisons of the method are motivated by the increasing number of approaches for visual perception employing deep learning techniques; the comparisons against the **iT** approach are followed by the introduction of a hybrid algorithm that leverages the ability of the neural network to learn salient locations in the input and the *generative* capabilities of the interaction tensor representation. Briefly speaking, the hybrid approach produces faster and meaningful detections as it uses an optimal and compact description of multiple affordance-object pairs. Specifically, the work presented in this chapter starts by providing the motivation and details regarding the use of a deep learning method over alternative machine learning approaches. Then, results are presented showing the underwhelming performance of the deep network on its own when used for the *traditional* classification task. Later, details are provided regarding the combination, experiments and evaluation of the proposed methods.

5.2 Learning on pointcloud data

Besides aiming to formulate a simple and direct method for affordance perception, one more reason that motivated a one-shot learning approach was the lack of 3D data for the type affordances that this thesis investigates. When available, and as discussed in Chapter 2, annotated data is for very specific interactions such as *human* affordances or object-part affordances for grasping. One of the shortcomings of adopting a one-shot learning approach with a handcrafted representation is the often likely sub-optimal selection of the parameters in the method or the difficulty of finding the best possible values. One way to overcome such issue is by having humans evaluate or "label" data to then optimise with those annotations as a target. Using large collections of annotated data to learn features or an optimal affordance representation has been a common approach in robotics and computer vision; those data annotations, however, turn out to be costly and time-consuming to produce.

As shown in the previous chapters, the **iT**'s hand-crafted nature relies on sampling affordance keypoints in order to form a descriptor. The location and number of such keypoints have been determined empirically. Whereas a sparse and empirically-found sample has shown to work well, improvements can be made by investigating and learning based on the ability of the method to produce multiple affordance candidate locations from a single *training* example. That is, learning from the data generated with the proposed method; which, as shall be detailed later, allows to efficiently sample and obtain insight into the optimal representation of affordance descriptors.

In Chapter 2 was shown that approaches for affordance learning have followed two paths in order to learn a mapping or representation: learning on hand-crafted features or more recently deep learning. The former is concerned with first computing a set of features on the available data to then apply machine learning algorithms; the latter attempts to learn features *directly* from the data. Notably, deep learning architectures have recently been one of the dominating techniques in visual perception since they can overcome the non-trivial problem of selecting a *good* representation from the available data (i.e. features). Naturally, recent works in affordance perception and learning have also taken advantage of the outstanding progress of deep neural networks; particularly the great progress in object detection, recognition and segmentation (see Section 2.4.3 of Chapter 2 for details). Generally speaking, these techniques (deep networks) allow to learn the object's features (or object-part) that *afford* certain interaction; however, approaches using this techniques have been limited to 2D representations of the world or well structured data (i.e. 2D images). Only recently, deep learning architectures have started to work with 3D input aiming to catch up with the progress achieved on 2D imagery. Examples of recent works in deep networks for 3D input are[218–225], which use voxelization to transform irregular data into occupancy grids that allow 3D convolutions to be applied. This quantisation or voxelization is due to the fact that typical deep learning architectures require regular input formats, a requirement that does not hold for pointcloud data.

Pointcloud data structures are sets of unordered data points and only recently researchers have made efforts to bring to the pointcloud domain the power that deep networks have shown on images. Among these latest approaches are [226–229], which present deep learning architectures for tasks such as object classification, object-part segmentation and scene semantic segmentation. These architectures seem to cope well with pointcloud irregular and unorganized nature, achieving impressive results in benchmarking datasets. Specifically, the PointNet+++ architecture [228] has shown a remarkable performance for shape classification on 3D data due to the architecture's ability to capture local structures induced by the metric space points live in as well as its robustness against non-uniform sampled (e.g. noisy) pointclouds.

First, utilizing a deep learning architecture for the data of the affordances under investigation serves as a baseline analogous to state-of-the-art approaches to affordance learning (e.g. [144, 171]), approaches who let the deep networks learn the mapping from input data to an affordance label or class by exploting large collections of annotated data. The advantage of the recently proposed PointNet++ architecture is that it allows to devise such baseline algorithm for the 3D input (e.g. pointcloud) with which the **iT** works; thus, comparisons and evaluations can be made in the same domain. Furthermore, authors of PointNet++ have made publicly available implementation details and the data needed for testing and training the architecture, which makes the comparison more fair and less prone to implementation errors. Additionally, considering a deep learning approach allows to bypass the selection and evaluation of feature representations that other machine learning methods would require; thus, enabling to work directly on the "raw" pointcloud representation similarly to the **iT** method.

One more motivation to employ a deep learning architecture is the big amount of data that can be generated with the **iT** method. These pointcloud data can be *easily* obtained by running the algorithm over multiple scenes and storing the pointclouds related to the predicted interactions. Moreover, the evaluations and validations presented in the previous chapter allow to use these predictions (above the optimal threshold) as ground-truth for the training and testing processes (i.e. annotated data); in other words, the large collection of data needed by the deep network approach can be automatically produced with the **iT** method.

In the following subsections is presented the series of experiments carried out in order to evaluate such deep learning approach for affordance learning on pointclouds, this includes details on the PointNet++ network, parameter values and experimental setup as well as results on a first comparison against the **iT** methods for single and multiple affordance determination (e.g. pointcloud classification). Later on, this chapter introduces the steps taken in order to leverage abstraction power of the PointNet++ architecture and the geometric features captured by the interaction tensor.

5.2.1 The PointNet++ architecture

PointNet++ is a deep neural network architecture that processes points in a hierarchical fashion. The architecture extracts local features from all over the input pointcloud by partitioning it into overlapping regions. This partitioning is based on the distance metric of the underlying space in which points are defined. The extracted features are meant to capture fine geometric properties from neighbourhoods all around the input; these features are then grouped into a more abstract representation and further processed to produce high-level representations. In order to obtain the attributes characterising the complete point set, the process (sampling and grouping) is repeated at different scales or hierarchies all over the pointcloud. In doing so, the PointNet++ architecture is able to improve on alternative methods by dropping the assumption that pointclouds are uniformly sampled and by relying on features extracted and processed from multiple regions and hierarchies, instead of single max pooling operations to aggregate the whole point set such as previous approaches [226].

Figure 5.1 shows an overview of the PointNet++ classification architecture, the numbers above each block represent the dimensionality of the input and output for each layer, which will be detailed later on (implementation details section). There are three main building blocks or layer types in the architecture: Sampling, Grouping and PointNet.

Sampling layers select points which are used as centroids of the regions from where features are extracted. Specifically, PointNet++ implements a farthes point sampling technique which ensures a maximal coverage over the input pointcloud,this also ensures that the samples are evenly distributed all over the *surface* of the pointcloud. Keep in mind that the architecture relies on computing features from overlapping regions.

Grouping layers gather data by sampling points within local regions, these local regions and their centroid data are concatenated to build new (internal) point sets which form a hierarchy. The size of the local regions is controlled by setting a radius which is then used to group data within the spheres centered at the points sampled in the previous layer.

PointNet layers is where the regions' data is encoded and represented in terms of centroids and local features in the form of vectors. The layers are inspired by the work of Qi et al. [226], a pioneer approach in deep networks for pointcloud data, an architecture that learns local patterns across the whole input. The PointNet layers or modules are composed of a series of convolutions, pooling operations (e.g. a multi-layer perceptron) that allow to transform and abstract the information contained by the pointsets within local regions. Formally, the pointnet modules are a function that maps an unordered set of points $\{x_1, x_2, ..., x_n\}$ with $x_i \in \mathbb{R}^d$ to a vector $f : \mathscr{X} \to \mathbb{R}$

$$f(x_1, x_2, ...x_n) = \gamma(MAX_{i=1,..n}\{h(x_i)\})$$

where γ and h are a multi-layer perceptron. The set function f is invariant to permutations in the input and the h can be interpreted as the spatial enconding of a point. For full details on the PoinNet classification architecture of Qi et al. see [226].

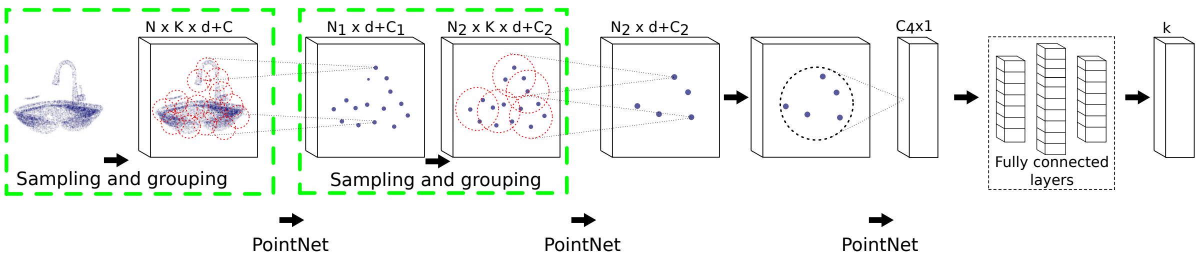


Figure 5.1: The PointNet++ architecture [228]. Red circles illustrate sampling regions, which are used in the set abstraction levels (green) to build hierarchical features. The output is a set of k scores or probabilities for the classes present in the data.

5.3 Data collection

One first difficulty is gathering the large amounts of data that the deep neural network requires in order to train. Conversely, one of the benefits of the **iT** method is its generative capabilities. In other words, the **iT** method allows to obtain, *training* from a single instance or single example, multiple instances of pointclouds that afford a target interaction. For instance, one could run the affordance query algorithm on a living-room scene and obtain thousands (potentially) of pointclouds that afford *Placing* different objects. This is exactly the approach followed to acquire enough affordance data to train a deep network.

It should be mentioned that for this part of the work only the affordances that were actually (more than 1K detections) predicted by the **iT** agglomeration method in the previous chapter are considered. That is, all affordances in Table 4.2 except for *Hanging* mug, *Placing* cellphone, *Placing* credit-card, *Placing* headphone-stand *Placing* magazine, *Placing* tablet. Also, as explained previously, the *Riding* affordance is no longer found for indoor scenes when multiple affordances are to be predicted; this interaction is also dropped out of the collection. This means that for the current part of the work a total of 84 affordance-object pairs are considered for the experiments. As shall be shown later, this allows for a fair comparison among all the methods.

Using the set of scenes introduced in Chapter 4 comprised of 20 synthetic scenes and 20 RGB-D scans, a large dataset is built by running the affordance query algorithm for 84 affordance-object pairs individually. Figure 5.2 shows the data distributions according to interaction and scene type.

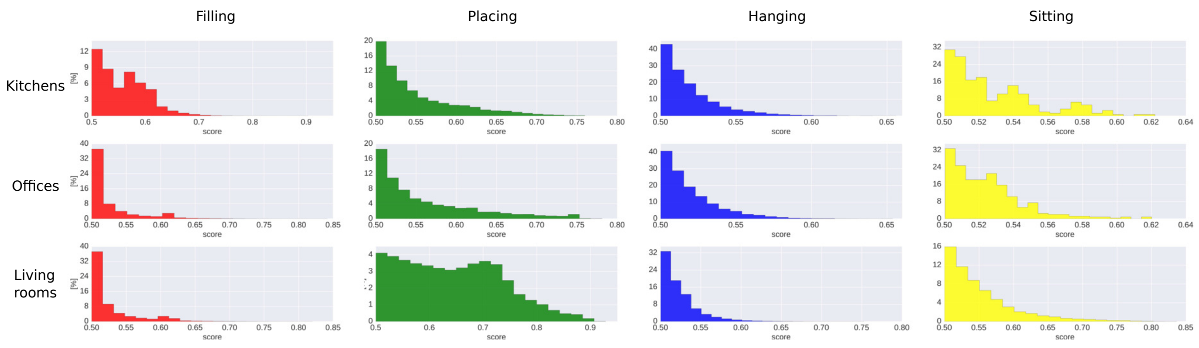


Figure 5.2: Affordance data distributions according to the type of scene and interaction.

By default, the output of the affordance query algorithm is the probability of a point location in the scene of affording the interaction. The pointcloud that actually allows the interaction to take place in that location is obtained by extracting the voxel surrounding the test point detected as a good location. That is, voxels are computed just like it is normally performed during affordance query algorithm, i.e. the voxel inside a sphere of radius equal to the query-object size (diagonal of the bounding box). Although, due to the fact that the aim is to represent multiple affordances for every pointcloud, the size of the voxel extracted for every affordance prediction is regulated by the largest query-object in the collection: the human model for *Sitting* affordances.

A dataset is formed by sampling up to 10K pointclouds per interaction with the pointclouds associated with affordance predictions. Additionally, for each affordance dataset, background or "negative" examples are generated from affordance detections made with a score lower than the optimal value introduced in Chapter 4 (i.e. $s \leq 0.5$). Training and validation sets are created by considering an 80/20 split on these data.

Pointclouds come from several scenes which most of the times have coordinates defined w.r.t. to different coordinate systems or world origin. In order to have every pointcloud in the dataset within the same coordinate system, all pointclouds are "zero-meaned" by translating the centre of the voxel to the point (0,0,0). In this way, every pointcloud is centred in the origin which, as shall be shown later, allows to conveniently track the data across multiple affordances from different scenes. At the end of the process, a large affordance dataset is gathered which contains 84 binary collections with approximately 10K examples each. This data is used to train and test the deep network architecture.

5.4 Deep learning on affordance data

5.4.1 Implentation details

The following two subsections describe experiments and show results of the evaluation and comparison of the deep learning architecture for affordance data. For all sets of experiments (unless otherwise stated) training is carried out using the Adam optimiser with initial learning rate of 0.001, momentum 0.9, batch size 32 and a decay rate of 0.7. Batch normalisation is used for every fully connected layer and dropout with a keep ratio of 0.5. Training is carried out for at least 250 epochs and until convergence is achieved. All layers (except for the last) work with ReLU activations and have dimensionality parameters as shown in Table 5.1 (see Fig 5.1 for correspondance).

Parameter	**Value**
N	512
N_1	128
N_2	256
K	Variable according to r1 and r2
d	3
$C = C_1 = C_2 = C_4$	0
r_1	20% of input
r_2	40% of input

Table 5.1: PointNet++ layers dimensionality parameters

Note that K corresponds to the number of points taken into account to extract features in each local region of the input pointcloud, this number varies according to the pointcloud density at that particular region and the size of the spheres used to extract such regions r_1 and r_2. Where r_1 is the radius for the local regions at the lowest level of the hierarchy and r_2 at the highest; both expressed in terms of the bounding box' diagonal of the input pointcloud. Both training and testing are performed in a single Titan X GPU similarly to experiments with the **iT** method of the previous chapter (Chapter 4), i.e. same PC specifications.

5.4.2 Affordance learning as shape classification

The first experiments carried out were designed to investigate the performance and behaviour of PointNet++ on individual affordances. In this regard, the first set of experiments correspond to a *conventional* classification task (e.g. shape classification) where the neural network is presented with an example pointcloud and its corresponding binary label. Thus, the label indicates whether the current pointcloud belongs or not to the affordance *class* being trained. One important observation should be made though; most 3D shape classification approaches normalise the training data to a unit-sphere or unit-box; this is not feasible in the 3D affordance learning approach studied in this work given that the interactions studied here are in real-world scale. To put it differently, having a pointcloud of a chair of 1 meter-height is substantially different from a *toy chair* with a height of 10 cm; the latter would not afford *Sitting* for a human adult. For this reason, and in contrast with [228], the network architecture is modified to allow the sampling regions to change proportionally to the current training pointcloud. These regions (red circles in Figure 5.1) are used by the network to learn features at different hierarchies and pointcloud densities. The value of this parameter is set to be 20% at the lowest level and 40% at the highest (relative to pointcloud bounding box). Other than parameter tuning, no further modifications are made to the network for this set of experiments. Figure 5.3 shows the performance achieved by the network when trained on binary data (i.e. 84 classifiers trained, one per affordance). Table 5.2 shows the average performance according to the top-level *generic* affordance categories.

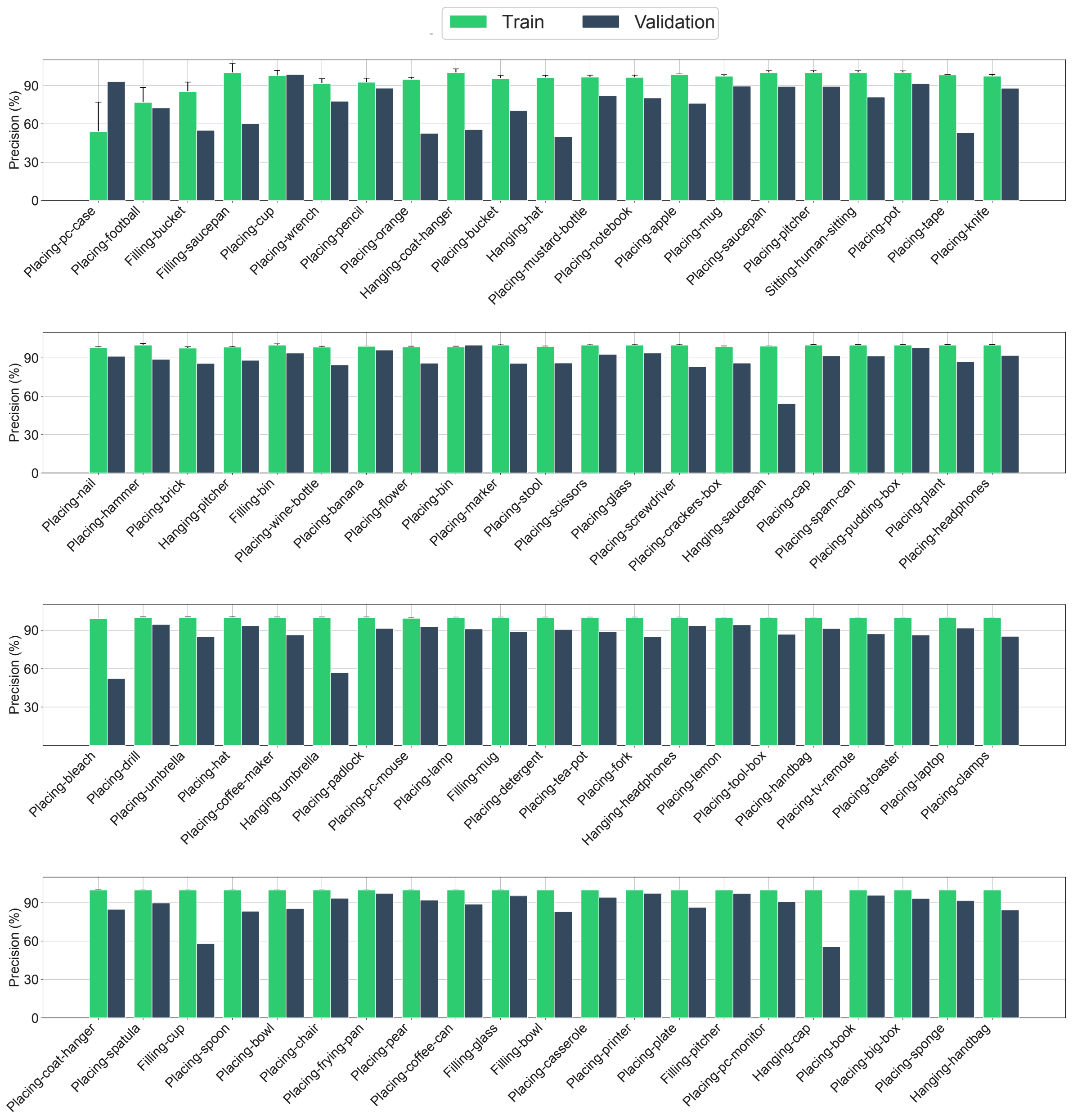


Figure 5.3: Performance of 84 binary m-Pointnet++ classifiers for training and validation data.

With a few exceptions, the network performs favourably training for affordance data for

	Train			Validation		
	Precision	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
Filling	98.16	98.22	0.9819	78.91	73.62	0.7553
Hanging	99.22	99.02	0.9912	67.32	68.37	0.6702
Placing	98.12	98.60	0.9834	87.33	58.72	0.6771
Sitting	100.00	100.00	1.0000	80.95	80.31	0.8063
Average	98.87	98.96	0.9891	78.63	70.26	0.7272

Table 5.2: Average performance of 84 m-PointNet++ networks trained in a binary classification setup. One model per affordance.

binary classification. Bear in mind that these results come from training the network separately for every affordance, the idea behind this batch of experiments was to test that the data is suitable for predicting affordances in a similar way to that of the first **iT** method; that is, answering whether a particular location (i.e. pointcloud) affords a target interaction.

The second set of experiments is carried out in order to investigate the performance of the network when multiple affordances are present in the pointcloud. The dataset for these experiments is created by merging the binary data into a single dataset with $k = 85$ classes (i.e. 84 affordances and background). Again, as in the standard shape classification problem one example of dataset has the label of the affordance that it belongs to, i.e. a multiple class single-label classification setup.

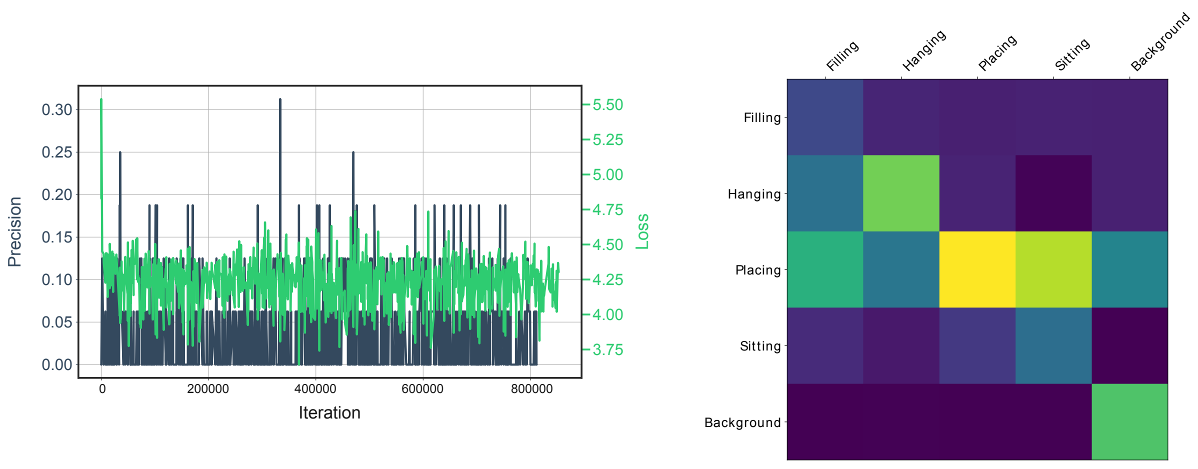


Figure 5.4: Performance observed in the network for training 84 affordances. No convergence is observed during training. The confusion matrix on the right gives some insight for possible explanations.

Figure 5.4 shows observed behaviour in the network while training for multiple class single label data, where it is shown that the network performs rather poorly. Looking at the confusion matrix on the right of Figure 5.4, which shows the top-level affordances categories, gives insight into the issue causing this behaviour. The reason for the low performance (no convergence) of the learning task is that the similarities shared by pointclouds from different affordances cause the network to get confused. Upon closer examination, it was observed that very often the same pointcloud is presented to the network with a different label, for instance, a pointcloud extracted near the edge of a dining table. This type of pointcloud affords *Placing* objects (a bottle for instance) on top of it, but also affords *Hanging* some other objects (for example an umbrella) and even *Sitting* for a human. Another example would be a pointcloud from the tap in a kitchen; this pointcloud affords *Filling* of several objects while also affording *Hanging* objects such as a handbag or even a pitcher. An example of this "confusing" pointcloud is shown in Figure 5.5.

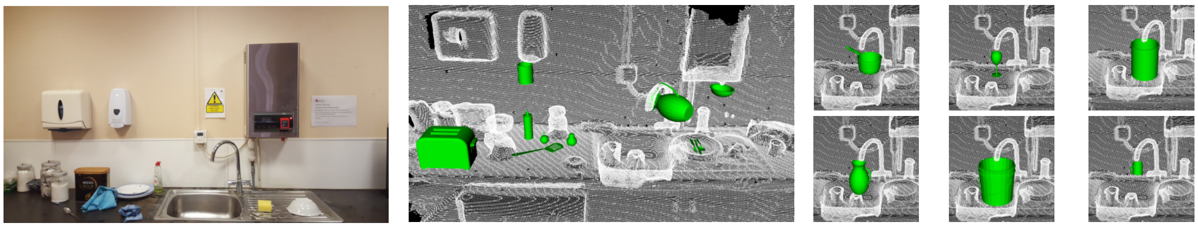


Figure 5.5: Multiple interactions afforded by the same pointcloud (tap/faucet)

These underwhelming results motivated for a change in the approach taken towards the task of multiple affordance learning, given that the affordances that concern this work precisely fall into such extended and natural conditions, where a single pointcloud can afford multiple interactions.

5.4.3 Multiple-affordance learning

In order to account for the multiple interactions that any given pointcloud can afford, the learning task is framed as multi-label, multi-class classification. In this respect, the training label of a pointcloud should encode all the affordance *classes* of the current example. In order to obtain such training pointclouds and labels, the data created for single affordance predictions is post-processed; since now the network needs to be presented with prediction examples shared by multiple affordances. The post-processing step is carried out by searching for common test-points regarded as good candidate locations. Whereas this process could be regarded as trivial, since all the data has been already gathered, it is important to note that the **iT** agglomeration method does not need such "multiple-affordance" training examples. Clustering affordance keypoints from individual interactions (from a single *training* example) allows to predict multiple affordances in any given pointcloud.

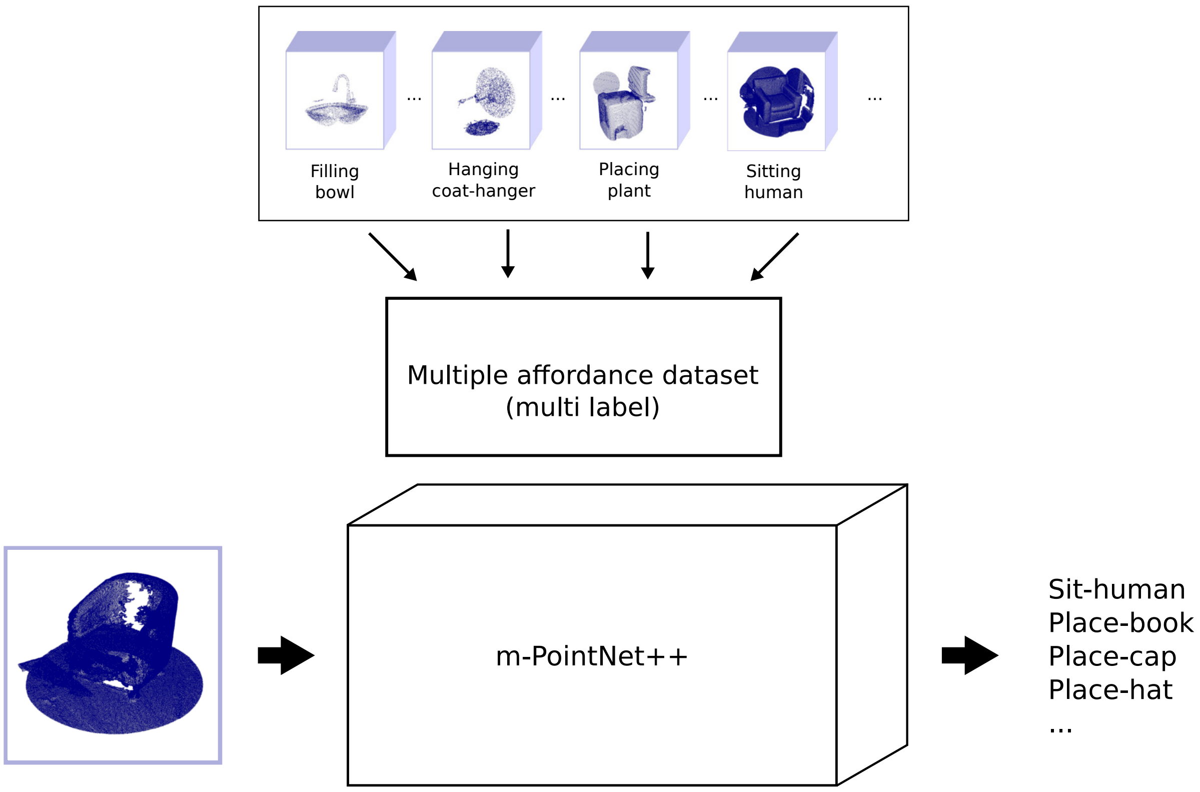


Figure 5.6: The modified version of PointNet++ is trained with multiple affordance labels per training example. This dataset is built using the single affordance predictions of the iT method.

Another important consideration is that the network architecture needs to be adapted for this classification task. For that purpose, the softmax layer at the output of PointNet++ is replaced with a sigmoid layer, and the training carried out with cross-entropy

$$L = -\sum_{i=1}^{k} y_i log(\hat{p}_i) + (1-y_i) log(1-\hat{p}_i)$$

with k = 85 (84 affordances and background). Additionally, L_2-norm regularisation is implemented since over-fitting was observed during preliminary experiments. This modified network architecture (sampling regions, sigmoid layer and regularisation) is referred to as "m-PointNet++" in the remainder of this work. Figure 5.6 illustrates the training process for the modified PointNet++ architecture in a multiple class multiple label setup.

After processing the data from individual affordance predictions, the dataset for m-PointNet++ is comprised by 918K pointclouds (10K per-affordance on average) with an 80/20 split for training and validation. Data augmentation is performed on-line by rotating the pointclouds around the vertical axis by a random angle between $[0, 2\pi)$, adding jitter and randomising the points sampled at the input. The idea behind such augmentation techniques is to increase the number of examples presented to the network but also to account for noise in the input pointcloud (i.e. jitter and randomisation). The performance per affordance of the network in these multi-class multi-label classification experiments is shown in Figure 5.7. Table 5.3 shows the performance observed for the 4 top-level interactions.

	Train			Validation		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
Filling	90.36	73.99	0.8116	78.31	60.63	0.6795
Hanging	81.29	75.80	0.7843	80.69	77.82	0.7893
Placing	88.57	45.58	0.6003	80.78	8.39	0.1386
Sitting	81.97	37.19	0.5117	1.14	0.84	0.0097
Average	85.55	58.14	0.6770	60.23	36.92	0.4043

Table 5.3: Performance of the m-PointNet++ architecture on the multiple-affordance dataset

5.5 Learning affordance descriptors from scene saliency

In this section is shown how the deep network architecture can be employed to learn an optimised multiple-affordance representation, i.e. a descriptor. This is achieved by investigating what the network is learning from the training data as a function of the input pointcloud, i.e. 3D point sets. As explained before, the PointNet++ architecture learns an abstract representation (e.g. feature vector) of the data by grouping local features at different hierarchies. In order to extract such information in an interpretable form, one can track the points at the input that derive the feature vector for an affordance class. These locations are actually used by the network to compute features at the lowest level of the hierarchy; these regions eventually account towards the global pointcloud feature. In other words, there are specific locations at the input that are used by the network in order to make predictions. These salient regions can be understood as scene saliency given that they come from pointclouds from scene data use to make predictions.

Scene saliency is thus obtained by keeping track of the points in the input that activate

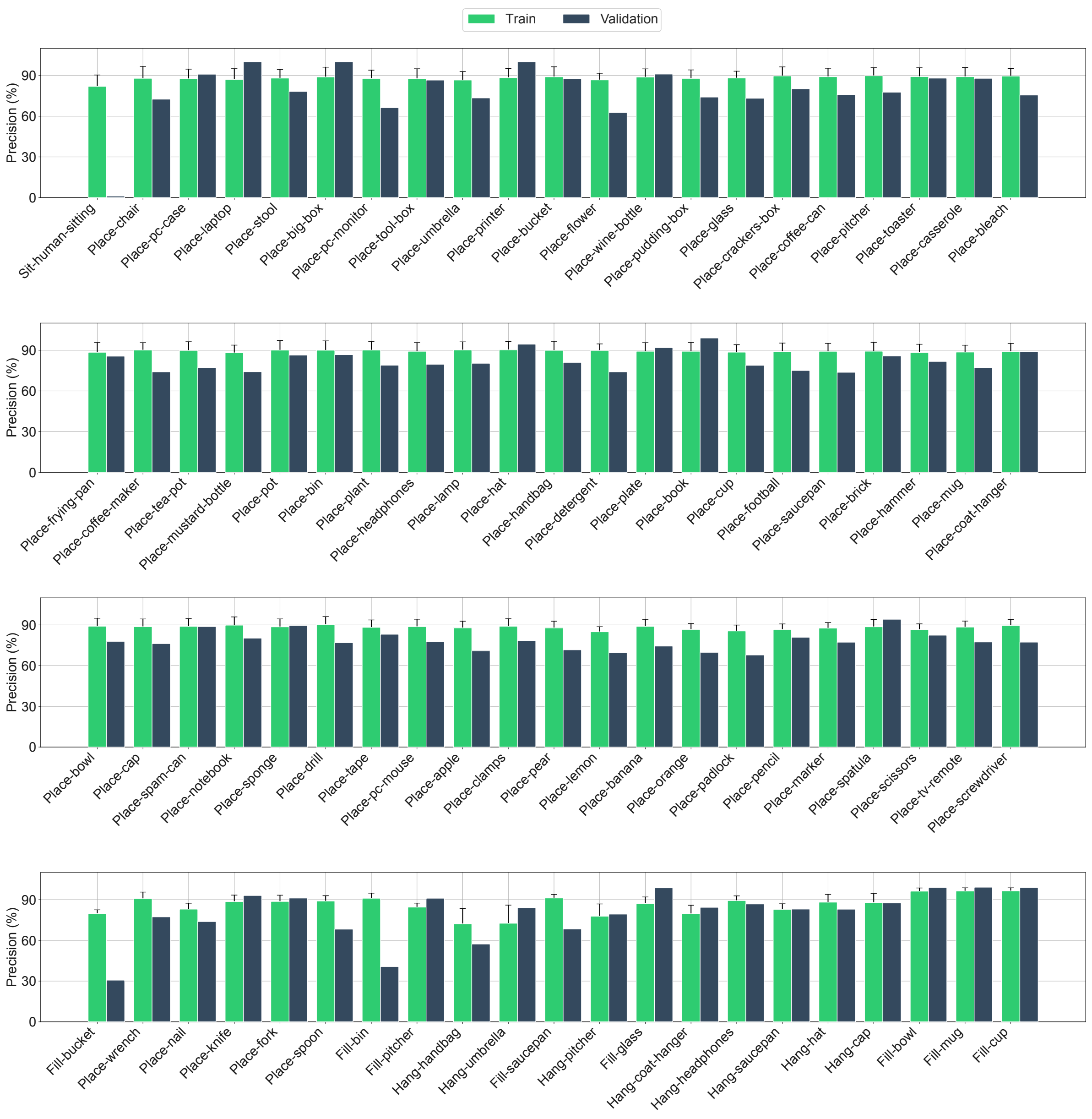


Figure 5.7: Performance achieved in validation set by the m-PointNet++ architecture in terms of precision for 84 affordances.

the network's neurons. This is analogous to the concept of critical pointsets presented in [226], which are the 3D points that contribute to the max-pooled features in the network's first level filters. Due to the fact that the **iT** relates points in the two interacting objects (i.e. scene and query-object), it is possible to *easily* project (scene) salient points back into their associated **iT** location. Briefly speaking, this projection is achieved by computing for all scene-salient points their nearest neighbour on the **iT** of the interactions afforded by the current pointcloud. This is the inverse procedure of that involved in the computation of an interaction tensor (Figure 3.7-right). As a reminder, an **iT** is formed by points on the IBS and provenance vectors that go from a point in the IBS to a point in the scene. Therefore, the process of projecting back scene saliency can also be seen as estimating the inverse of a provenance vector for an **iT**.

Given that the interaction tensors are very dense entities, a grid representation (i.e. cell grid) is employed to alleviate the back-projection process. Thus, the back projection is carried out by computing the nearest cell centroid for every salient point learned by the network. This process is performed for every salient point in every pointcloud in the dataset. Once all salient locations have been projected into their associated **iT**, a new multiple-affordance representation is generated by considering as affordance keypoints the locations in the cell centroids that *received* projections the most; that is that received least 50% of the projections. Figure 5.8 illustrates the back-projection of scene saliency into an **iT** agglomeration.

Notice that since all the pointclouds in the dataset have been “zero-meaned”, the saliency projection process is straightforward to perform, as the interaction tensors themselves are expressed with respect to the reference point in the scene used to generate them. In other words, both the saliency points and interaction tensors use the same coordinate system.

Similarly to the **iT** agglomeration method introduced in Chapter 4, using 3D scene saliency allows to reduce the dimensionality of the representation greatly. This translates into the ability to provide candidate locations for multiple affordances (i.e. affordance prediction) at fast rates. Figure 5.9 shows examples of the descriptors achieved with different sampling methods. The descriptors for all the remaining interactions can be found in Appendix A.3.

One of the first things noticeable in Figure 5.9 is that, as speculated, different interactions need a different number of keypoints; which differs from the sample size used at the beginning of this research ($N = 512$). It is also interesting to see that the sample related to scene saliency seems to be halfway between uniform sampling and weight-driven sampling. This can be seen in the descriptors for *Placing* and *Filling*, where affordance keypoints seem to take into account what was regarded as high-weight regions but also seems to account for regions further away in the interaction (lower-weight). Notice as well that *Hanging* seems to agree more with the uniform-sampling method, whose affordance keypoints more spread around all the way down. In contrast, the descriptor for *Sitting* affordance seems to agree more with the one obtained using weight-driven sampling, where the keypoints are concentrated the central part of the

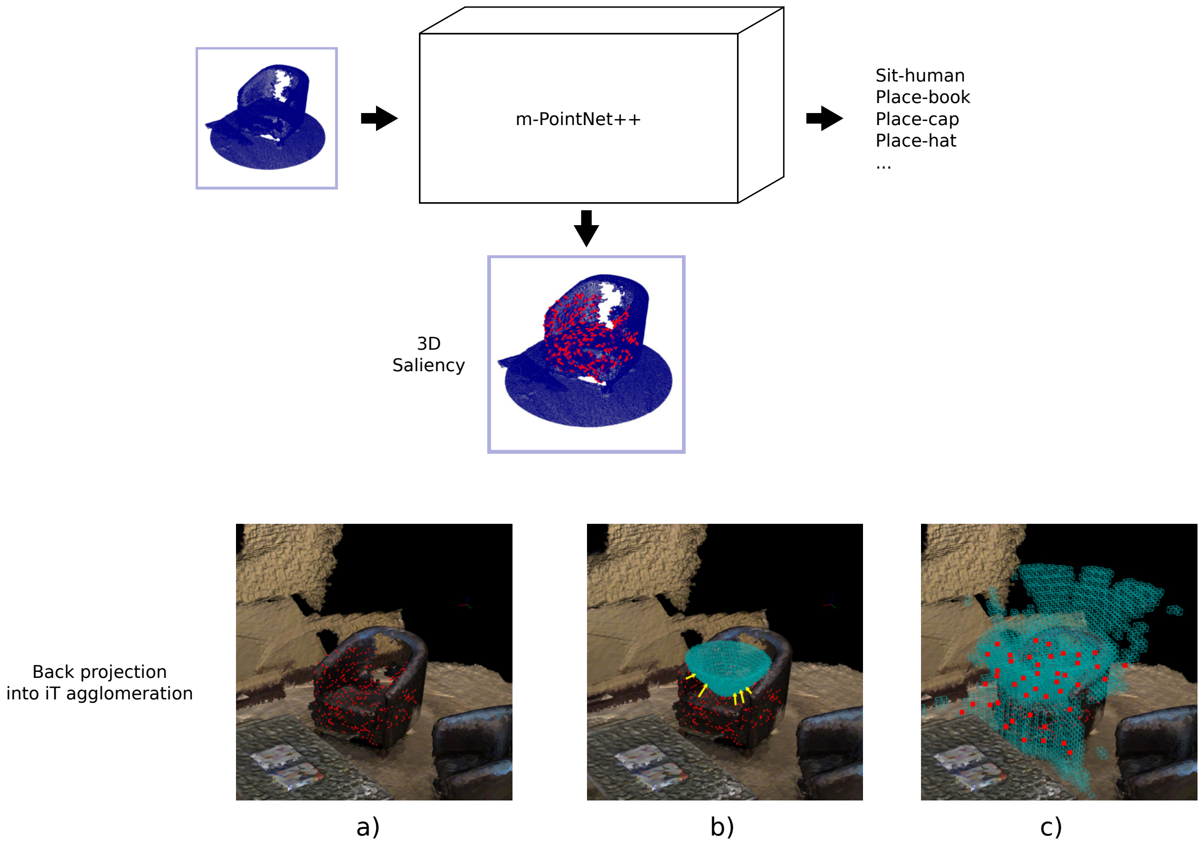


Figure 5.8: Diagram on the top illustrates an example of 3D salient points used by the network to *correctly* classify the pointcloud. Images in the bottom row show the back-projection of saliency into **iT** agglomeration. a) Scene saliency (red points) for an example pointcloud. b) The nearest neighbour in the agglomeration (pointed by yellow arrows) is computed for every salient point (red points) in the scene c) This projection is carried out for every pointcloud in the dataset, the new affordance keypoints (red points) are comprised by the cell centroids that received projections the most.

descriptor, i.e. the parts closer to the human legs. Lastly, note the large reduction in the number of keypoints when compared with previous descriptors. As an example, the descriptor for *Filling* a mug is reduced from 512 keypoints down to only 47; this represents a reduction of nearly 90%. Even the affordance descriptor for the largest object, i.e. the human model for sitting, can be reduced roughly 8 times. Perhaps more striking in the reduction of the dimensionality when compared with the original **iT** dense representations, for instance, *Placing* a laptop has a tensor of dimension 203K, whereas the descriptor devised via saliency only needs 118 keypoints.

It should be noted that a similar approach can be followed to optimise individual affordance descriptors, that is following the experiments of subsection 5.4.2. In that case, the scene saliency extracted from the network corresponds to single affordances and can be projected back into

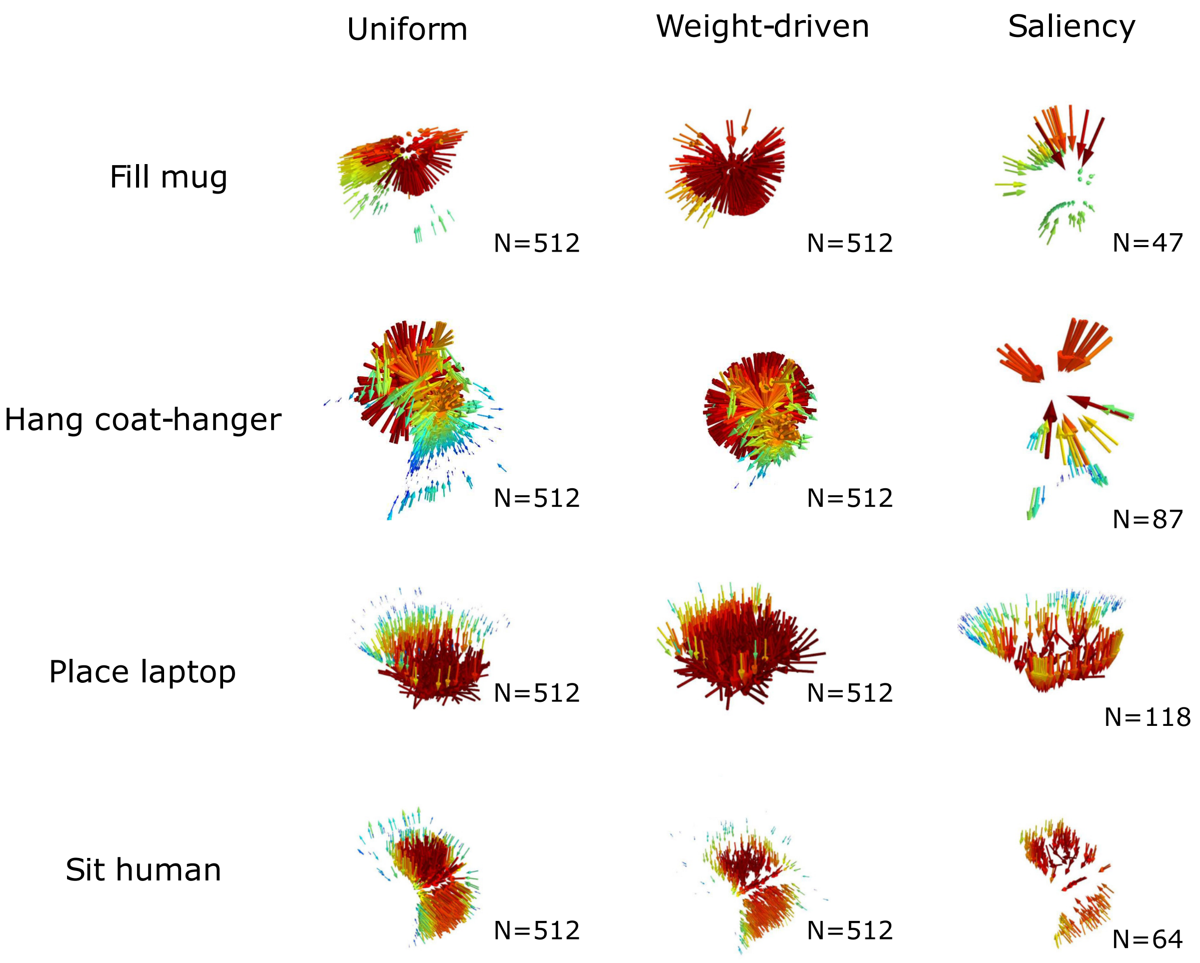


Figure 5.9: Examples of **iT** descriptors based different sampling methods, from left to right: uniform sampling, weight-driven sampling and saliency-based sampling. For uniform and weight-driven sampling the number of keypoints is the same as introduced in Chapter 3, i.e. $N = 512$. The reduction of keypoints in these examples goes from 90% (*Filling*) to 77% (*Placing*).

the **iT** of such interaction. Results of affordance prediction when considering this alternative approach are shown in the following subsections and is denoted as "Individual". This alternative representation is achieved by first extracting saliency and then projecting into the **iT**s on an individual basis. That new set of keypoints, extracted from saliency of individual affordances, is then agglomerated and clustered to form a multiple-affordance descriptor. In other words, following the multiple-affordance approach presented in Chapter 4.

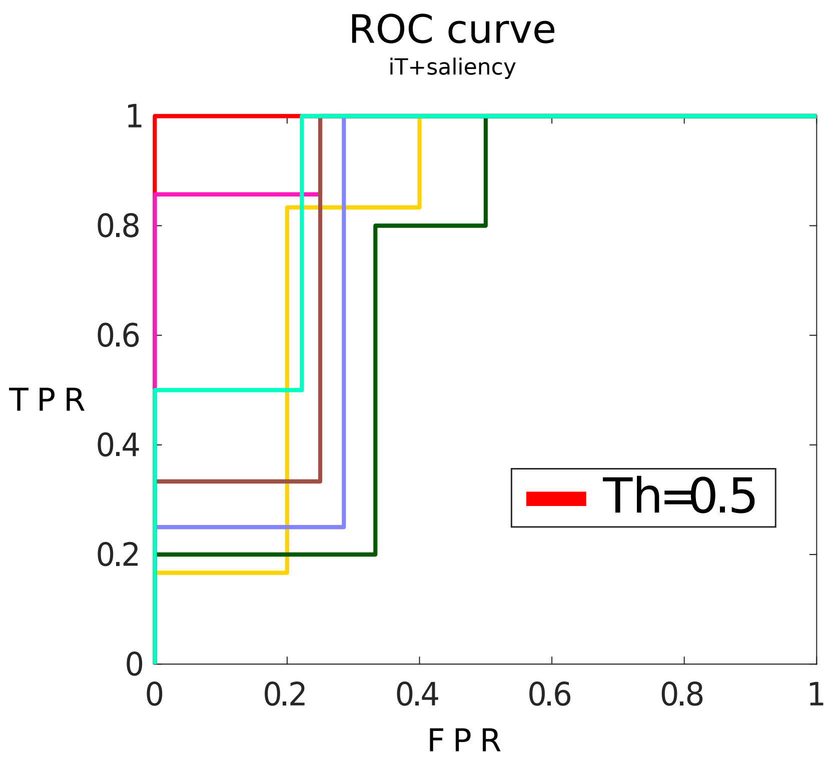
5.6 Evaluation

In a similar fashion to the first multiple-affordance approach (i.e. **iT** agglomeration), the aim here is in giving an answer to the perceptual question of "What can I afford to do here?". Therefore, the algorithm to answer such question is the same as that used to perform affordance queries shown earlier (Algorithm 1). Though, this time the representation used in the pipeline corresponds to the saliency-based descriptor. In this way, the experiments carried out aimed to predict up to 84 affordances or interaction possibilities in any given location of an input scene.

Optimal detections

Affordance location predictions are made by setting a threshold to the output (score) of the querying algorithm. Human validation via Amazon Mechanical Turk is used to determine the threshold that produces the best results with this new multiple-affordance representation. Just as in the previous experiments, people (turkers) were asked to evaluate the predictions made with the descriptor. A total of 2.4K example predictions representing different scores were shown to 69 turkers. The task for these turkers consisted in selecting from two options the image that best depicted the described interactions. Using these pairwise preferences, a Bradley-Terry model [216] was used to compute the "true" ranking of human evaluations. This ranking is used to asses the performance of the new descriptor. Figure 5.10 shows the family of classifiers induced by setting different threshold values at the score of the saliency-based algorithm. Note that many of the lines (i.e. classifiers at different scores) in this plot go through the top-left corner; however they cannot be easily discerned as they are stacked on top of each other.

From this assessment can be observed that predictions with a score $s >= 0.5$ are regarded as good every single time by the turkers, a value that contrasts with the threshold of the **iT** agglomeration method. The method based on agglomeration and clustering of **iT** descriptors needs a threshold of 0.7 in order to produce predictions that agree with human criteria; on the other hand, the saliency-based method performs equally with a lower threshold. This is related to the fact that, as shall be shown in the following subsection, the saliency-based method achieves higher precision rates; meaning that the prediction threshold can be relaxed without compromising the quality of the predictions.



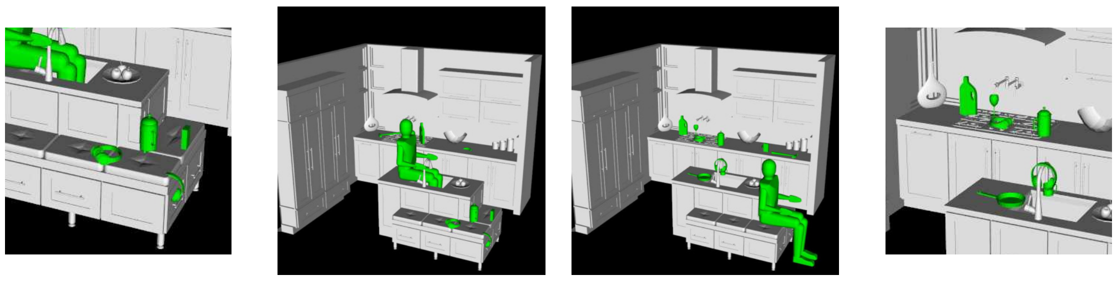


Figure 5.10: Mechanical Turk evaluation. ROC plot on the top shows the family of classifiers generated by setting different thresholds bands to the prediction score. Affordance predictions with a score above a threshold of 0.5 are deemed as good candidates according to humans every single time. Images on the bottom row show an example of the evaluation task presented to humans: select the option that best illustrates the intended interactions.

Individual vs Multiple predictions

Similar to the evaluation made for **iT** agglomeration method, the predictions generated with the saliency-based method are compared against those produced in a single-affordance scenario. As a reminder, these ”baseline“ predictions are obtained by individually querying the 84 affordances individually, i.e. single-affordance descriptor. For fairness, for this round of experiments was used only the data that was not previously considered to generate the affordance dataset for m-PointNet++. All the predictions (made with $s_i >= 0.5$) are treated as "ground-truth" in order to compute performance metrics and make comparisons across all the methods. It is also worth keeping in mind that this comparison is motivated by the good performance observed and validated previously for single affordance predictions (Chapter 4).

Another important aspect to bear in mind is the likely imbalance that scene pointclouds have. For instance, all the vertical surfaces such as walls, cabinet doors, etc, represent negative examples or the background *class* which dominates the scene. One could achieve very high accuracy by predicting no-interaction or background every time. These negative predictions, however, do no represent valuable information since the aim here is to investigate the interactions that are actually possible in a scene. Is for this reason that evaluations are made with precision-recall metrics, as one would do in an information retrieval task. Figure 5.11 shows the performance achieved for every affordance considered in the representation for three cell sizes. Table 5.4 shows the performance of the saliency-based method when for the top-level affordances.

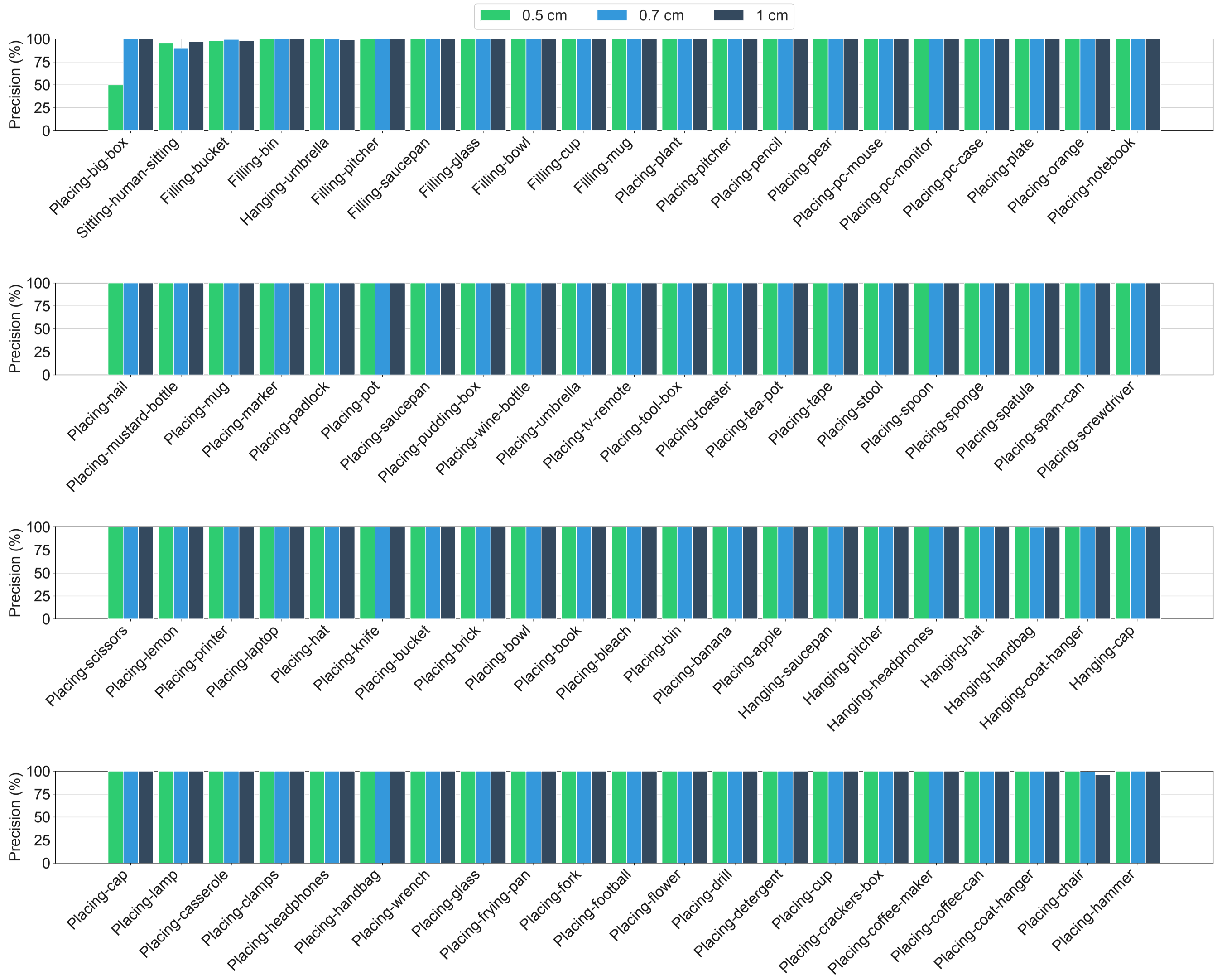


Figure 5.11: Precision achieved with multiple-affordance representations of cell size 0.5cm. Important differences are noted in *Placing* affordances which were largely regarded as the *easiest* interactions. Interestingly both approaches struggle the most with *Placing big-box*, which has several short-length vertical provenance vectors (under the box) difficult to match during testing.

	0.5cm			0.7cm			1cm		
	Precision	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
Filling	99.73	42.71	0.5856	99.78	14.97	0.2553	99.93	14.08	0.2457
Hanging	100.00	10.54	0.1583	99.85	7.38	0.1262	99.98	6.61	0.1145
Placing	99.25	12.81	0.2247	99.95	11.12	0.1983	99.98	8.81	0.1591
Sitting	95.34	23.57	0.3780	96.91	22.98	0.3715	89.71	24.84	0.3890
Average	98.58	22.41	0.3366	99.12	14.11	0.2378	97.40	13.58	0.2271

Table 5.4: Performance metrics for the saliency-based method when compared with single affordance predictions in the top-level affordance categories

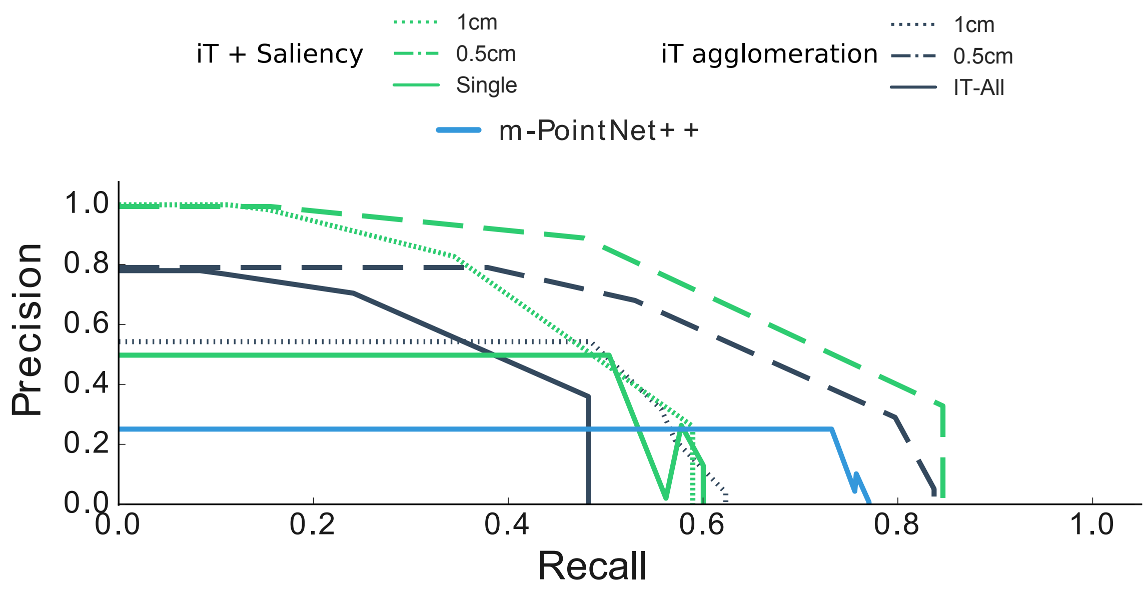
From these results can be observed that the saliency-based method achieves very high precision rates for almost all the affordances. It only struggles with *Placing* big-box for the smallest cell size. Interestingly, in terms of precision, there was no major difference across the different cell sizes used during the saliency back projection step of the method. Having said that, the recall achieved by the method does change across the different cell sizes. This behaviour is similar to the one observed for the iT agglomeration method, where larger cell sizes show a lower recall rate. Generally speaking, all cell sizes manage to make "good" predictions; however, not as many as those generated by single affordance descriptors.

In order to get more insight from this comparison, one more evaluation was made. This consisted in computing performance metrics for data ranked according to the prediction score. For this last round of evaluations, two additional versions of the multiple-affordance algorithms were included:

- **iT-All** This alternative version of the agglomerative approach consists in keeping all the keypoints (and their associated provenance vectors) inside the cell during clustering. The intention of this alternative method was to investigate the possible loss of information (i.e. performance) caused by only considering the closest per-affordance keypoint inside each cell. Thus, the iT-All alternative still uses cell centroids to estimate test-vectors (via NN-search), but all the keypoints (4096 per affordance) are used to compute a score.

- **Single** The alternative version for the saliency-based approach is produced by learning scene saliency on an individual basis (i.e. per affordance); and then agglomerating the associated keypoints to produce a multiple-affordance representation. The saliency extracted in this way corresponds to one learned by training 84 PointNet++ binary classifiers.

In summary, the evaluation compares the results ranked by their predicted score for the iT agglomeration and saliency-based methods of 1 cm and 0.5 cm cells, their variations explained above as well as the performance achieved by m-PointNet++ when tested on its own. Figure 5.12 shows precision-recall curves and the average performance for the methods under investigation.



Method	Cell size	AUC
iT+Saliency	0.5	0.6816
	1	0.4588
	Single	0.2722
iT Agglomeration	0.5	0.5467
	1	0.3043
	IT-All	0.3102
m-PointNet++		0.1879

Figure 5.12: Figure on the top shows Precision-Recall curves for the multiple-affordance detection methods. Table on the bottom shows the average performance of the methods in terms of Area Under the PR Curve (AUC)

Noticeably, the methods with the smaller cell sizes showed an overall better performance. However, perhaps the most interesting phenomenon is the fact that precision drops to zero at specific recall values; which is associated with the quantisation error introduced by fitting grid cells to the **iT** agglomerations. The issue of precision dropping to zero can be explained as follows:

1. Due to quantisation errors, the affordance locations detected by the individual descriptors are also found by the multiple-affordance descriptors but with a lower score, e.g. $s_i = 0.99$ for individual vs $s_i = 0.9$ for multiple-affordance.

2. In order to recover or predict affordance locations at lower scores (as detected with the single affordance descriptors), the prediction threshold of the multiple-affordance representations needs to be lowered.

3. Lowering the prediction threshold would go against the validations of humans, as this would cause false positives to appear, i.e. bad predictions according to human criteria.

In short, none of the methods achieves 100% recall when compared against individual affordance predictions; meaning that none of them is able to recover or predict every possible affordance in every location as single descriptors can. In a way, this is the compromise made so that fast and multiple-affordance predictions can be produced. To put it differently, these methods are optimal if the task at hand is to quickly evaluate the affordance possibilities in a location of interest with a high precision rate. Conversely, if the task is to retrieve all possible "combinations" or every affordance that exists across all the scene, while speed is not crucial, performing single-affordance predictions is perhaps a better approach.

Despite such differences, it is possible to show that the predictions made with the multiple-affordance descriptors are equivalent to those produced individually. In order to do so, the predictions are evaluated through human judgement by asking human evaluators to select from two options the one that best depicted the intended interactions. The options showed to humans consisted of the following:

- Top predictions made by the single affordance algorithm (i.e. single affordance descriptor)
- Top predictions made with the saliency-based algorithm.
- Top predictions made with a naive baseline method

The Naive baseline algorithm uses ICP to compute a score from the best alignment between a target pointcloud (interaction training example) and the pointcloud being tested (input scene). A total of 1200 pair-wise comparisons were shown to 48 turkers, where it was found that 48% of the time people chose the multiple-affordance predictions as the best option when compared against single-affordance predictions. On the other hand, when compared with the baseline method (ICP), the predictions made with the methods (single and saliency-based) were chosen 87% of the time.

The important thing to notice here is that a random guess is 50%, which would mean that the predictions regarded as good as the "ground truth" (i.e. single affordance predictions). According to the results from pair-wise comparisons, most of the time (48% vs 50%) people were not able to distinguish the interactions generated by the single affordance predictions from the saliency-based methods. In contrast, the two of them were preferred most of the time over the baseline, i.e. 87%.

Finally, it is worth mentioning that the **iT** agglomeration method performed favourably even when a single *training* example is considered. What is more, the combination of the interaction tensors with the salient locations learn by the deep network allowed to improve the performance and, as shall be shown next, more quickly asses the interaction opportunities of a given location in the scene. On the other hand and somewhat surprisingly, the network on its own showed a rather poor performance; when compared with single affordance predictions the network was outperformed by all other methods. This highlights the importance of the geometric features that the interaction tensor is able to describe.

Frame rates and quantisation

The effect that the size of cells in the grid has in terms of speed or prediction time for the saliency-based method was also analysed. The experiments were carried out using the same hardware setup from the previous chapter: a desktop PC with an NVIDIA Titan X GPU. Figure 5.13 shows the dimension of the multiple-affordance representations and the prediction rates according to the cell size. Results from the iT agglomeration method are also shown as a reference.

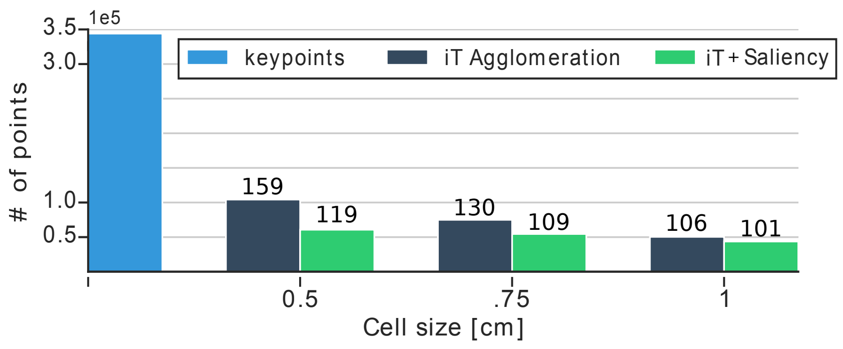


Figure 5.13: Bar plot shows the dimensionality reduction achieved with the proposed methods for different cell sizes. A reduction of up to 6 times in the number of keypoints required to make predictions is observed. Numbers above each bar show prediction time (milliseconds) per test-point of the input scene.

Looking at this figure, it stands out the large reduction achieved by the proposed proposed approaches: a decrease in the number of keypoints in the representation of nearly 6 times (344K vs 60K keypoints). The prediction rates on the same figure show that using grids with a cell size of 1 cm^3 allows to detect up to 84 affordances at 10 different locations in just over 1 second. This is significantly faster (7x improvement) than predicting affordances by trying 84 descriptors at test time, which would require 644 ms per test-point if individual descriptors were tried one after the other (Figure 4.15 in Chapter 4). Due to the fact that the prediction algorithm performs a NN-search in order to estimate test-vectors and compare them against provenance vectors, the complexity of such operation depends heavily on the dimension of the multiple-affordance representation (i.e. the number of centroids/keypoints). Reducing the representation greatly benefits the method, as it allows to perform faster evaluations at test-time. Even with such a reduction in dimensionality, the method is able to produce top quality affordance predictions.

5.6.1 Predictions on RGB-D data

In order to show the quality of the predictions made by the multiple-affordance representation based on scene saliency, affordance queries are carried out for the publicly available dataset introduced in Chapter 4, i.e. ScanNet. Figure 5.14 shows example predictions made with the saliency-based method using cells of 0.5 cm. Just as before, these example images are generated via a post-processing step. The algorithm implemented to produce such images reads the affordance predictions results and produces query-object *placements* while checking for collisions. Recall that the problem of deciding on-line (i.e. at test time) ”what happens where? “, or which affordance should take place at a particular location (out of all the possibilities) is regarded as non-trivial and has not been addressed in this research.

Geometrically plausible predictions

Due to the fact that the approach for affordance detections is purely based on the geometry of the interaction, there are predictions that seem unlikely in the context of real-world physics. These predictions are caused by challenging scenes with noisy or missing data, i.e. bad or incomplete reconstruction. Common causes for unlikely predictions are caused by pointcloud boundaries, concavities and holes, as well as unobserved (unavailable to the sensor) regions of the scene. Keep in mind that the affordance prediction algorithm is not aware of semantic nor physics

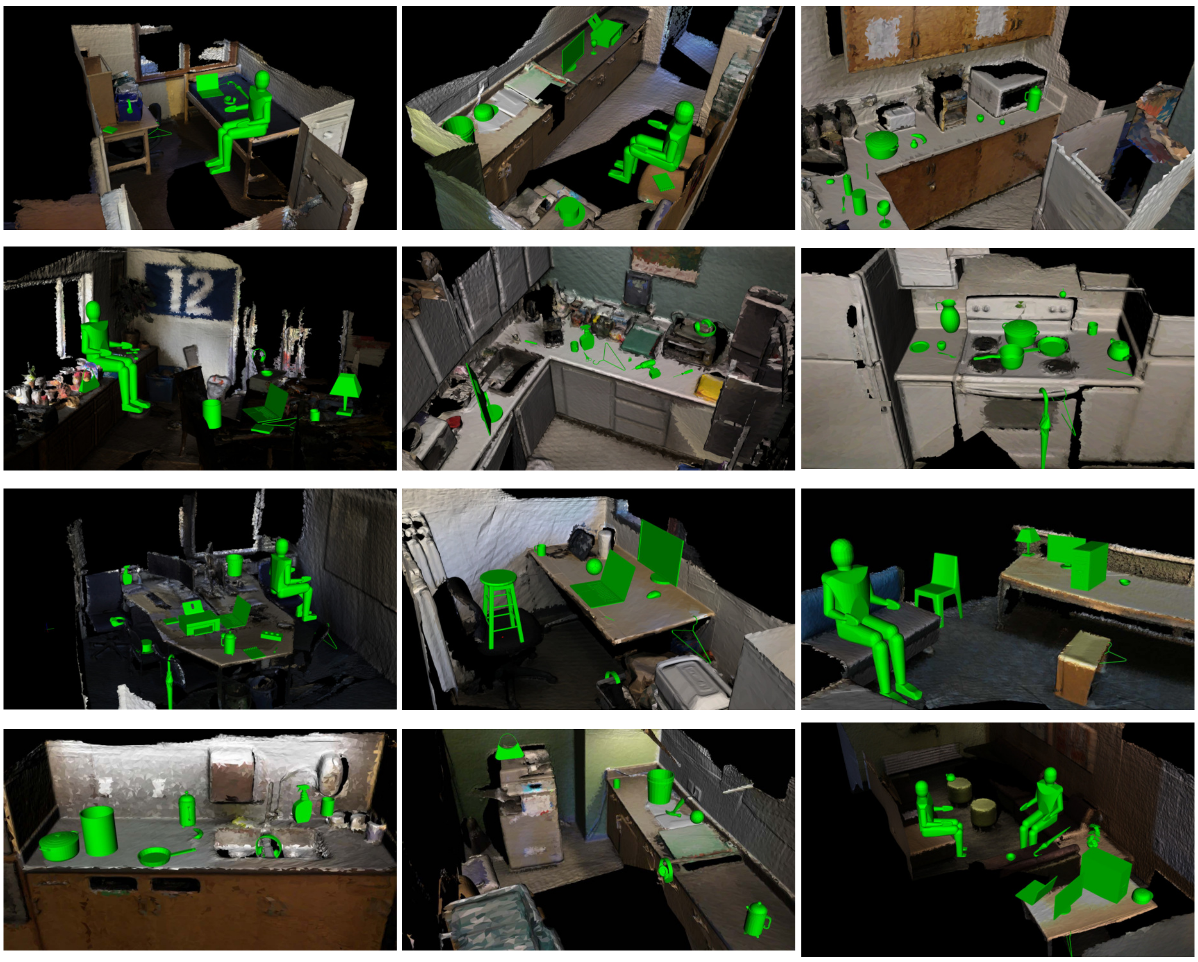


Figure 5.14: Example predictions made with the saliency-based method in RGB-D scans.

information; therefore, assuming that the pointcloud is a good representation of the environment, the algorithm will predict all the interactions that the geometry allows. Examples of these geometrically plausible affordance predictions are highlighted in Figure 5.15

In Figure 5.15, notice that the boundary of the pointcloud appears to afford hanging objects due to the thin and elongated geometry of the walls. Another example shown in this figure is *Placing* objects on flat surfaces originated by scanning the room's ceiling. A third example shown in the figure is *Hanging* an umbrella, which is afforded by the small gap between the wall and the curtains. The hole in the wall affording *Hanging* affordances is caused by the sensor not being able to observe or recover that area of the room. These type of prediction examples could be straightforward to avoid by considering height constraints in the predicted interactions, or by restricting the search area to a region of interest actually available to the sensor.

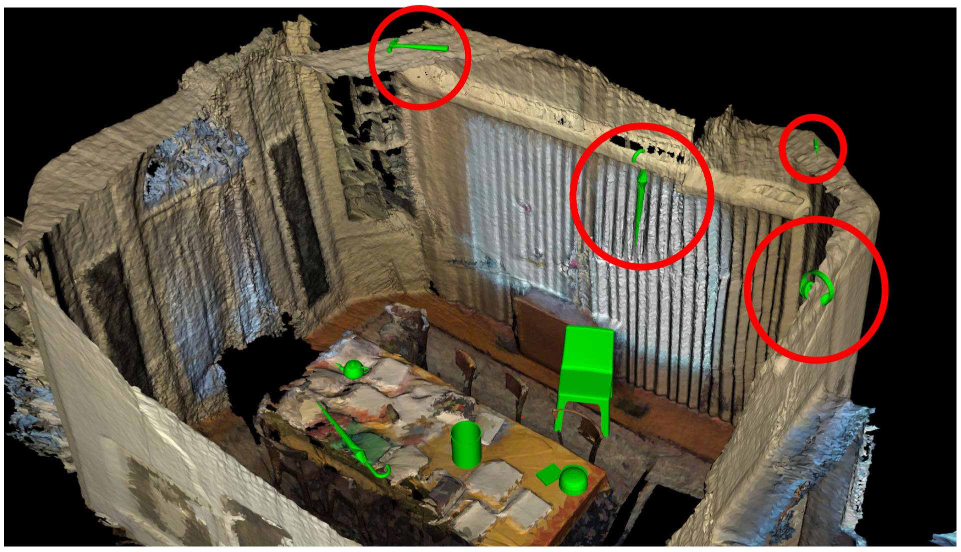


Figure 5.15: Image shows highlighted in red prediction examples that could be regarded as unrealistic yet geometrically valid. The highlighted predictions correspond to (from left to right): *Placing* a hammer, *Hanging* an umbrella, *Placing* a screwdriver and *Hanging* headphones.

5.7 Conclusion

This chapter has presented the hybrid approach proposed to devise an optimal multiple-affordance representation. The proposed algorithm leverages a state-of-the-art deep neural network architecture to learn salient locations in the scene (i.e. scene saliency) associated with an affordance of interest. The combination of saliency knowledge with the **iT**'s description power allows to obtain a compact representation for fast affordance detection on RGB-D scans of real indoor environments. Evaluations and comparisons showed the advantage of the proposed method against alternative approaches; this was additionally corroborated by crowdsourced evaluations, which showed that 87% of times humans preferred the predictions made with the method. The quality of the results and the rate at which affordance predictions enables the application of the algorithm in robotics and computer vision, as shall be shown in the following chapter.

Chapter 6

Affordance detection in real-time systems

6.1 Introduction

Previous chapters have shown that the methods proposed in this thesis allow to predict meaningful affordance candidate locations as validated by human criteria. The methods presented so far have also been evaluated and compared with alternative approaches, in this chapter is presented an outlook of applications for the proposed approach for affordance detection in Robotic and Augmented Reality systems. This is motivated by the proven ability of the approach to quickly evaluate the likelihood of an interaction taking place in previously unseen scenarios. First, in this chapter examples are shown for affordance detection in simulated robotic scenarios. These include detections carried out using the virtual robot's on-board RGB-D camera while navigating different indoor environments. In such examples scenarios, the robot can perceive locations which answer questions such as "Where can I afford to *place*/*hang*/*sit* ?". In a way, this is analogous to the affordance query experiments introduced earlier, with the important difference that this time the input to the system is directly the pointcloud obtained with the sensor.

Later, examples of affordance predictions are shown for an augmented reality system which has been called "Affordance Lantern". This system enables the detection of affordances in previously-unseen, real indoor environments. An important feature of the **iT** method is that enables the scene augmentation to be performed dynamically (i.e. on-line) at the location pointed by an RGB-D camera; for instance, given answer to the question "What can I afford to do here?".

6.2 Affordances for real-time systems

The Interaction Tensor representation with the prediction algorithm proposed in previous chapters has the ability to produce meaningful affordance candidates in novel scenes with high rates of precision and in a fast manner. The research presented in this thesis aims to investigate and develop methods suitable for robotic systems yet predictions examples and experiments have considered RGB-D and synthetic datasets. This was primarily due to the fact that datasets allow for a more controlled or principled way of performing multiple experiments with different methods and parameters. In contrast, the following subsections present two example applications for the affordance detection method in systems that require real-time performance and that need to dynamically detect where and how to interact with a novel scenario. In this example applications, affordance predictions are performed within the environment immediately available to the robot (or camera) and not across the whole scene; thus, results are shown just as they are *discovered* by the system. This is interactive setup is closer to a real-world scenario for affordance detection.

Whereas the robotic applications is limited to a simulated environment, it serves to illustrate that the combination of *conventional* hardware and robotic tools allow to incorporate the affordance detection method into the perception pipeline of a robotic systems. Furthermore, the second application example (affordance lantern) provides an even more realistic demonstration as it uses an out-of-the-box RGB-D sensor, a commodity desktop configuration and standard robotic software tool to perform affordance prediction in real indoor environments.

It is worth mentioning that this chapter serves as a demonstration of the capabilities of the **iT** approach rather than more evaluations of its performance. The evaluations and validations can be found previously in chapters 4 and 5, for now on only illustrative examples are considered for the application of the method in realistic and simulated interactions; in a way this highlights venues for future work and limitations of the current methods. Overall, this chapter aims at showing, via example cases, that the proposed methods have great potential for their utilization in dynamic, noisy and unstructured environments; this has been the motivation of the research presented in this thesis since the beginning, the development of visual perception methods designed for robotic systems that need to interact with their environment.

6.3 The Interaction Tensor for robotic perception

A key objective of Robotics is to devise systems that can operate in previously unknown environments. This has so far been a stumbling block for autonomous and cognitive robots; which, as discussed in Chapter 2, largely fall back onto either well-scripted scenarios or scenarios for which the amount of training needed undermines the notion of little prior knowledge. In the context of affordances, visual perception can be described as a process to understand what can be done where. Which is fundamentally different from asking the two separate questions of "what is this?" and subsequently asking "how can I use it?". Such a unified approach to the perception of the world is immediately useful, as by definition it is one that already takes into account the possible interactions that the robot can perform within its environment without requiring to semantically identify surfaces or shapes in terms of categories and specific features.

As shown in Chapter 4, by taking advantage of modern commodity PC hardware, i.e. GPUs, the **iT** method allows evaluating the likelihood of an affordance taking place at a location in the scene in 10 ms on average. This frame rate motivates for the integration of the proposed algorithm in the perception pipeline of a robotic system, in this particular case a robotic simulation with standard robotic software tools.

Experimenal setup

The simulations were carried out in the Robot Operative System ROS [230] and Gazebo[1] framework, which is a collection of open source software libraries and tools that allow the development of robotic applications. Specifically, a robot navigates a virtual scene and makes short pauses to look around and investigate ”Where can I place a bottle? “, ”Where can I fill a mug? “, ”Where can I sit¿‘, etc. Using the Gazebo simulator, a virtual world is built using CAD models of the scenes introduced earlier in Chapter 4, e.g. kitchens, living-rooms, etc; a total of 15 scenes were tested, 5 of each type (see Figure A.6 in Appendix A). Among the many features of the Gazebo simulator is the fact that allows to simulate the robot sensors realistically, this feature was exploited to demonstrate affordance detections performed in pointclouds generated by a Kinect sensor mounted on a PR2 robot’s head. In this sense, the input to the affordance detection algorithm is no longer a pointcloud generated by sampling a full CAD model but a more realistic pointcloud provided by the sensor (obtained via ray tracing). Other robot sensors in the simulation, e.g. optical

[1]`http://gazebosim.org/`

encoders in the robot's joints, allow to compute and apply the transformation required to obtain the pointcloud w.r.t. the robot's reference frame; thus, allowing to recover the gravity-aligned pointcloud required by the **iT** algorithm.

Once the scenario is prepared, the robot recieves remote commands to navigates the environment, this commands are generated usen the PC keyboard and an open source motion planning framework[2]. The whole system runs in a single PC with a Titan X GPU, a quad-core processor running at 3.4 GHz and 16 GB of RAM.

Qualitative results

Figures 6.1-6.4 illustrate examples of the simulations for the robot exploring different indoor environments. Similar to previous images of detection results, the images shown in this chapters show in green the query-object illustrating the predicted interactions. From the figures it can be observed that *Placing, Sitting and Filling* affordances do show meaningful predictions; notably, this interactions do not require highly detailed pointclouds thus the "raw" pointcloud captured by the sensor suffices. This is important because it highlights and reinforces the characteristics of the method, an affordance perception method that is economic and does not rely complex representations of the world for the agent to be able to dynamically discover interactions immediately available in the environment. An example a real world scenario where the simulated scenario could happen would be a service robot deployed to assist the eldery suggesting the best place to sit in a room. Thus, the examples shown in Figures 6.1-6.4 show simplified yet illustrative scenarios of the possibilities that the **iT** method provides.

[2]Ioan A. Sucan and Sachin Chitta, "MoveIt!", [Online] `Available:http://moveit.ros.org`

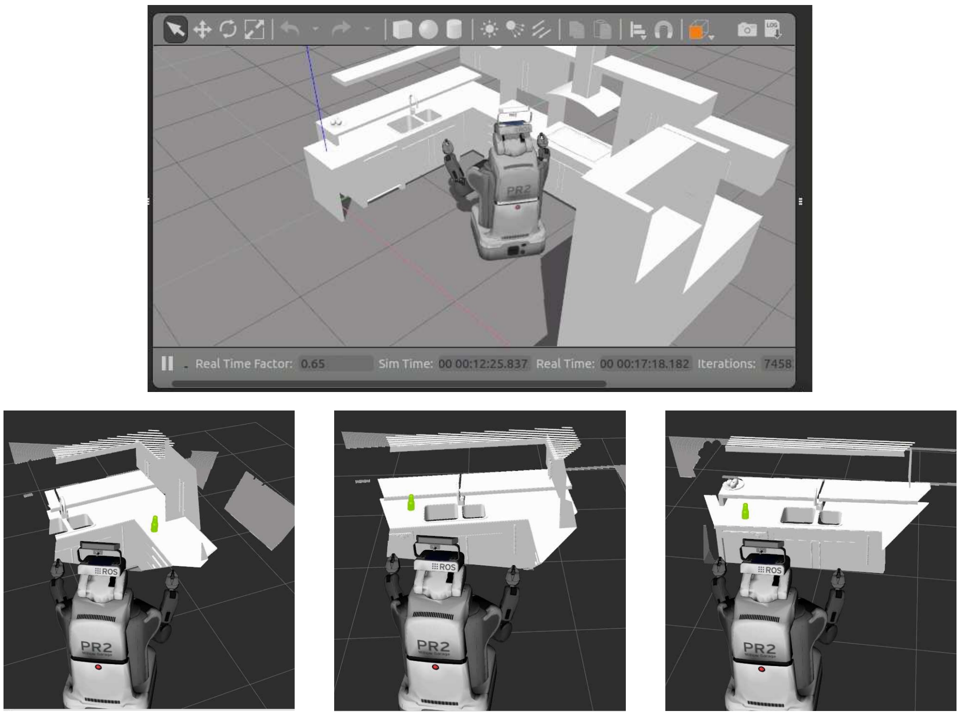


Figure 6.1: *Placing*-bottle detection for kitchen scenes in a simulated environment

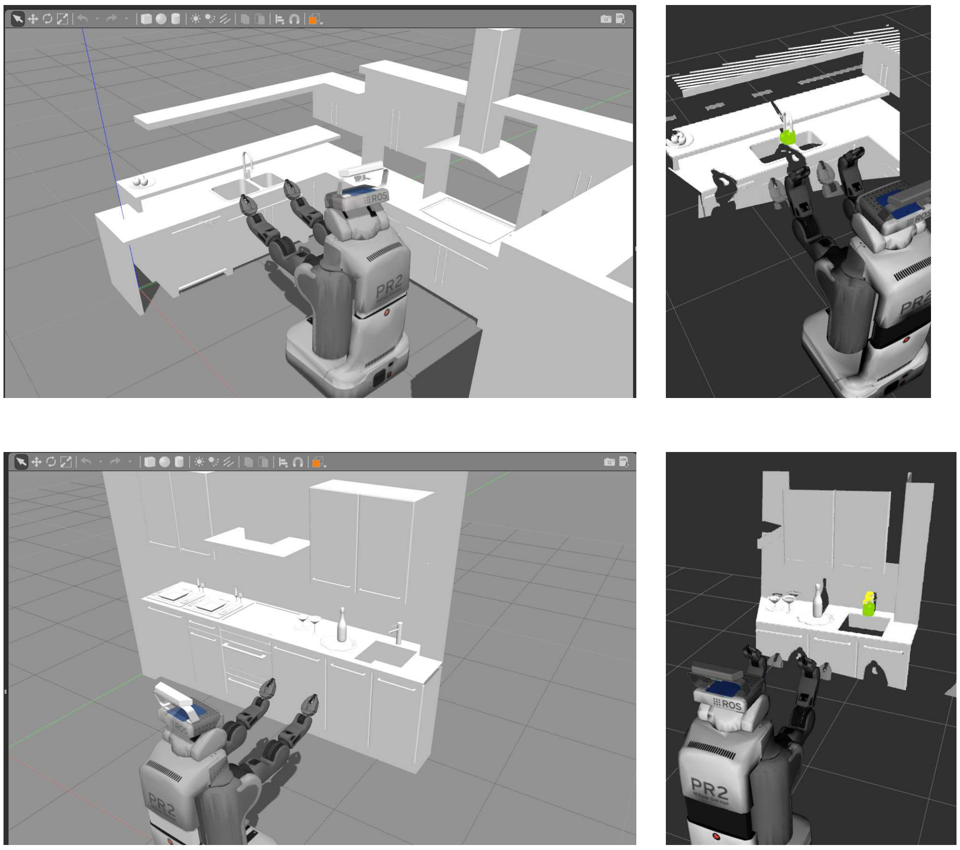


Figure 6.2: *Filling*-mug detection for kitchen scenes.

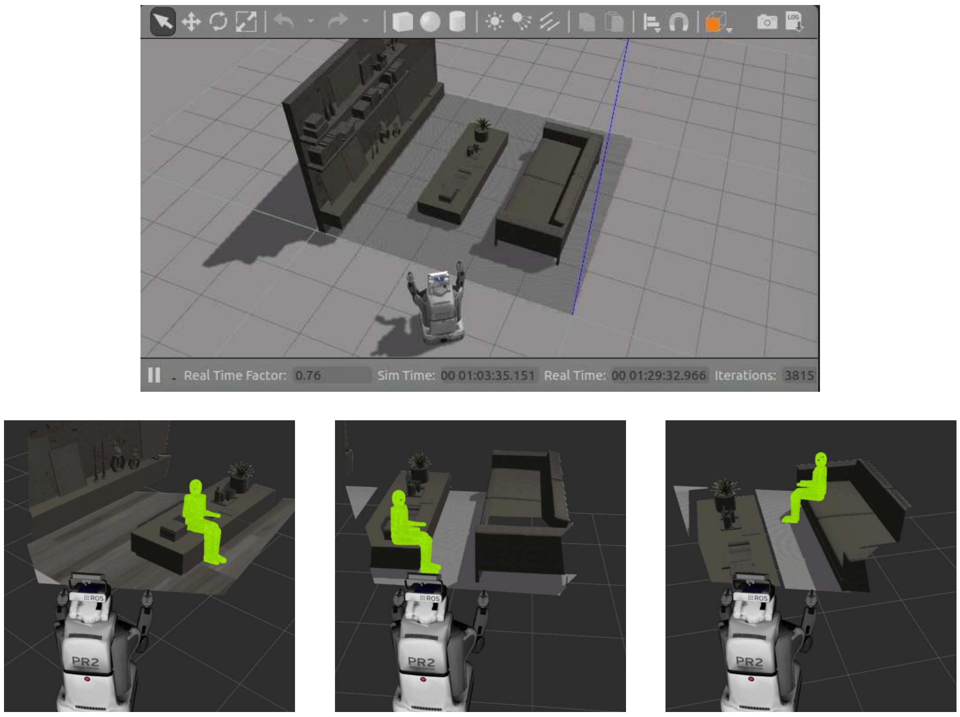


Figure 6.3: *Sitting* detection in a simulated environment of a living room

In contrast, interactions such as *Hanging* affordances were found to require better scene reconstructions or estimations of the underlying geometry in the scene, an explanation for this is as follows. *Hanging* interactions typically take place in regions of the scene with thin geometries (such as edges of tables or bookshelves); or geometries with cylinder-like structure, such as hanging racks or pipes. If the pointcloud coming from the sensor contains such structure, the **iT** algorithm will predict *Hanging* interactions. However, this structure is very commonly found in noisy or incomplete captures from the sensor. As an example of such situations Figure 6.4 shows a pointcloud of a kitchen sink which is only partially detected by the sensor, this makes it look like there is a hole and the prediction algorithm suggests it as a good candidate for hanging a coat-hanger. This interaction would not be possible in reality given that the ”detected“ hole is not there in the real scene. Similarly, pointcloud boundaries seem to afford *Hanging* objects given that they "appear" to be edges of flat surfaces such as tables or bookshelves. The next section introduces an alternative to mitigate bad predictions for this type of interactions, which consists on focusing the search space or *area of interest* for the algorithm in the a region of the input pointcloud that is more likely to have good geometric estimates. Chapter 7 builds more in this topic and lays out more alternatives to solve these challenges.

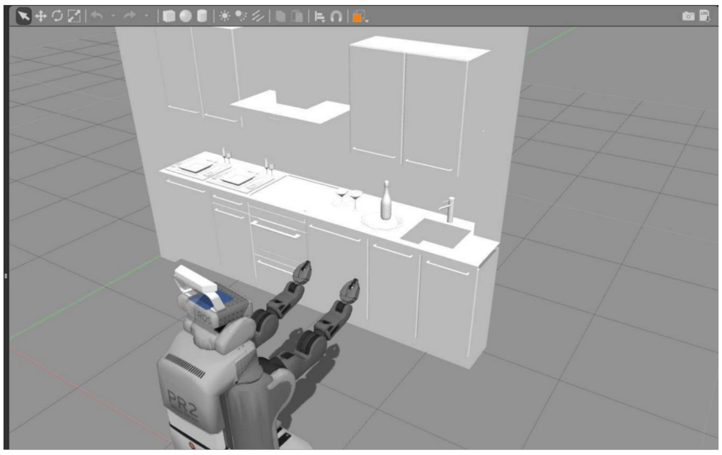
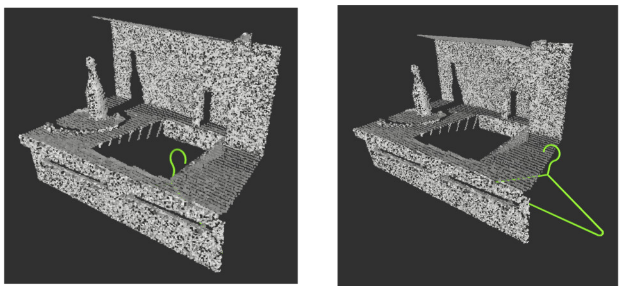



Figure 6.4: Affordance detection for hanging require a richer surface reconstruction from the environment; noisy pointclouds give raise bad or unrealistic predictions.

6.4 Affordance Lantern

One of the advantages of the speed at which affordance predictions are performed is the ability to make such predictions in an *interactive* mode. This, in addition to the validated good quality of the predictions, motivated for the application of the proposed approach in an augmented reality (AR) system. For instance, one could enter into a room, point a sensor (e.g. Kinect) to a location of interest and investigate "What can be done here?". This application where scenes can be augmented using dynamically (i.e. online) discovered affordances is named ”Affordance Lantern“.

Generally speaking, AR systems aim to insert virtual objects or entities into an image sequence in a seamless manner. Azuma in [231] provides what is perhaps the most widely accepted definition of an AR system: a system that combines real and virtual objects, has real-time interaction capabilities and is registered in 3D. For decades, most of the work in AR systems was focused on techniques for Tracking, Calibration and Registration, Display and in the Interaction interfaces [232]. However, as pointed out by Kim et al. [233] the latest trends incorporate new areas of interest for the AR community, among them are Perception, Reconstruction and Modelling as well as Evaluation on the AR systems. The emergence of such areas reflects the importance that the community has found on providing systems that are perceptually accurate, not only in that they provide a *realistic* graphical visualisation or rendering but in the sense of observers being able to perceive realistic interactions with the world. It is here where affordances result appealing as they naturally take into account the interactions opportunities in the environment.

Despite the benefit that affordances can provide, AR systems have made a scarce use of affordance perception as part of them. Among the approaches that leverage the environment's *features* to offer interaction opportunities to the user is Opportunistic Control[234], which provides passive haptics that facilitate the gesture input and recognition in a maintenance inspection task. In [235–237] authors showed that affordances are useful to assess the perceptual fidelity of AR environments for tasks such as walking-through and stepping-over, where the perception of size and distances (i.e. 3D space) is highly relevant. The concept of afforded consequences introduced in [238] uses physics simulations to predict the outcome of perceived affordances (e.g. support, move, roll) in a cube stacking scenario. An AR-based system that allows interacting with out-of-reach objects was presented in [239], in what the authors called UBII: Ubiquitous interface and interaction. Another interesting approach, albeit not explicitly invoking the affordance concept, is the work of Salas-Moreno et al. [240], which presents an AR-capable system that allows for virtual characters to navigate a scene and find places to sit based on the prediction of 3D semantically meaningful geometries (e.g. chairs).

Note that these approaches build on the assumption that the useful set of affordances is already in the environment; then, the detection of such affordances consists in detecting the objects or markers of which affordances have been pre-defined. As discussed along this thesis, one big advantage of the affordance detection approach based on the interaction tensor is the ability to work on completely novel scenarios. Additionally, in contrast with methods based on *traditional* computer vision, affordance predictions do not rely on specific instances or classes of objects and shapes.

Affordance-guided scene augmentation

Just as it has been discussed at various points in this thesis, one requirement for affordance detection with the **iT** descriptor is the ground plane calibration. Another aspect to consider is the quality or density of the pointcloud that is used to query affordances. Due to these reasons, a decision was made to use a state-of-the-art dense mapping system that allows to track the camera pose and provides a dense reconstruction of the scene. As reviewed in [232, 233, 241], Simultaneous Localization And Mapping (SLAM) systems paired with RGB-D cameras are a common approach for camera tracking and scene reconstruction in AR systems. In particular, this application makes use of the ROS implementation of the dense mapping system presented in [242].

The goal of the scene augmentation application is to embed the predicted interactions (query-object with its pose) in the *real world*, i.e. the image stream from the camera pointing to a location of interest. The virtual objects must be precisely aligned with the world in the perspective of the camera/user. Figure 6.5 shows how the real world is augmented with the affordance predictions in the virtual world.

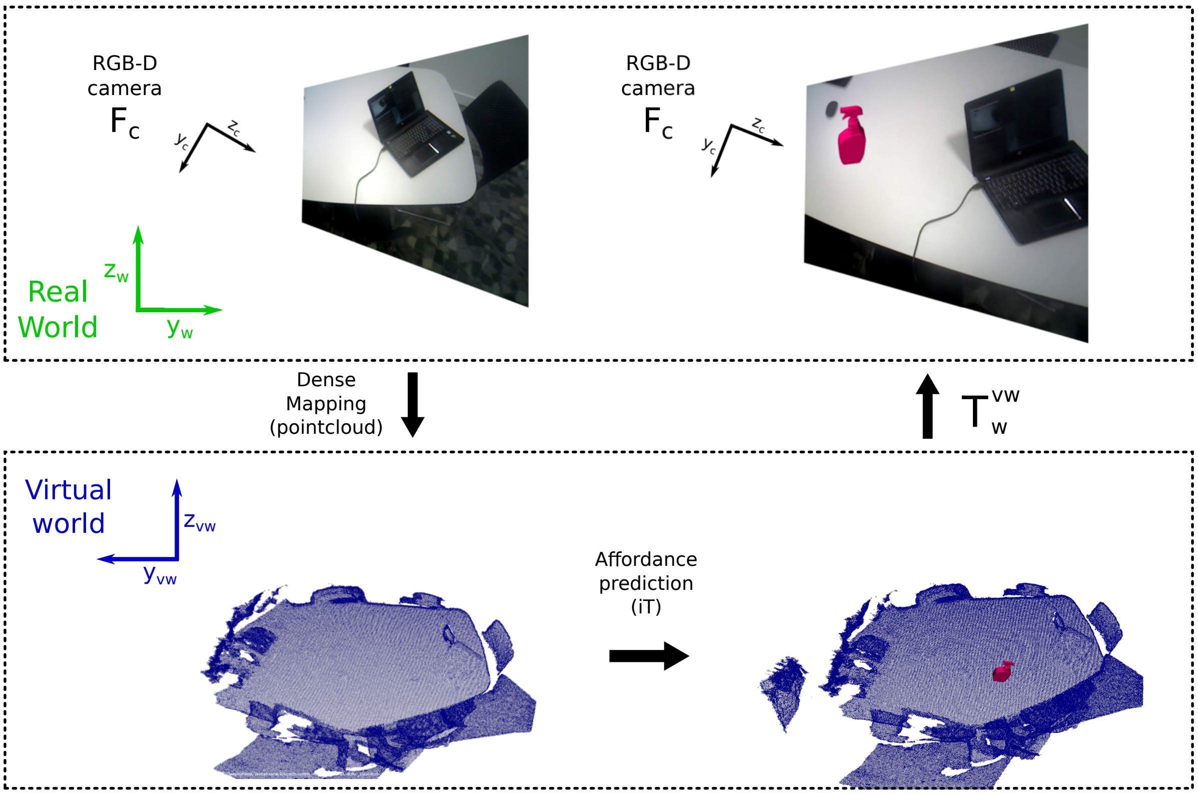


Figure 6.5: Affordance lantern working principle with the coordinate systems involved in scene augmentation. A dense mapping system provides the camera pose used to map the scene point-cloud from real-world w to virtual-world vw, i.e T_{vw}^{w}. The inverse transformation (T_{w}^{vw}) allows to embed the predicted affordance locations (displayed as query-objects) back into the real-world.

Implementation details

The mapping system does not provide an absolute position relative to the world but rather the camera pose relative to a local frame manually established. The lack of pose relative to the global or world frame is not an issue for the **iT** algorithm since the descriptor only cares about the gravity being consistent with real-world, i.e. ground plane in the scene and ground plane in the world have the same normal vector. In order to meet such requirement, an *initialisation* step is included where the RGB-D camera is pointed towards the floor or any other flat surface parallel to it. Using this surface a *virtual* ground plane is computed and used to correct the pitch angle of the camera, i.e. corrected to reflect the real angle w.r.t. to the floor (XY plane). After the calibration step, the pose of the camera is used to compute the transformation that aligns the real-world (w) pointcloud with the virtual world (vw) T^{w}_{vw}. The inverse transformation is applied to map the affordance detections back into the real-world frame, i.e. T^{vw}_{w}.

The scene reconstruction available from the mapping system is provided by an external package that implements meshing through OpenVBD[3] libraries, which slow down the whole system if requested constantly. One one hand, mesh representation can be useful in AR systems to handle occlusions and collisions between real and virtual worlds. On the other hand, the **iT** method has proven to handle undesired collisions with the scene, provided that scene representation is a dense enough pointcloud. In light of these circumstances, it was decided to add some extra functionality to the mapping system so it could provide access to a dense representation of the scene in the form of pointcloud. This dense pointcloud is constructed from the depth maps that the mapping algorithm already builds as part of its pipeline, the extra functionality added to the algorithm consisted of the code to efficiently extract and share the dense pointcloud from the mapping system's internal data structures.

Occlusions are handled by ray-casting with an occupancy grid fitted to the scene pointcloud. Fitting this occupancy grid does not constitute a problem for prediction speeds due to the fact that the grid is based on the Octree structure that the **iT** algorithm already requires for NN-search. This allows to only make predictions on the actually visible parts of the scene. The visibility test consists in verifying that a ray from the camera to the predicted query-object follows a collision-free path, i.e. not intersecting scene neighbouring voxels. This is similar to the approach presented in [243].

[3]http://www.openvdb.org/

The working principle of the affordance lantern is similar to that of an actual lantern (i.e. light source) that needs to be pointed to a target location in order to *light-up* that location. In this way, the **iT** algorithm focuses the search or prediction on a region of interest (ROI) in the centre of the image currently being captured by the camera. The location in the pointcloud that corresponds to the ROI is computed from a neighbourhood (20 cm radius) around the middle point of the image provided by the camera. This ROI is used only to sample test-points since the dimension of the voxel used by the prediction algorithm is dictated by the size of the descriptor (i.e. query-object). The benefits of considering only a ROI for prediction are that 1) avoids expensive computations required to search all over the input scene, 2) reduce the number of collision checks, and perhaps more importantly 3) avoid the previously observed $faulty$ predictions that occur on the pointcloud boundaries. Figure 6.6 shows an overview of the processing pipeline with the computation times involved in the main stages.

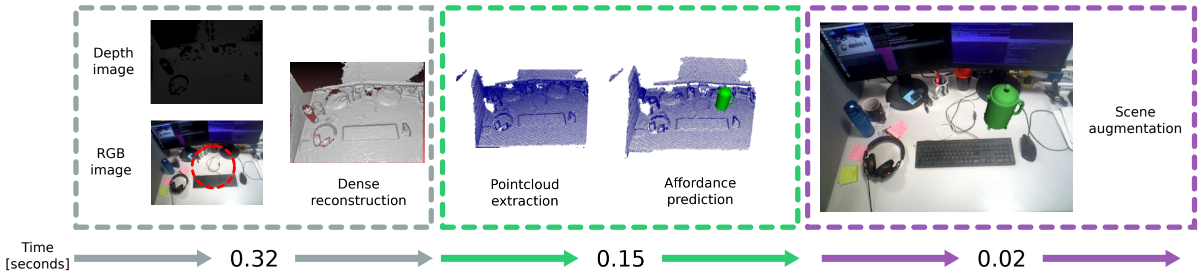


Figure 6.6: Affordance lantern pipeline. The input to the system is the RGB and depth images of the current scene. The red circle in the RGB image on the left illustrates the ROI used for detections

All the modules run within the ROS framework, which allows to *easily* integrate and execute in parallel the multiple computing modules comprising the affordance lantern, in addition to enable the recording/playing/processing of RGB-D video, camera poses, pointclouds, etc. in an *automatically* synchronised manner. The affordance lantern system runs in a desktop PC with a Titan X GPU, where the graphics rendering is implemented through OpenCV and OpenGL libraries. The scenes used for this demonstration are of a similar nature of those presented along this thesis: indoor environments such as offices, kitchens and living-rooms/lounges. More concretelt, the experiments are carried out within the University of Bristol at the 1CS building. In order to be able to capture the necessary data (i.e. RGB-D video), an Asus Xtion sensor and laptop running Linux OS are used to first record video which is later used as input the affordance lantern system in the desktop PC.

Qualitative Results

Example predictions of the Affordance Lantern system are presented in Figures 6.7-6.10. As it has been discussed earlier, the prediction algorithm is able to provide multiple affordance candidates for every test-point in the scene; however, for ease and clarity in the display of results, predictions are shown individually, i.e. one query-object per image. Also, notice that these images represent a video sequence showing multiple viewpoints of the same object along the columns for every row. Examples for *Filling* affordances were purposely tested in a kitchen scenario as they would not be seen in other type of environments (see Figure 6.10). The examples shown below are drawn with a *basic* rendering, keep in mind that the goal of this chapter is to show that the proposed method is able to dynamically discover or predict interactions in realistic and unknown environments. Figure 6.8 and 6.9 show a video sequence with prediction examples for *Placing* and *Hanging* predictions in a office desk environment. Figure 6.7 shows *Sitting*, *Hanging* and *Placing* affordances in a similar environment, an office with a desk and chair. Results are promising although there were important challenging situations mainly attributed to the mapping and camera tracking system, these challenges scenarios are discussed next.

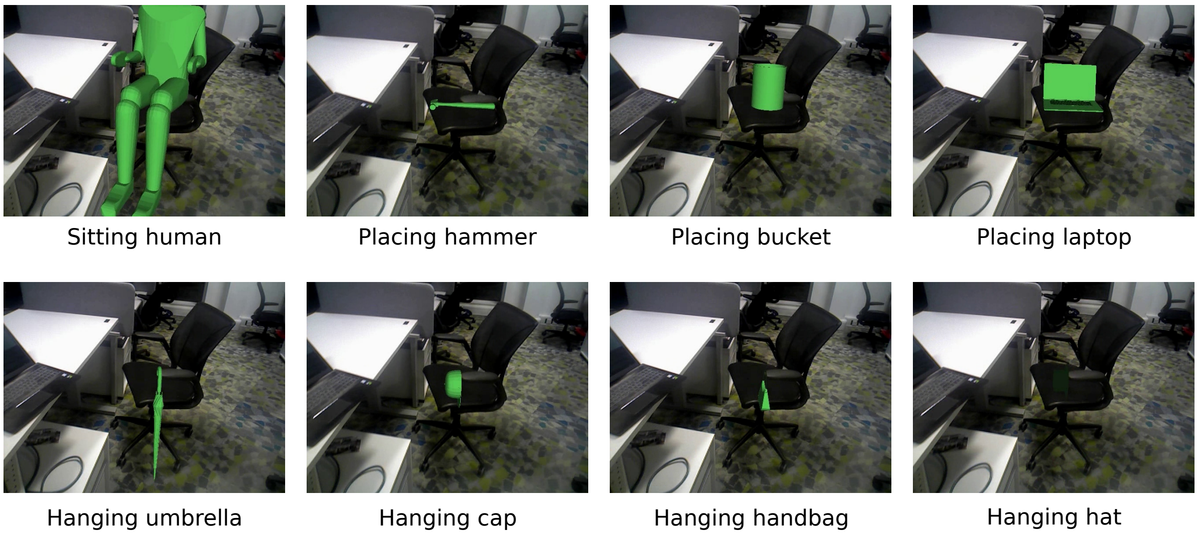


Figure 6.7: Affordance lantern example predictions for *Sitting, Placing* and *Hanging* objects in an office environment.

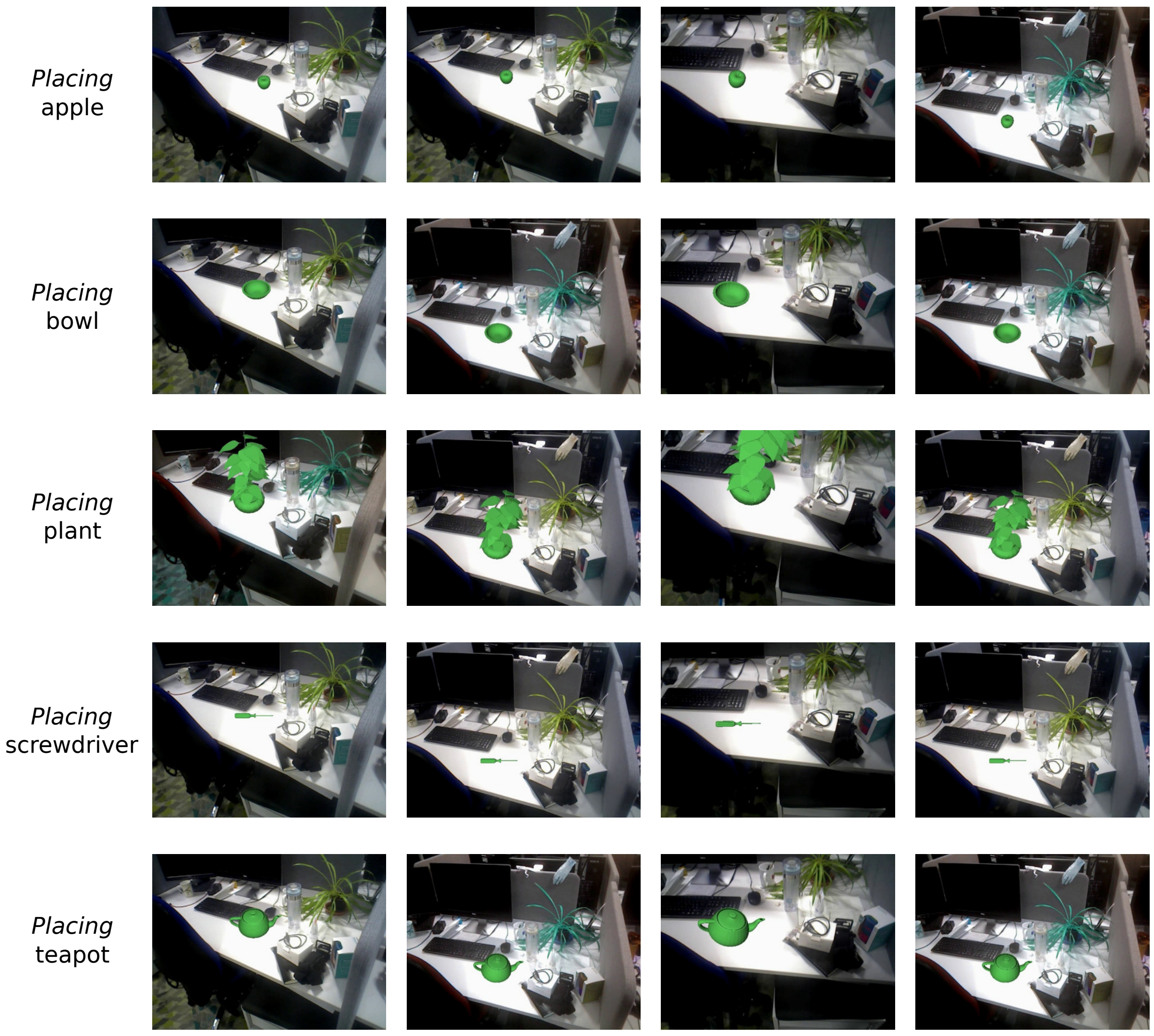


Figure 6.8: Affordance lantern example predictions for *Placing* affordance. Rows show different affordance predictions, i.e. query-objects and columns show multiple viewpoints extracted from the same video sequence.

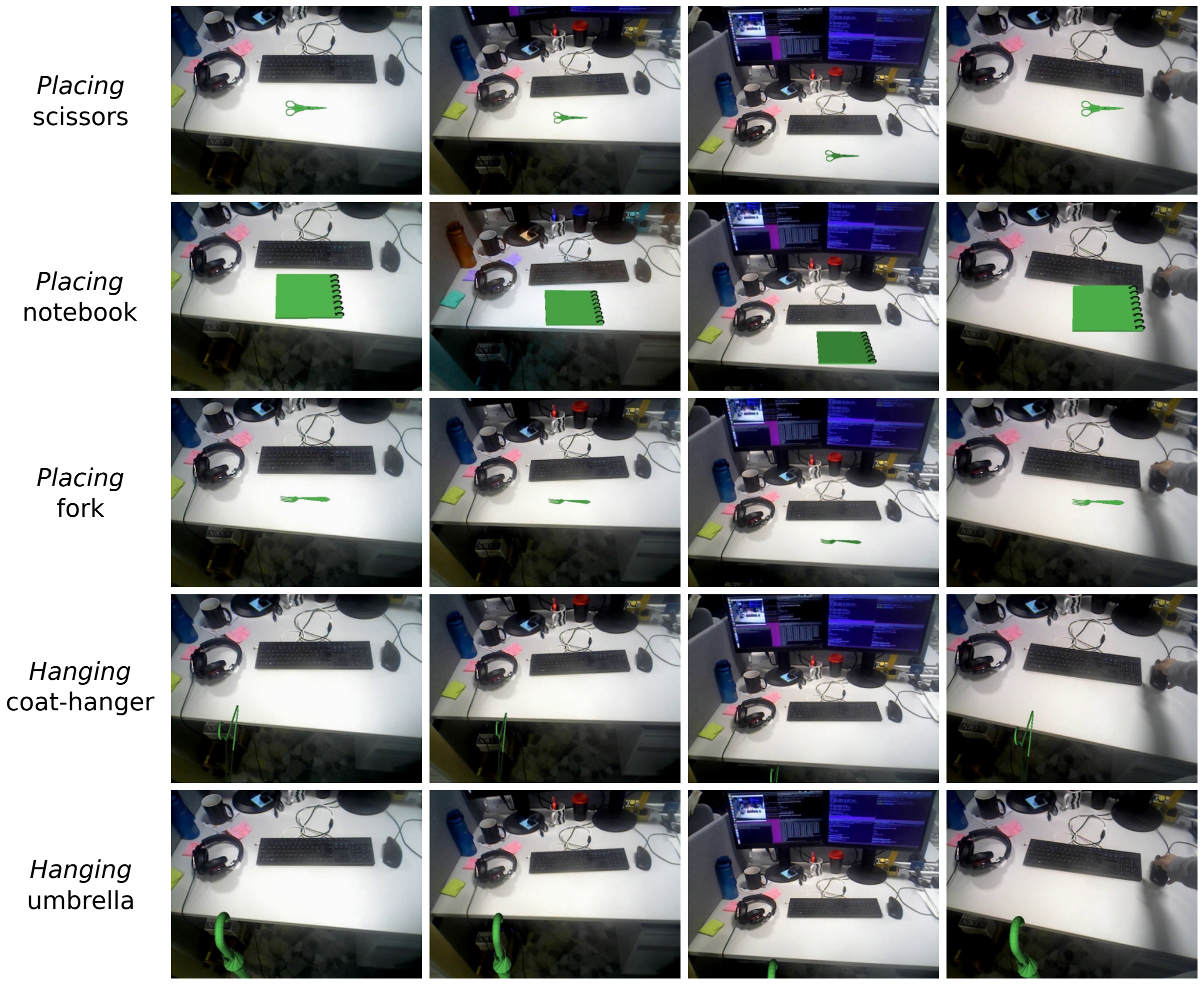


Figure 6.9: Affordance lantern example predictions for *Placing* and *Hanging* affordances. Note the lack of occlusion rendering for the *Hanging* objects; this issue could be avoided by having access to mesh information from the scene.

Challenges

During the scene augmentation experiments in this application, errors in the camera pose caused two main issues. First, because the plane calibration step was only carried out at the beginning, errors accumulated over time in the camera pose estimation provoke that the query-objects are not optimally displayed or embedded into the camera's image stream. This phenomenon can be seen in the third column images of Figure 6.8, which show the query objects with a slight tilt to the right. If the video sequences were longer, the error would be larger causing the query-objects to be misplaced in the camera's viewpoint. The effect of errors in camera pose estimates is also evident in the example predictions for *Filling* affordances in Figure 6.10, where the query-objects show a slight tilt away from the camera.

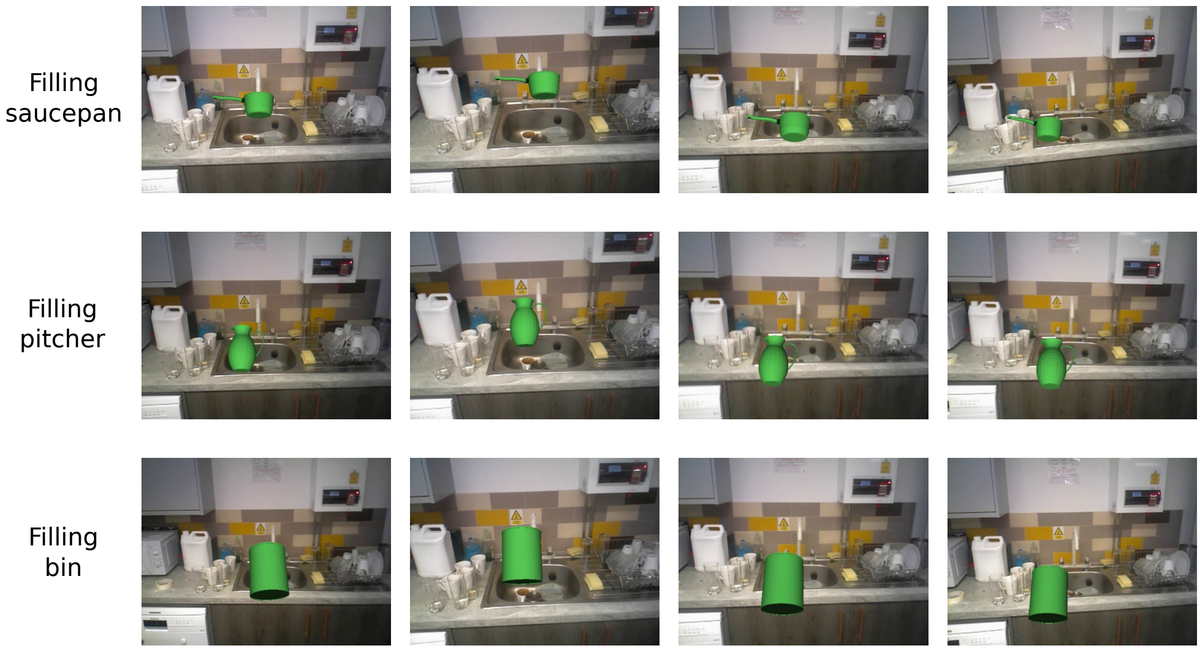


Figure 6.10: Affordance lantern example predictions for *Filling* affordances. Notice that the tilted pose of the query-objects is due to errors in the camera pose estimates, despite the noisy camera pose estimates, the affordances are correctly predicted in the tap.

Second, given that the camera pose estimate accumulates error over time, the pointcloud provided by the mapping system will gradually lose the required gravity-aligned pose. This gives rise to a virtual ground floor that progressively deviates from the required orientation. An example of this situation is shown in Figure 6.11, where can it be seen that the pointclouds' ground plane is tilted towards one side. Although the **iT** algorithm manages to predict many interactions correctly, other affordances are not discovered since they rely on short provenance vectors; these vectors are not sufficiently matched when flat horizontal surfaces in the scene are not horizontal in the pointcloud. A more accurate camera estimation and mapping system would reduce this effect and allow for more and better affordance predictions to be made.

The effect caused by the lack of mesh data to correctly render partial occlusions between scene and query-object is shown in Figure 6.9 for *Hanging* a coat-hanger. In this example, the candidate location is predicted on the edge of the desk; however, in this position, the lower part of the coat-hanger would be under the desk (i.e. not visible). The current system does not take into account such occlusions between parts of the scene and parts of the query-object; thus, the coat hanger appears to be placed on top of the desk image. A similar scenario is illustrated in Figure

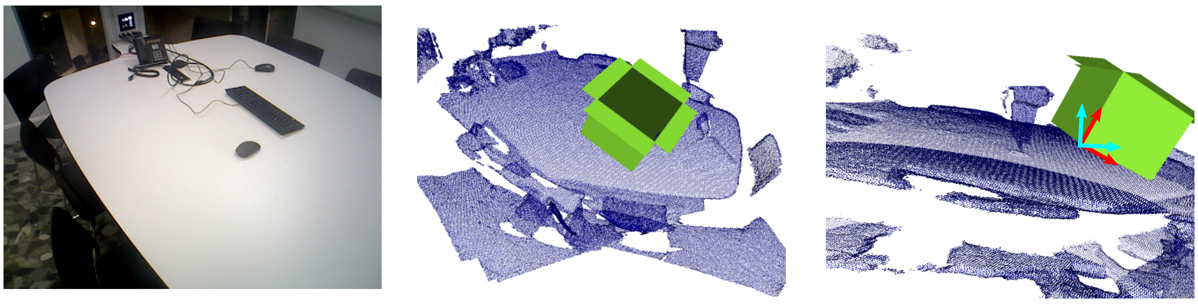


Figure 6.11: Errors in the camera pose estimation accumulated over time give rise to incorrectly orientated pointclouds in the virtual world (i.e. for prediction). The image in the centre illustrates an attempt to *Place* a big cardboard box to highlight the error in the orientation. The image in the right shows the expected virtual world coordinate frame as the X-Z axes in red, and the frame estimated using the camera pose (cyan).

6.7 for *Sitting*, where the human model appears to be sat on the chair's armrest even though the predicted location for this affordance is the actual seat. These issues, though, are mostly related to the realistic rendering of the interactions and not to the quality or precision of the predicted affordance.

6.5 Conclusion

This chapter has presented two applications of the affordance descriptor based on the interaction tensor. Motivated by the fast and high-quality predictions achieved with the proposed method, examples have been presented for the application of affordance detection for robotic and augmented reality systems. These examples serve to highlight the potential of the affordance perception method based on the interactions tensor for systems that require detections at a fast rate. Using open-source software libraries, the implemented systems showcased how a robot can explore a scene searching for candidate locations to *Place, Sit, Fill*, etc. by only using the pointcloud perceived with its sensors. It has also been shown that, with the help of a dense mapping system, an interactive scene augmentation (i.e. augmented reality) application can be devised to investigate the interaction possibilities at a location of interest in real indoor scenarios. The following and final chapter summarises the work presented in this thesis, presents the final discussion and introduces avenues for future work.

Chapter 7

Conclusions

Affordance perception is a fundamental ability for agents that need to interact with their environment, or more generally, understand the interactions that take place (or could take place) in their surroundings. Perceiving the world in this way can lay the basis to learn more complex concepts. Whereas there has been much progress in building models and systems for affordance perception and learning in the fields of robotics and computer vision, there are still significant challenges that remain open. Examples of these challenges are the imposition of specific geometric parametrizations (e.g. planes or cubes), or the need for complex feature representations that can compromise real-time performance of the system in realistic and novel scenarios. Additional challenges faced by current affordance perception systems are their restriction to small sets of objects and affordance categories, and the requirement for large amounts of training examples or costly learning stages.

As reviewed in the related literature in Chapter 2, works investigating affordances have used a variety of features computed usually from visual information. However, the representation that remains present across multiple approaches is shape or geometry. Geometrical information has an outstanding power to richly represent functional cues for objects in everyday tasks, as it is geometry who usually determines the possible physical interactions that objects afford.

This thesis addressed the problem of the determination of affordance locations in 3D environments based purely on geometric information. Inspired by ideas for functional analysis in computer graphics, methods have been presented for the prediction of affordances from visual information, i.e. pointclouds from RGB-D sensors. The problem of affordance detection has been framed as the problem of answering questions such as "*Where can I afford to do this?*" and "*What can I afford to do here?*" in a fast manner for a multitude of interactions, while remaining agnostic to semantic information and complex surface features. A fundamental part of the proposed approach is that it builds on the characterisation of the interaction between two entities, not in the study of the elements in isolation, i.e. scene and query-object.

7.1 Summary of Contributions

The details on the contributions made during the development of this research are as follows:

- A 3D geometrical representation for the characterisation of interactions between two rigid entities in space, namely, the **Interactions Tensor**. The approach, introduced in Chapter 3, is able to robustly characterise the interaction between two rigid, solid entities in 3D space. Moreover, the generality of the method not only allows to study everyday objects such as a cup or a screwdriver but also to model human interactions with the environment, e.g. *Sitting* or *Riding*, by considering a 3D model of a human body. The ability to represent this kind of *human* affordances can be useful, for instance, for robotic assistants suggesting to the elderly the best place to sit; or more generally, scenarios where the task or objects do not require manipulation by the robotic platform. Figure 3.8 of Chapter 3 shows the outstanding capabilities of the interaction tensor to capture key geometrical information across similar interactions. More examples can be seen in Figures A.8-A.14 in Appendix A.3.

- An affordance descriptor that can be conveniently obtained by sampling keypoints from an interaction tensor representation. These weighted and sparsely-sampled **affordance keypoints** allow for a fast evaluation of the likelihood of an affordance taking place in a novel scenario. The proposed similarity function takes advantage of the provenance component embedded in the descriptor, i.e. **provenance vectors**, in order to avoid expensive computations at prediction time. The similarity function accounts for possible noisy geometric estimates in the input data, which provides the possibility of predicting affordance location in never-before-seen RGB-D scenarios from one shot, i.e. from a single training example.

This is a central part of the proposed approach: the ability to perform one-shot affordance predictions. As a result of the proposed detection algorithm and taking advantage of GPU computing power of a commodity PC, affordance predictions can be made at rates suitable for robotic applications. The general view of the proposed algorithm for one-shot affordance prediction approach can be seen in Chapter 4 Figures 4.1 - 4.2.

- A scalable **one-shot learning** algorithm that allows to efficiently query **multiple affordances** at any given location in novel environments. The approach leverages affordance keypoint clustering and a **highly parallelisable** algorithm that enables the detection of over 80 affordance-object pairs in just 150 ms per test-point. Results of these fast-rate affordance predictions for a publicly available dataset of indoor scenes were shown in Figure 4.18 of Chapter 4. These predictions purely based in the geometric information of the input pointclouds were validated via crowdsourcing, which showed that the proposed affordance locations align well with the human criteria, and those above the optimal threshold are regarded as good all of the time.

- A **hybrid approach** that combines data-driven abstraction power and rich geometric information to learn compact, **multiple-affordance** representations that allow to predict meaningful interaction opportunities at high frame rates. This hybrid approach, presented in Chapter 5, leverages the ability of the one-shot detection algorithm to generate in an automatic manner pointcloud data collections suitable to train a state-of-the-art deep neural network. The network allowed to learn an **optimised and compact** descriptor of the interactions opportunities between multiple objects and the environment. The devised descriptor enabled the ability to produce multiple-affordance predictions at high frame rates that outperformed both methods in isolation, as well as surpassing the performance of alternative approaches.

Lastly, and as a demonstration of the quality and high prediction rates of the proposed representation, Chapter 6 showcased two applications of the affordance detection algorithm. These applications included simulations of a robotic platform that is able to detect interaction possibilities in its surroundings while navigating a novel scene. Then, the affordance lantern concept was presented, which demonstrated an affordance-based scene augmentation application. This application uses a standard RGB-D sensor with open source dense mapping systems to *dynamically* discover affordance candidate locations; the predicted interactions can later be embedded in the image stream of the camera, i.e. an augmented reality system.

7.2 Limitations and Future work

The current approach uses only geometry to compute the likelihood of a location in the scene of affording an interaction. This assumes that the sensor is able to perceive the scene correctly; errors in the sensing could affect the performance or quality of the predictions. Whereas most of the interactions do not need a high-quality reconstruction of the scene geometry, some others do. For instance *Hanging* affordances, where the system will "try to hang" objects from edges of flat surfaces or from thin and elongated patches of points. The former is very common when the sensor does not see certain parts of scene correctly; as an example, a hole in the middle of a pointcloud would "appear" to be apt for hanging objects (e.g. an umbrella or a coat hanger) around the edges of the perceived opening. One way to avoid such artefacts is, as shown for the affordance lantern, by performing visibility check with the object in the predicted pose. If the objects were to be hanged from a non-existent hole, they would not pass the visibility test, and therefore those predictions could be avoided. This type of problem could also be mitigated by assuming an active perception approach [244], where the agent can explore or examine parts of the scene that seem *interesting* regarding its need (e.g. to hang an umbrella). By doing so, the agent could obtain a better geometry estimate through the sensor and consequently better affordance predictions. Alternatively, and as shown for the affordance lantern application, it is possible to mitigate some of this issues by focusing the detection on the “eye's fovea” or region of interest (e.g. centre of the image), where the pointcloud geometry is usually closer to the real scene.

In a related issue, it is a known fact that state-of-the-art RGB-D sensors do not cope well with reflective surfaces such as those commonly found in taps and sinks in kitchen environments, or glass in windows or cabinet doors. If the camera does not see these objects, it is very unlikely that their affordances are predicted *correctly* or predicted at all. This can be observed from the results on scenes from the ScanNet dataset, where none of the scenes had taps or sinks in them and the method was not able to predict realistic *Filling* affordances. This issue, however, can be overcome with approaches that do not rely on structured light to produce pointcloud representations of the scene such as [245–247], which estimate depth maps directly from RGB images.

In chapter 5 a deep neural network was employed to extract salient locations in the input that lead to the correct classification of multiple affordances happening in a single pointcloud. Although this proved to work by helping to devise a compact and optimal descriptor, it was assumed that the probabilities of the affordances in a given pointcloud were independent of each other (e.g. binary cross entropy loss). A more interesting approach would be to learn co-occurrences or interdependencies that exist among affordances, perhaps through a probabilistic graphical model such as those used in developmental robotics approaches or with a loss function that accounts for those dependencies [248]. For instance, computing a high probability for one affordance (or a set of them) taking place in a scene location could prevent spending computations on testing other less likely affordances. As an example, detecting a high probability of *Filling* a mug would probably discard testing for *Placing, Sitting* or *Riding* affordances.

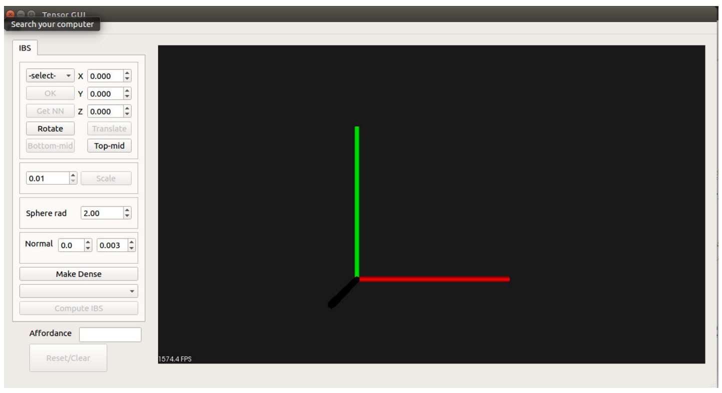
Finally, the proposed representation has shown to be able to characterise generic interactions between two entities in the environment robustly. In this sense, it has been assumed that the query-object is already in the agent's hand; thus, the goal was to detect where the agent could *Place, Hang* or *Fill* this object. An interesting avenue for future work is the investigation of grasping affordances, i.e. the interaction between a hand and other objects. For instance, the model of the robot's hand would serve as a query-object that interacts with a scene-object in the environment.

In summary, this thesis has shown that top-quality affordance detections can be achieved by exploiting the geometric information involved in the interaction between two entities. Among the main benefits of the approach is its generic properties, allowing the description of interactions between everyday objects and the environment as well as the interactions between a human and the environment (e.g. *Sitting*). We have shown that geometric information suffices to predict valid affordance locations in unknown environments; however, this does not mean that other sources of information should be discarded. For instance, it would be very interesting to explore the incorporation of material or physics information, which are also tightly related to the interactions that the environment affords.

APPENDIX A

APPENDIX A

A.1 AffordanceSim



X 0.000
Y 0.000
Z 0.000
0.01
Sphere rad 2.00
Normal 0.0 0.003
Affordance
1574.4 FPS

Figure A.1: GUI main window

A.1.1 Computing an interaction example

In order to compute an Interaction Tensor for *training*, three things are needed:

1. Load an object model or dense pointcloud as **scene-object**
2. Load an object model or dense pointcloud as **query-object**
3. Place the objects simulating the intended interaction

Execute the binary to bring up the GUI. The first step is to load the scene-object (e.g. table for *Placing*, sink for *Filling*, etc.). The five scene-objects used for the interactions presented in the ICRA'18 paper; the objects are a table, a sink, a hanging-rack, a motorcycle and a sitting stool. The GUI allows to quickly load these by selecting the associated interaction from the dropdown menu on the upper left area.

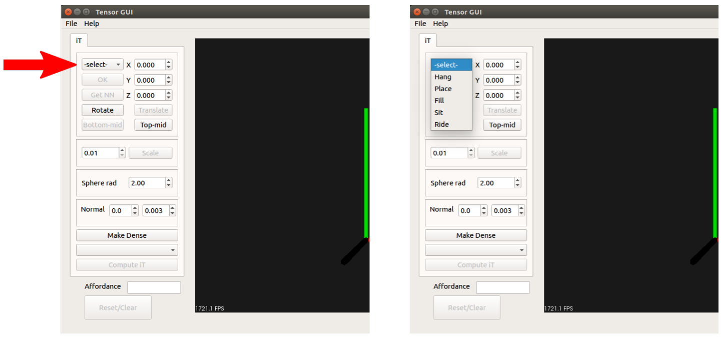


Figure A.2: Affordance selection on GUI

The scene-object will be displayed in red (pointcloud), and a test-point will be selected (hard-coded but editable).

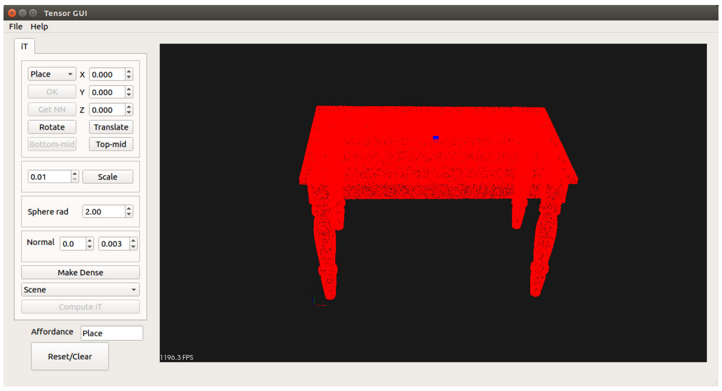


Figure A.3: GUI reference point selection

This test-point is only a guide to align or place the objects simulating the interactions. For instance, in the case of a table the test-point is set to the centre on top of the table, or in case of a motorbike, it would be the centre on top of the saddle. Different points can be chosen by selecting a point directly from the pointcloud (left click on the mouse while holding shift). A second way of selecting a test-point is by using the bottom-mid and top-mid buttons; these will select the centre (top or bottom) point of the bounding box of the current object/pointcloud. As an example, when the table has been loaded clicking bottom-mid will select a point "on the floor" in middle of the table legs.

If test-points are manually selected confirmation will be needed by clicking the **OK** button on the upper left. Other object CAD models (PLY/OBJ files) or pointclouds (PCD file) can be loaded using the File menu: **File -> Load**. Make sure the model/pointcloud meets the following criteria:

- Model/pointcloud is oriented correctly, the system assumes the gravity vector along the negative part of the Z axis.

- Scene-object needs '*scene-*' as base name. Examples of the naming convention can be seen in the files of the **data** directory.

The GUI can create dense pointclouds by clicking on the **Make dense** button (300000 points uniformly sampled). This number can be changed; have a look at the *denseButtonPressed()* function in src/mainwindow.cpp

A similar process is followed to load a query-object and get a dense pointcloud. As an example try loading the bowl model located under data/ directory (bowl.ply) and click on **Make dense** button to get a dense pointcloud.

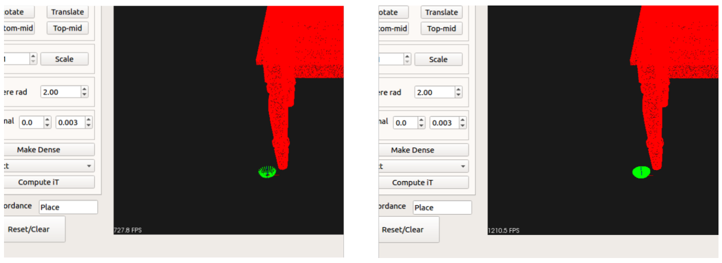
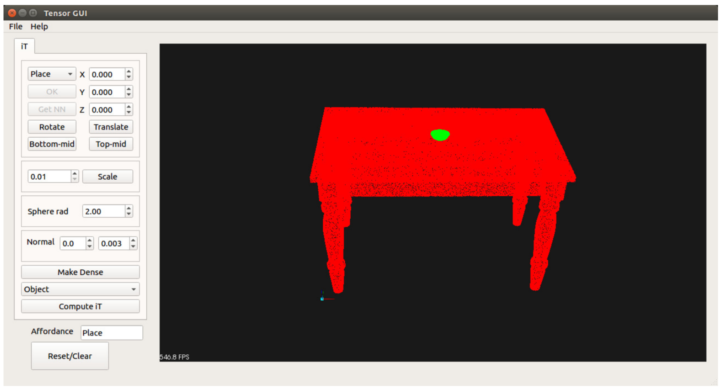


Figure A.4: CAD model to dense pointcloud

To place an object on top of the table (for *Placing* affordance) two points need to be selected: a test-point on the query-object and a test-point on the scene-object. For a *Placing* a bowl try clicking bottom-mid for the bowl and top-mid for the table. Once both test-points have been selected click on "Translate" button to apply the transformation.



Place X 0.000
Y 0.000
Z 0.000
0.01
Sphere rad 2.00
Normal 0.0 0.003
Object
Affordance Place

Figure A.5: Interaction example ready to compute tensor

Notice that the text-box with the label "Affordance" was automatically filled with **Place**. This text can be changed manually according to the affordance or interaction being simulated.

The final step at this point is to click on the **Compute iT** button which will compute the interaction tensor, sample affordance keypoints and produced a lot of auxiliary files. Some these auxiliary files are needed in order to make predictions.

A.2 Scene Database

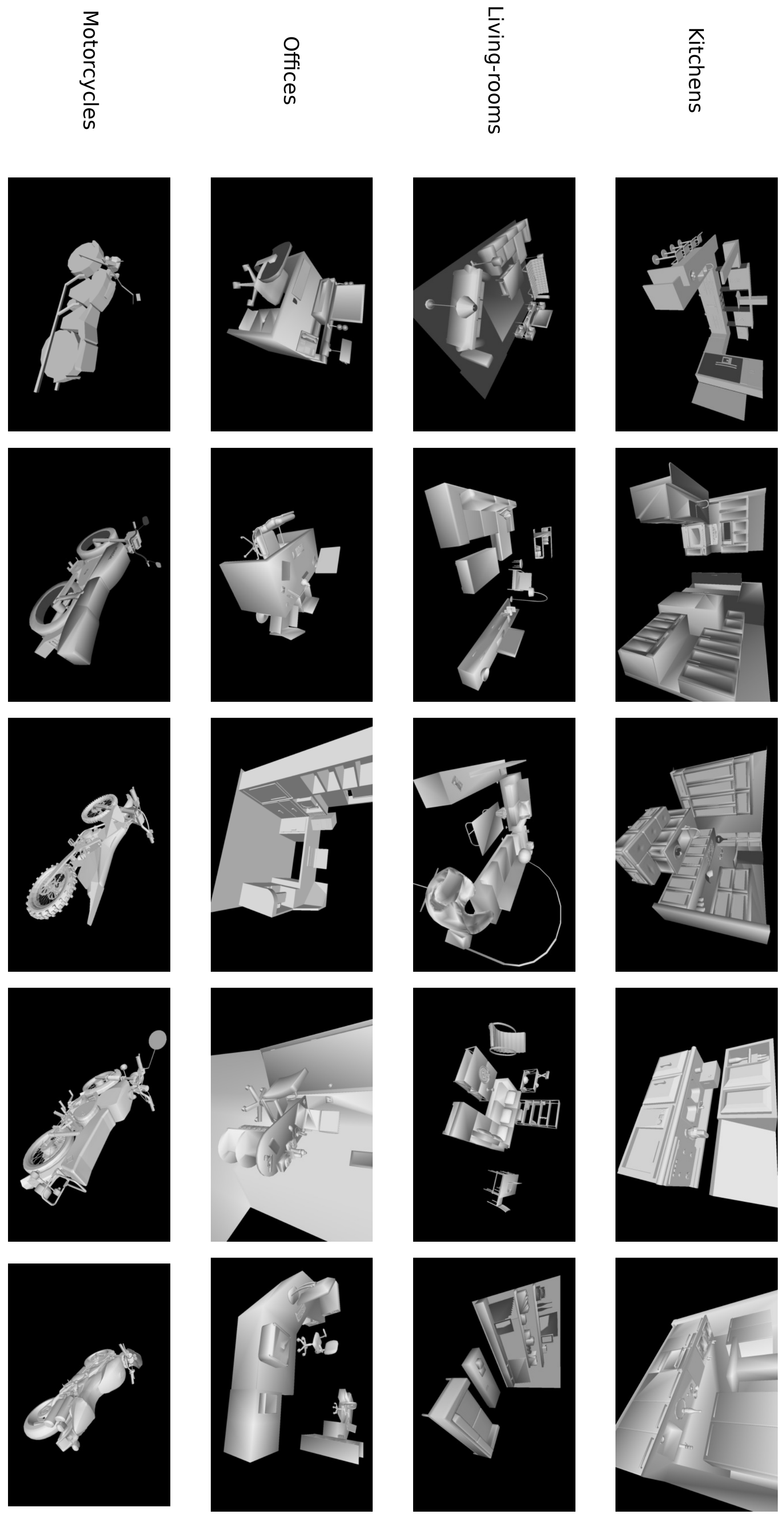


Figure A.6: Synthetic scenes from publicly available repositories

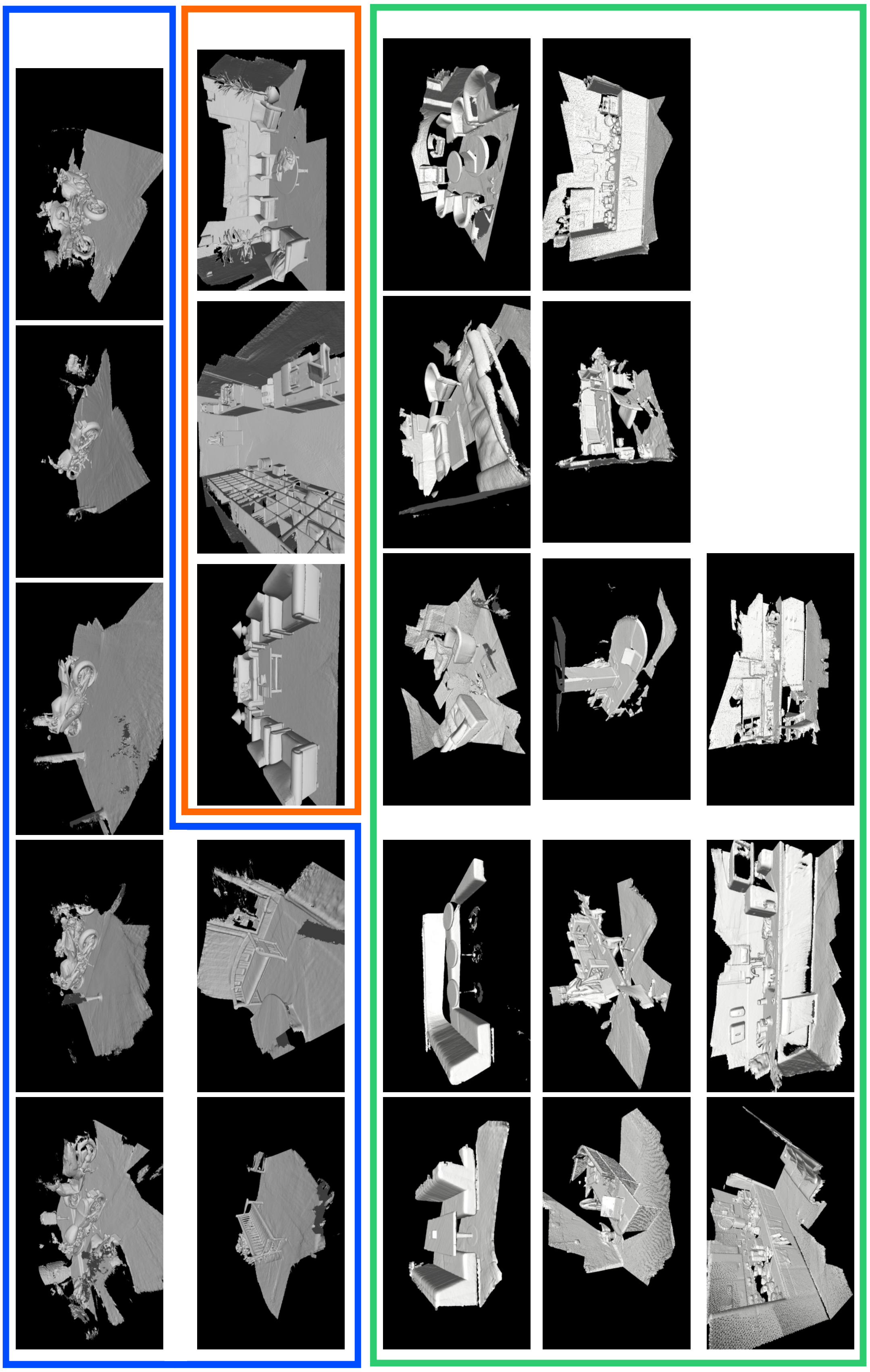


Figure A.7: RGB-D scans of real environments. In blue at the top scenes from [212]. Highlighted in orange are indoor scans from [211]. In green are shown our scans captured with [210]

A.3 Interaction Tensors

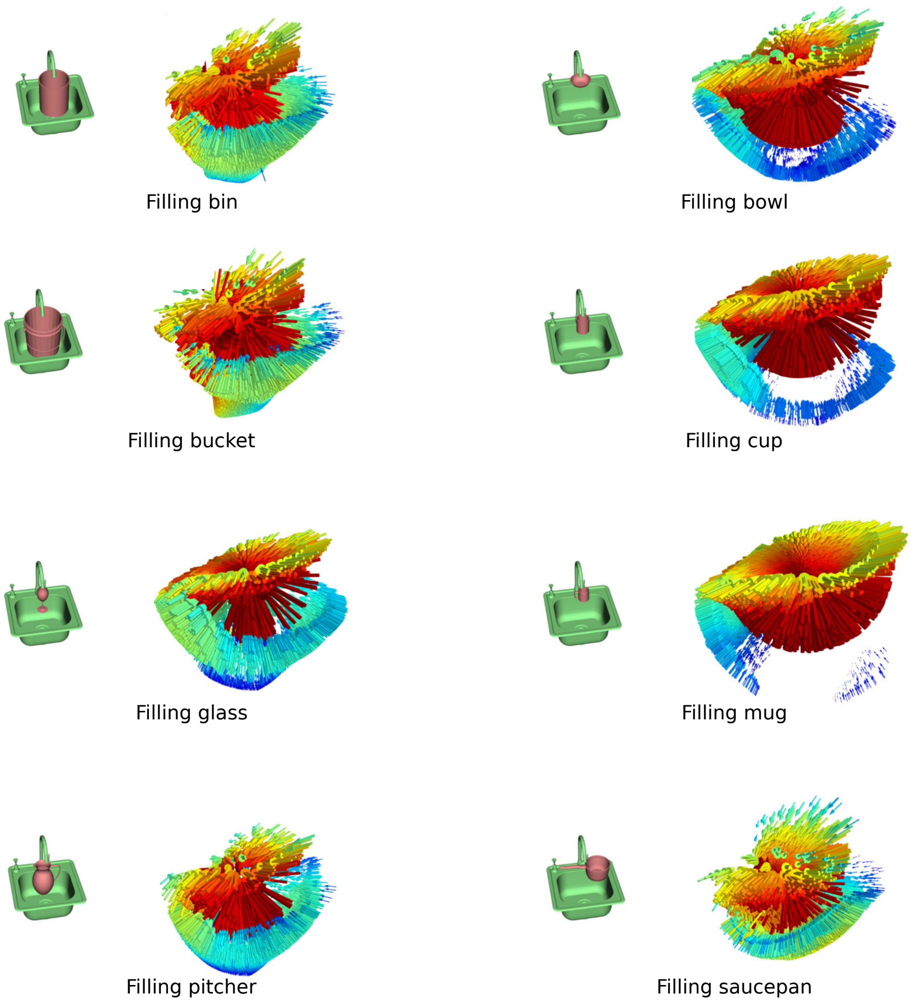


Figure A.8: Interactions tensors for *Filling*

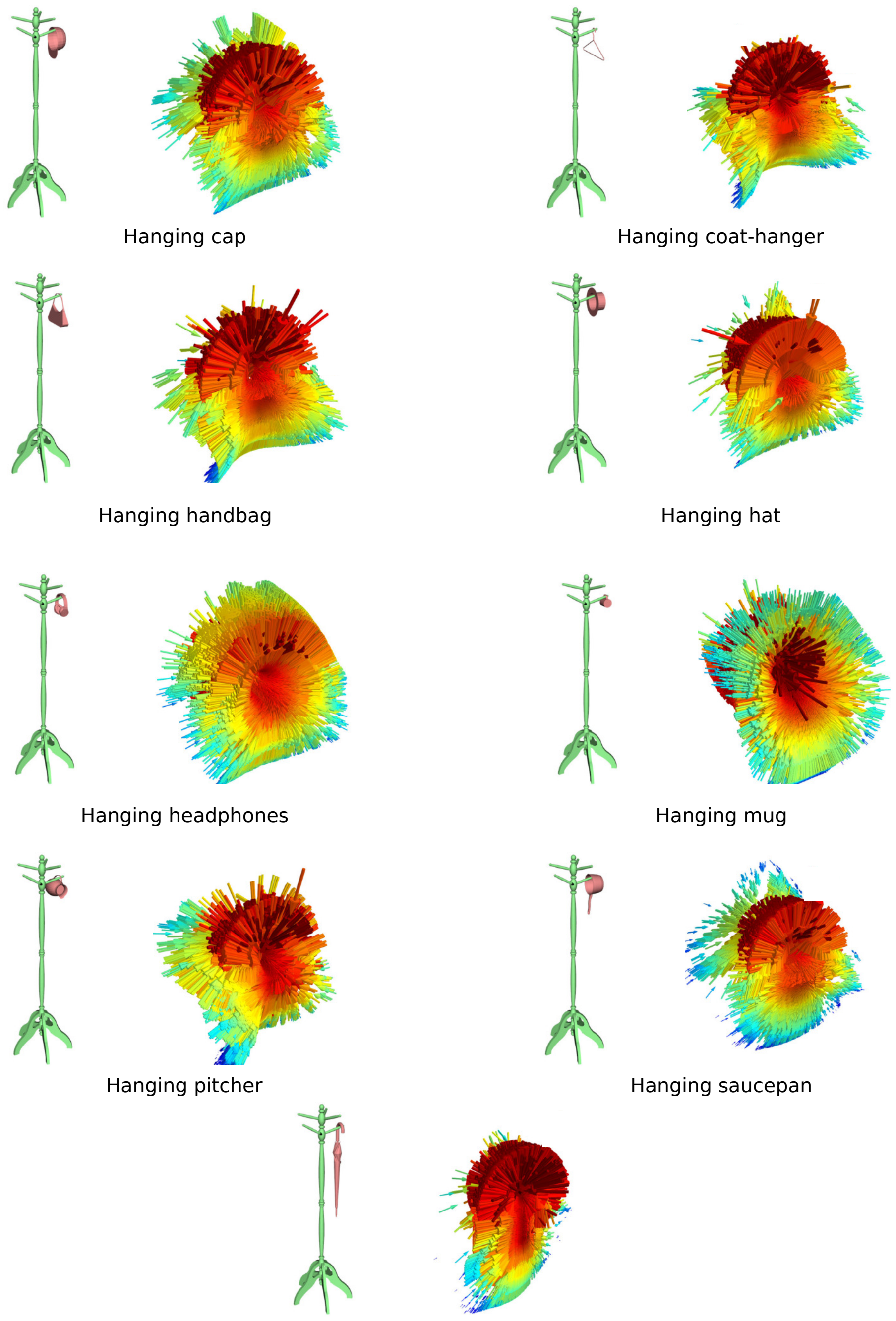


Figure A.9: Interactions tensors for *Hanging*

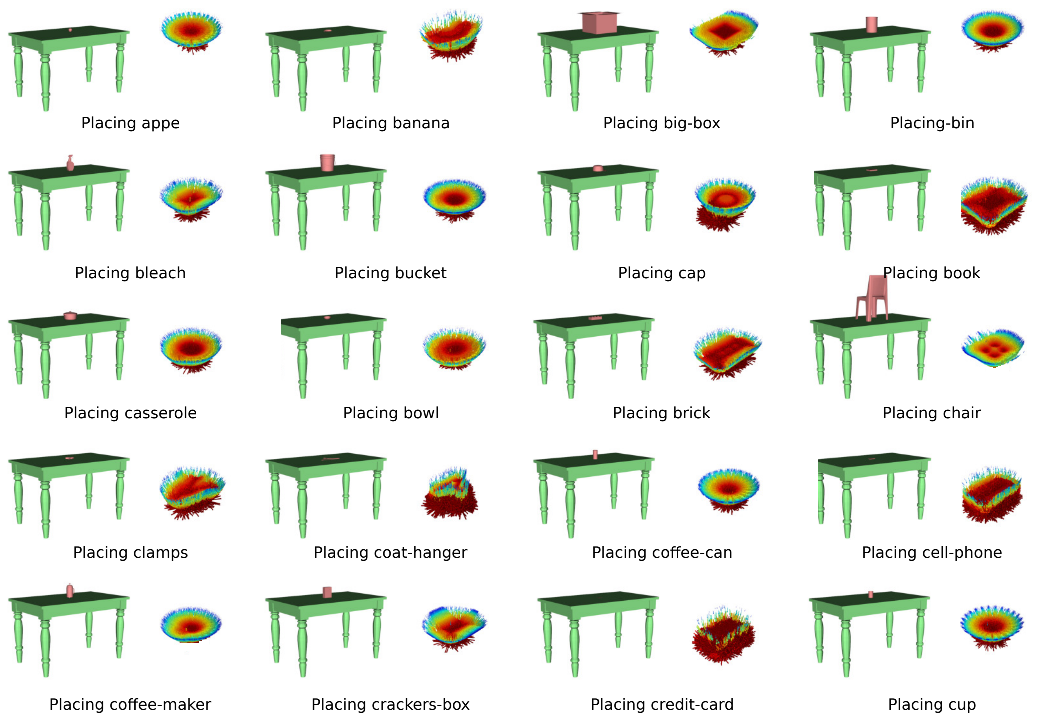


Figure A.10: Interactions tensors for *Placing*(1)

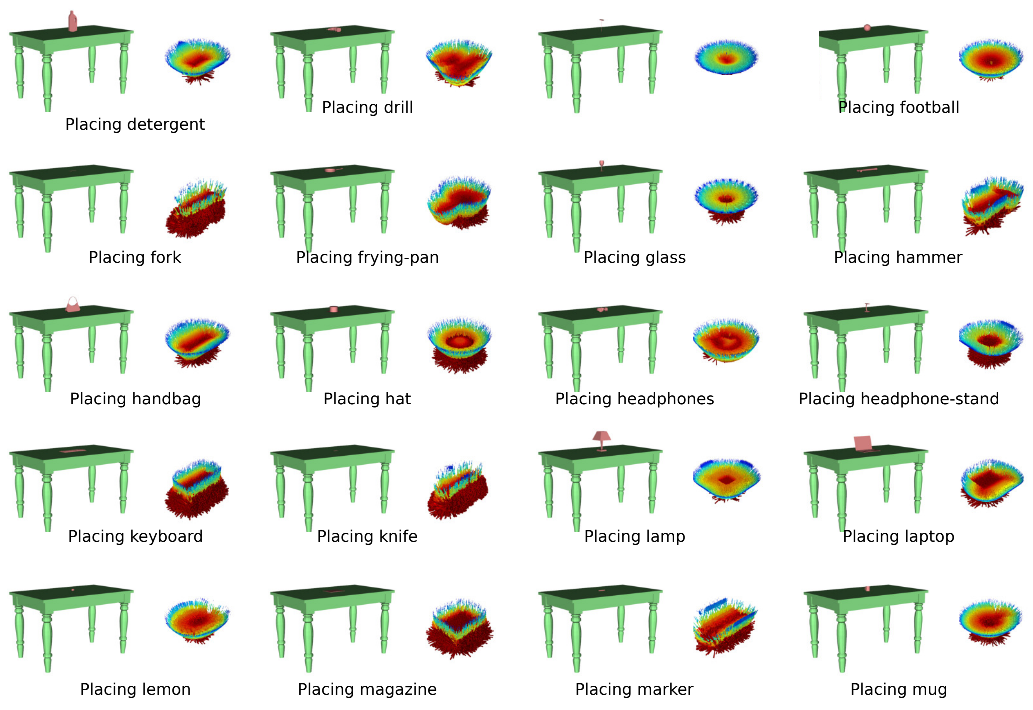


Figure A.11: Interactions tensors for *Placing*(2)

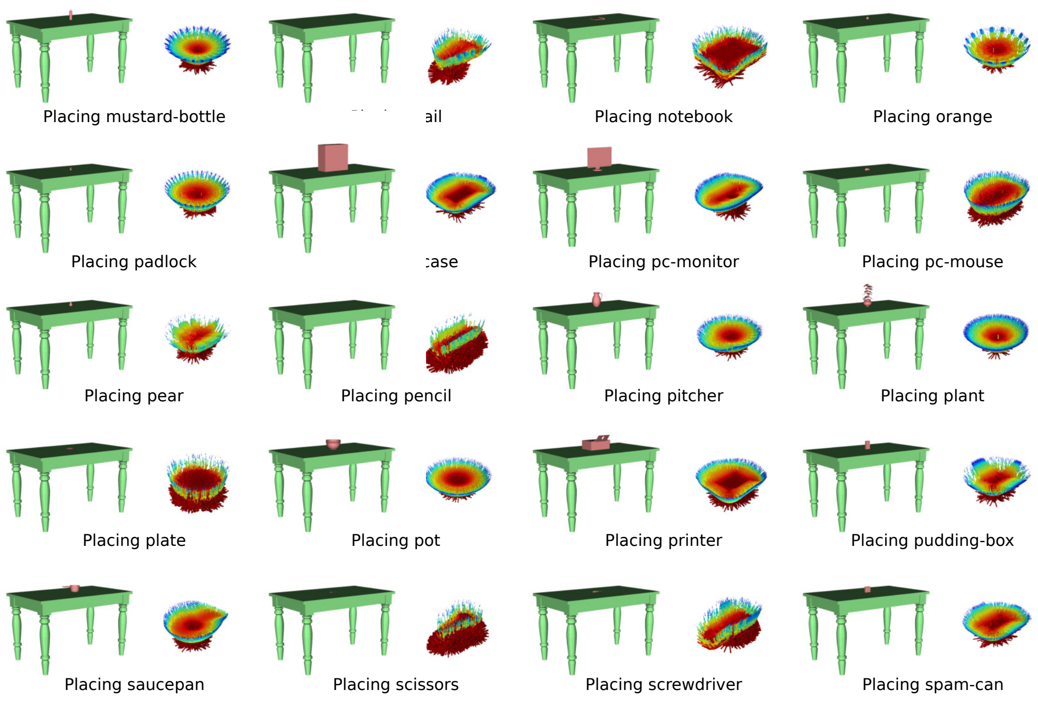


Figure A.12: Interactions tensors for *Placing*(3)

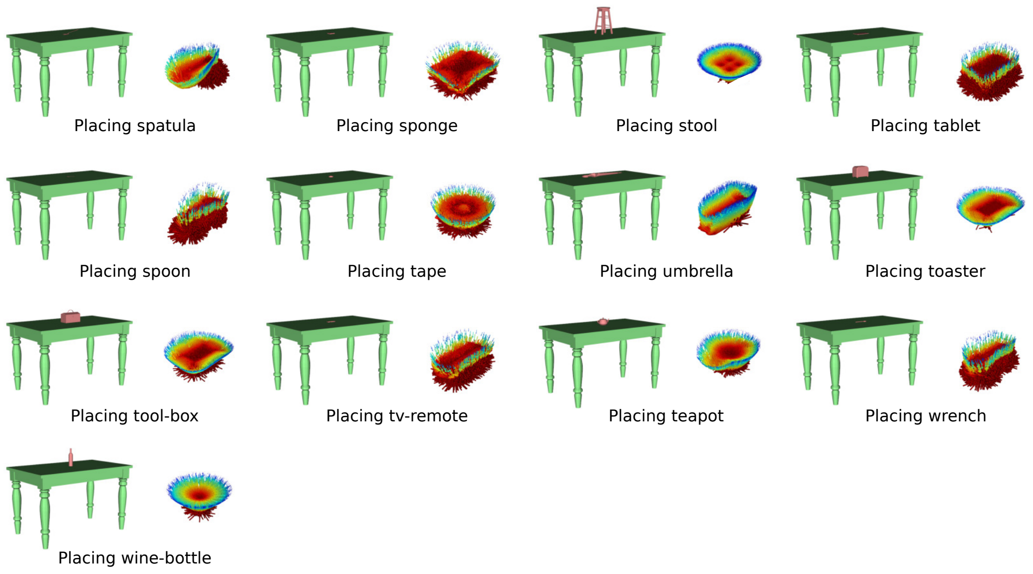


Figure A.13: Interactions tensors for *Placing*(4)

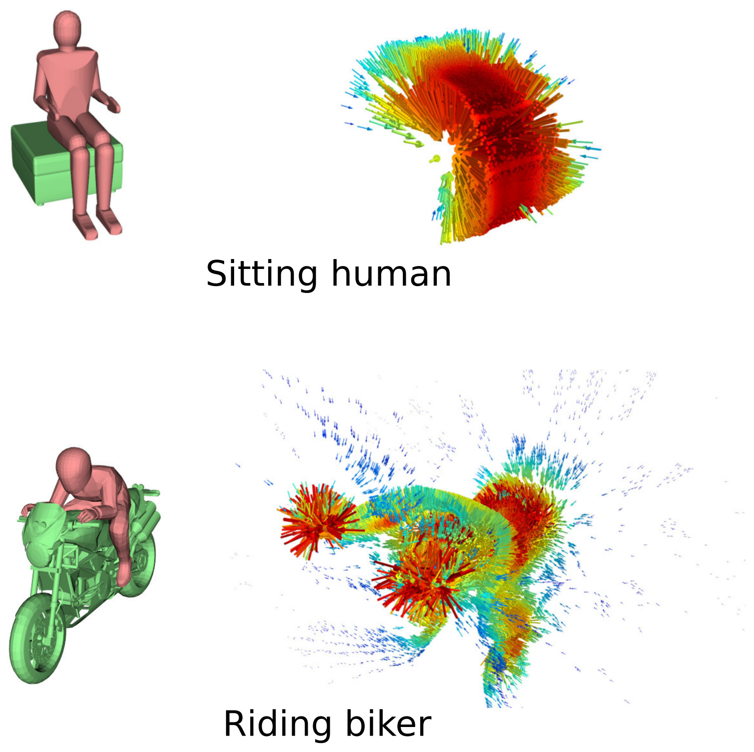


Figure A.14: Interactions tensors for *Sitting* and *Riding*

Bibliography

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

[2] D. Marr, “Vision: A computational investigation into the human representation and processing of visual information,” *New York, NY: W.H. Freeman and Company*, vol. 8, Nov. 1981.

[3] M. A. Goodale and A. D. Milner, “Separate visual pathways for perception and action,” *Trends in neurosciences*, vol. 15, no. 1, pp. 20–25, 1992.

[4] A. David Milner and M. Goodale, “The visual brain in action,” *Optometry and Vision Science - OPTOMETRY VISION SCI*, vol. 74, Jan. 1997.

[5] G. Rizzolatti and L. Fadiga, “Grasping objects and grasping action meanings: The dual role of monkey rostroventral premotor cortex (area f5),” *Sensory guidance of movement*, vol. 218, pp. 81–103, 1998.

[6] G. Vingerhoets, K. Vandamme, and A. Vercammen, “Conceptual and physical object qualities contribute differently to motor affordances,” *Brain and Cognition*, vol. 69, no. 3, pp. 481–489, 2009.

[7] M. Jeannerod, M. A. Arbib, G. Rizzolatti, and H. Sakata, “Grasping objects: The cortical mechanisms of visuomotor transformation,” *Trends in neurosciences*, vol. 18, no. 7, pp. 314–320, 1995.

[8] A. Murata, V. Gallese, G. Luppino, M. Kaseda, and H. Sakata, “Selectivity for the shape, size, and orientation of objects for grasping in neurons of monkey parietal area aip,” *Journal of Neurophysiology*, vol. 83, no. 5, pp. 2580–2601, 2000.

[9] E. Oztop, H. Imamizu, G. Cheng, and M. Kawato, “A computational model of anterior intraparietal (aip) neurons,” *Neurocomputing*, vol. 69, no. 10-12, pp. 1354–1361, 2006.

[10] J. J. Gibson, “The senses considered as perceptual systems.,” 1966.

[11] T. E. Horton, A. Chakraborty, and R. S. Amant, “Affordances for robots: A brief survey,” *Avant: Trends in Interdisciplinary Studies*, vol. 3, no. 2, pp. 70–84, 2012.

[12] H. Min, C. Yi, R. Luo, J. Zhu, and S. Bi, “Affordance research in developmental robotics: A survey,” *IEEE Transactions on Cognitive and Developmental Systems*, vol. 8, no. 4, pp. 237–255, 2016.

[13] P. Zech, S. Haller, S. R. Lakani, B. Ridge, E. Ugur, and J. Piater, “Computational models of affordance in robotics: A taxonomy and systematic classification,” *Adaptive Behavior*, vol. 25, no. 5, pp. 235–271, 2017.

[14] L. Jamone, E. Ugur, A. Cangelosi, L. Fadiga, A. Bernardino, J. Piater, and J. Santos-Victor, “Affordances in psychology, neuroscience, and robotics: A survey,” *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 1, pp. 4–25, 2018.

[15] M. Hassanin, S. Khan, and M. Tahtali, “Visual affordance and function understanding: A survey,” *arXiv preprint arXiv:1807.06775*, 2018.

[16] M.T.Turvey, “Affordances and prospective control: An outline of the ontology,” *Ecological Psychology*, vol. 4, no. 3, pp. 173–187, 1992.

[17] A. H. Vera and H. A. Simon, “Situated action: A symbolic interpretation,” *Cognitive Science*, vol. 17, no. 1, pp. 7–48, 1993.

[18] R. Shaw, “Processes, acts, and experiences: Three stances on the problem of intentionality,” *Ecological Psychology*, vol. 13, no. 4, pp. 275–314, 2001.

[19] H. Heft, “Affordances, dynamic experience, and the challenge of reification,” *Ecological Psychology*, vol. 15, no. 2, pp. 149–180, 2003.

[20] T. A. Stoffregen, “Affordances as properties of the animal-environment system,” *Ecological Psychology*, vol. 15, no. 2, pp. 115–134, 2003.

[21] A. Chemero, “An outline of a theory of affordances,” *Ecological Psychology*, vol. 15, no. 2, pp. 181–195, 2003.

[22] C. F. Michaels, “Affordances: Four points of debate,” *Ecological Psychology*, vol. 15, no. 2, pp. 135–148, 2003.

[23] D. A. Norman, *The Design of Everyday Things*. New York, NY, USA: Basic Books, Inc., 2002.

[24] M. Steedman, “Plans, affordances, and combinatory grammar,” *Linguistics and Philosophy*, vol. 25, no. 5, pp. 723–753, Dec. 2002.

[25] E. Şahin, M. Çakmak, M. R. Doğar, E. Uğur, and G. Üçoluk, “To afford or not to afford: A new formalization of affordances toward affordance-based robot control,” *Adaptive Behavior*, vol. 15, no. 4, pp. 447–472, 2007.

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

[27] N. Krüger, C. Geib, J. Piater, R. Petrick, M. Steedman, F. Wörgötter, A. Ude, T. Asfour, D. Kraft, D. Omrčen, A. Agostini, and R. Dillmann, “Object–action complexes: Grounded abstractions of sensory–motor processes,” *Robotics and Autonomous Systems*, vol. 59, no. 10, pp. 740–757, 2011.

[28] M. Mishkin, L. G. Ungerleider, and K. A. Macko, “Object vision and spatial vision: Two cortical pathways,” *Trends in neurosciences*, vol. 6, pp. 414–417, 1983.

[29] J. Norman, “Two visual systems and two theories of perception: An attempt to reconcile the constructivist and ecological approaches,” *Behavioral and brain sciences*, vol. 25, no. 1, pp. 73–96, 2002.

[30] J. C. Culham and K. F. Valyear, “Human parietal cortex in action,” *Current Opinion in Neurobiology*, vol. 16, no. 2, pp. 205–212, 2006, Cognitive neuroscience.

[31] L. G. Nowak and J. Bullier, “The timing of information transfer in the visual system,” in *Extrastriate cortex in primates*, Springer, 1997, pp. 205–241.

[32] A. Milner, D. Perrett, R. Johnston, P. Benson, T. Jordan, D. Heeley, D. Bettucci, F. Mortara, R. Mutani, E. Terazzi, *et al.*, “Perception and action in ‘visual form agnosia’,” *Brain*, vol. 114, no. 1, pp. 405–428, 1991.

[33] J. Norman, “Ecological psychology and the two visual systems: Not to worry!” *Ecological Psychology*, vol. 13, no. 2, pp. 135–145, 2001.

[34] G. Rizzolatti and M. Matelli, “Two different streams form the dorsal visual system: Anatomy and functions,” *Experimental Brain Research*, vol. 153, no. 2, pp. 146–157, Nov. 2003.

[35] A. Murata, L. Fadiga, L. Fogassi, V. Gallese, V. Raos, and G. Rizzolatti, "Object representation in the ventral premotor cortex (area f5) of the monkey," *Journal of neurophysiology*, vol. 78, no. 4, pp. 2226–2230, 1997.

[36] G. Rizzolatti and L. Fadiga, "Grasping objects and grasping action meanings: The dual role of monkey rostroventral premotor cortex (area f5).," *Novartis Foundation symposium*, vol. 218, pp. 81–95, 81–95, 1998.

[37] M. Franca, L. Turella, R. Canto, N. Brunelli, L. Allione, N. G. Andreasi, M. Desantis, D. Marzoli, and L. Fadiga, "Corticospinal facilitation during observation of graspable objects: A transcranial magnetic stimulation study," *PLOS ONE*, vol. 7, no. 11, pp. 1–9, Nov. 2012.

[38] P. Cardellicchio, C. Sinigaglia, and M. Costantini, "The space of affordances: A tms study," *Neuropsychologia*, vol. 49, no. 5, pp. 1369–1372, 2011.

[39] M. A. Goodale, "Action without perception in human vision," *Cognitive Neuropsychology*, vol. 25, no. 7-8, pp. 891–919, 2008.

[40] G. Vingerhoets, K. Vandamme, and A. Vercammen, "Conceptual and physical object qualities contribute differently to motor affordances," *Brain and Cognition*, vol. 69, no. 3, pp. 481–489, 2009.

[41] G. Humphreys, "Objects, affordances ... action!" *The Psychologist*, vol. 14, pp. 408–412, Aug. 2001.

[42] D. H. Ballard and C. M. Brown, "Principles of animate vision," *CVGIP: Image Understanding*, vol. 56, no. 1, pp. 3–21, 1992, Purposive, Qualitative, Active Vision.

[43] A. P. Duchon, L. P. Kaelbling, and W. H. Warren, "Ecological robotics," *Adaptive Behavior*, vol. 6, no. 3-4, pp. 473–507, 1998.

[44] R. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal on Robotics and Automation*, vol. 2, no. 1, pp. 14–23, Mar. 1986.

[45] M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, and C. Yoshida, "Cognitive developmental robotics: A survey," *Autonomous Mental Development, IEEE Transactions on*, vol. 1, no. 1, pp. 12–34, May 2009.

[46] A. Cangelosi and M. Schlesinger, *Developmental robotics: From babies to robots*. MIT Press, 2015.

[47] P. Fitzpatrick, G. Metta, L. Natale, S. Rao, and G. Sandini, "Learning about objects through action - initial steps towards artificial cognition," in *2003 IEEE International Conference on Robotics and Automation*, vol. 3, Sep. 2003, 3140–3145 vol.3.

[48] G. Metta and P. Fitzpatrick, "Better vision through manipulation," *Adaptive Behavior*, vol. 11, no. 2, pp. 109–128, 2003.

[49] Fritz, Paletta, Breithaupt, and Rome, "Learning predictive features in affordance based robotic perception systems," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pp. 3642–3647.

[50] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[51] L. Paletta, G. Fritz, F. Kintzler, J. Irran, and G. Dorffner, "Perception and developmental learning of affordances in autonomous robots," in *KI 2007: Advances in Artificial Intelligence*, J. Hertzberg, M. Beetz, and R. Englert, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 235–250.

[52] E. Ugur, M. R. Dogar, M. Cakmak, and E. Sahin, "The learning and use of traversability affordance using range images on a mobile robot," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, Apr. 2007, pp. 1721–1726.

[53] M. R. Dogar, M. Cakmak, E. Ugur, and E. Sahin, "From primitive behaviors to goal-directed behavior using affordances," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2007, pp. 729–734.

[54] M. Cakmak, M. Dogar, E. Ugur, and E. Sahin, "Affordances as a framework for robot control," in *Proceedings of the 7th international conference on epigenetic robotics, epirob'07*, 2007.

[55] M. R. Dogar, E. Ugur, E. Sahin, and M. Cakmak, "Using learned affordances for robotic behavior development," in *2008 IEEE International Conference on Robotics and Automation*, May 2008, pp. 3802–3807.

[56] E. Uğur and E. Şahin, “Traversability: A case study for learning and perceiving affordances in robots,” *Adaptive Behavior*, vol. 18, no. 3-4, pp. 258–284, 2010.

[57] D. Kim, J. Sun, S. M. Oh, J. M. Rehg, and A. F. Bobick, “Traversability classification using unsupervised on-line visual learning for outdoor robot navigation,” in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, May 2006, pp. 518–525.

[58] K. I. Laws, “Rapid texture identification,” vol. 0238, 1980.

[59] J. Sun, J. L. Moore, A. Bobick, and J. M. Rehg, “Learning visual object categories for robot affordance prediction,” *The International Journal of Robotics Research*, vol. 29, no. 2-3, pp. 174–197, 2010.

[60] T. Hermans, J. Rehg, and A. Bobick, “Guided pushing for object singulation,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct. 2012, pp. 4783–4790.

[61] T. Hermans, J. M. Rehg, and A. F. Bobick, “Decoupling behavior, perception, and control for autonomous learning of affordances,” in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 4989–4996.

[62] T. Hermans, F. Li, J. M. Rehg, and A. F. Bobick, “Learning contact locations for pushing and orienting unknown objects,” in *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*.

[63] ——, “Learning stable pushing locations,” in *Development and Learning and Epigenetic Robotics (ICDL), 2013 IEEE Third Joint International Conference on*, IEEE, 2013, pp. 1–7.

[64] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor, “Modeling affordances using bayesian networks,” in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pp. 4102–4107.

[65] M. Lopes, F. S. Melo, and L. Montesano, “Affordance-based imitation learning in robots,” in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pp. 1015–1021.

[66] B. Moldovan, P. Moreno, M. van Otterlo, J. Santos-Victor, and L. De Raedt, "Learning relational affordance models for robots in multi-object manipulation tasks," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 4373–4378.

[67] B. Moldovan, P. Moreno, and M. van Otterlo, "On the use of probabilistic relational affordance models for sequential manipulation tasks in robotics," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 1290–1295.

[68] B. Moldovan and L. De Raedt, "Occluded object search by relational affordances," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2014, pp. 169–174.

[69] ——, "Learning relational affordance models for two-arm robots," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, Sep. 2014, pp. 2916–2922.

[70] G. Salvi, L. Montesano, A. Bernardino, and J. Santos-Victor, "Language bootstrapping: Learning word meanings from perception–action association," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 3, pp. 660–671, Jun. 2012.

[71] J. Sinapov and A. Stoytchev, "Learning and generalization of behavior-grounded tool affordances," in *2007 IEEE 6th International Conference on Development and Learning*, Jul. 2007, pp. 19–24.

[72] V. Tikhanoff, U. Pattacini, L. Natale, and G. Metta, "Exploring affordances and tool use on the icub," in *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Oct. 2013, pp. 130–137.

[73] A. Goncalves, J. Abrantes, G. Saponaro, L. Jamone, and A. Bernardino, "Learning intermediate object affordances: Towards the development of a tool concept," in *Development and Learning and Epigenetic Robotics (ICDL-Epirob), 2014 Joint IEEE International Conferences on*, pp. 482–488.

[74] A. Goncalves, G. Saponaro, L. Jamone, and A. Bernardino, "Learning visual affordances of objects and tools through autonomous robot exploration," in *Autonomous Robot Systems and Competitions (ICARSC), 2014 IEEE International Conference on*, pp. 128–133.

[75] A. Goncalves, J. Abrantes, G. Saponaro, L. Jamone, and A. Bernardino, "Learning intermediate object affordances: Towards the development of a tool concept," in *Development and Learning and Epigenetic Robotics (ICDL-Epirob), 2014 Joint IEEE International Conferences on*, Oct. 2014, pp. 482–488.

[76] T. Mar, V. Tikhanoff, G. Metta, and L. Natale, "Self-supervised learning of grasp dependent tool affordances on the icub humanoid robot," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 3200–3206.

[77] A. Dehban, L. Jamone, A. R. Kampff, and J. Santos-Victor, "Denoising auto-encoders for learning of objects and tools affordances in continuous space," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 4866–4871.

[78] A. Antunes, L. Jamone, G. Saponaro, A. Bernardino, and R. Ventura, "From human instructions to robot actions: Formulation of goals, affordances and probabilistic planning," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 5449–5454.

[79] G. Saponaro, P. Vicente, A. Dehban, L. Jamone, A. Bernardino, and J. Santos-Victor, "Learning at the ends: From hand to tool affordances in humanoid robots," in *2017 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, Sep. 2017, pp. 331–337.

[80] T. Mar, V. Tikhanoff, G. Metta, and L. Natale, "Self-supervised learning of tool affordances from 3d tool representation through parallel som mapping," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 894–901.

[81] S. Griffith, J. Sinapov, M. Miller, and A. Stoytchev, "Toward interactive learning of object categories by a robot: A case study with container and non-container objects," in *2009 IEEE 8th International Conference on Development and Learning*, Jun. 2009, pp. 1–6.

[82] S. Griffith, J. Sinapov, V. Sukhoy, and A. Stoytchev, "A behavior-grounded approach to forming object categories: Separating containers from noncontainers," *IEEE Transactions on Autonomous Mental Development*, vol. 4, no. 1, pp. 54–69, Mar. 2012.

[83] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *Proceedings of the 19th International Conference on Neural Information Processing Systems*, ser. NIPS'06, Canada: MIT Press, 2006, pp. 801–808.

[84] E. Ugur, S. Szedmak, and J. Piater, “Bootstrapping paired-object affordance learning with learned single-affordance features,” in *Development and Learning and Epigenetic Robotics (ICDL-Epirob), 2014 Joint IEEE International Conferences on*, pp. 476–481.

[85] E. Ugur and J. Piater, “Emergent structuring of interdependent affordance learning tasks,” in *Development and Learning and Epigenetic Robotics (ICDL-Epirob), 2014 Joint IEEE International Conferences on*, pp. 489–494.

[86] E. Ugur and J. Piater, “Bottom-up learning of object categories, action effects and logical rules: From continuous manipulative exploration to symbolic planning,” in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, IEEE, 2015, pp. 2627–2633.

[87] E. Ugur, Y. Nagai, E. Sahin, and E. Oztop, “Staged development of robot skills: Behavior formation, affordance learning and imitation with motionese,” *IEEE Transactions on Autonomous Mental Development*, vol. 7, no. 2, pp. 119–139, Jun. 2015.

[88] E. Ugur and J. Piater, “Emergent structuring of interdependent affordance learning tasks using intrinsic motivation and empirical feature selection,” *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, no. 4, pp. 328–340, Dec. 2017.

[89] P. Kaiser, D. Gonzalez-Aguirre, F. Schültje, J. Borràs, N. Vahrenkamp, and T. Asfour, “Extracting whole-body affordances from multimodal exploration,” in *2014 IEEE-RAS International Conference on Humanoid Robots*, Nov. 2014, pp. 1036–1043.

[90] P. Kaiser, M. Grotz, E. E. Aksoy, M. Do, N. Vahrenkamp, and T. Asfour, “Validation of whole-body loco-manipulation affordances for pushability and liftability,” in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, Nov. 2015, pp. 920–927.

[91] P. Kaiser, E. E. Aksoy, M. Grotz, and T. Asfour, “Towards a hierarchy of loco-manipulation affordances,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2016, pp. 2839–2846.

[92] P. Kaiser, C. Mandery, A. Boltres, and T. Asfour, “Affordance-based multi-contact whole-body pose sequence planning for humanoid robots in unknown environments,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 3114–3121.

[93] A. Jøsang, “A logic for uncertain probabilities,” *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 9, no. 03, pp. 279–311, 2001.

[94] A. Bicchi and V. Kumar, “Robotic grasping and contact: A review,” in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, Apr. 2000, 348–353 vol.1.

[95] J. Bohg, A. Morales, T. Asfour, and D. Kragic, “Data-driven grasp synthesis ;a survey,” *Robotics, IEEE Transactions on*, vol. 30, no. 2, pp. 289–309, Apr. 2014.

[96] J. D. Sweeney and R. Grupen, “A model of shared grasp affordances from demonstration,” in *2007 7th IEEE-RAS International Conference on Humanoid Robots*, Nov. 2007, pp. 27–35.

[97] M. Stark, P. Lies, M. Zillich, J. Wyatt, and B. Schiele, “Functional object class detection based on learned affordance cues,” in *International conference on computer vision systems*, Springer, 2008, pp. 435–444.

[98] R. Detry, E. Baseski, M. Popovic, Y. Touati, N. Kruger, O. Kroemer, J. Peters, and J. Piater, “Learning object-specific grasp affordance densities,” in *2009 IEEE 8th International Conference on Development and Learning*, Jun. 2009, pp. 1–7.

[99] R. Detry, C. H. Ek, M. Madry, J. Piater, and D. Kragic, “Generalizing grasps across partly similar objects,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 3791–3797.

[100] R. Detry, C. H. Ek, M. Madry, and D. Kragic, “Learning a dictionary of prototypical grasp-predicting parts from grasping experience,” in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 601–608.

[101] D. Song, C. H. Ek, K. Huebner, and D. Kragic, “Multivariate discretization for bayesian network structure learning in robot grasping,” in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 1944–1950.

[102] ——, “Embodiment-specific representation of robot grasping using graphical models and latent-space discretization,” in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2011, pp. 980–986.

[103] D. Song, N. Kyriazis, I. Oikonomidis, C. Papazov, A. Argyros, D. Burschka, and D. Kragic, “Predicting human intention in visual observations of hand/object interactions,” in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 1608–1615.

[104] D. Song, C. H. Ek, K. Huebner, and D. Kragic, “Task-based robot grasp planning using probabilistic inference,” *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 546–561, Jun. 2015.

[105] O. Kroemer and J. Peters, “A flexible hybrid framework for modeling complex manipulation tasks,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1856–1861.

[106] O. Kroemer, E. Ugur, E. Oztop, and J. Peters, “A kernel-based approach to direct action perception,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 2605–2610.

[107] H. Dang and P. Allen, “Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct. 2012, pp. 1311–1317.

[108] H. Dang and P. K. Allen, “Semantic grasping: Planning task-specific stable robotic grasps,” *Autonomous Robots*, vol. 37, no. 3, pp. 301–316, Oct. 2014.

[109] S. Belongie, J. Malik, and J. Puzicha, “Shape matching and object recognition using shape contexts,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 4, pp. 509–522, Apr. 2002.

[110] D. Katz, A. Venkatraman, M. Kazemi, J. A. Bagnell, and A. Stentz, “Perceiving, learning, and exploiting object affordances for autonomous pile manipulation,” *Autonomous Robots*, vol. 37, no. 4, pp. 369–382, Dec. 2014.

[111] A. t. Pas and R. Platt, “Localizing handle-like grasp affordances in 3d point clouds,” in *Experimental Robotics: The 14th International Symposium on Experimental Robotics*. Cham: Springer International Publishing, 2016, pp. 623–638.

[112] A. ten Pas and R. Platt, “Using geometry to detect grasp poses in 3d point clouds,” in *Robotics Research*, Springer, 2018, pp. 307–324.

[113] A. Zeng, S. Song, K. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, N. Fazeli, F. Alet, N. C. Dafle, R. Holladay, I. Morena, P. Q. Nair, D. Green, I. Taylor, W. Liu, T. Funkhouser, and A. Rodriguez, “Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1–8.

[114] G. Saponaro, G. Salvi, and A. Bernardino, “Robot anticipation of human intentions through continuous gesture recognition,” in *Collaboration Technologies and Systems (CTS), 2013 International Conference on*, pp. 218–225.

[115] A. Pandey and R. Alami, “Mightability maps: A perceptual level decisional framework for co-operative and competitive human-robot interaction,” in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, Oct. 2010, pp. 5842–5848.

[116] A. K. Pandey and R. Alami, “Affordance graph: A framework to encode perspective taking and effort based affordances for day-to-day human-robot interaction,” in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pp. 2180–2187.

[117] W. Chan, Y. Kakiuchi, K. Okada, and M. Inaba, “Determining proper grasp configurations for handovers through observation of object movement patterns and inter-object interactions during usage,” in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, Sep. 2014, pp. 1355–1360.

[118] T. Shu, X. Gao, M. S. Ryoo, and S. Zhu, “Learning social affordance grammar from videos: Transferring human interactions to human-robot interactions,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 1669–1676.

[119] A. Pieropan, C. H. Ek, and H. Kjellstrom, “Functional object descriptors for human activity modeling,” in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 1282–1289.

[120] A. Pieropan, C. H. Ek, and H. Kjellström, “Recognizing object affordances in terms of spatio-temporal object-object relationships,” in *2014 IEEE-RAS International Conference on Humanoid Robots*, Nov. 2014, pp. 52–58.

[121] G. Stockman and L. G. Shapiro, *Computer Vision*, 1st. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

[122] D. Paulius, Y. Huang, R. Milton, W. D. Buchanan, J. Sam, and Y. Sun, “Functional object-oriented network for manipulation learning,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2016, pp. 2655–2662.

[123] H. S. Koppula and A. Saxena, “Anticipating human activities for reactive robotic response,” in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pp. 2071–2071.

[124] H. S. Koppula, R. Gupta, and A. Saxena, “Learning human activities and object affordances from rgb-d videos,” *The International Journal of Robotics Research*, vol. 32, no. 8, pp. 951–970, 2013.

[125] H. S. Koppula and A. Saxena, “Physically grounded spatio-temporal object affordances,” in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., ser. Lecture Notes in Computer Science, vol. 8691, Heidelberg: Springer, 2014, pp. 831–847.

[126] Y. Jiang and A. Saxena, “Modeling high-dimensional humans for activity anticipation using gaussian process latent crfs.,” in *Robotics: Science and Systems*, 2014, pp. 1–8.

[127] H. S. Koppula and A. Saxena, “Anticipating human activities using object affordances for reactive robotic response,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, pp. 14–29, 2016.

[128] A. Aldoma, F. Tombari, and M. Vincze, “Supervised learning of hidden and non-hidden 0-order affordances and detection in real scenes,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 1732–1739.

[129] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, “The princeton shape benchmark,” in *Proceedings Shape Modeling Applications, 2004.*, IEEE, 2004, pp. 167–178.

[130] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, “Fast 3d recognition and pose using the viewpoint feature histogram,” in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2010, pp. 2155–2162.

[131] A. E. Johnson and M. Hebert, “Using spin images for efficient object recognition in cluttered 3d scenes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 5, pp. 433–449, May 1999.

[132] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in *Computer Vision – ECCV 2010*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 356–369.

[133] D. Kim and G. Sukhatme, "Semantic labeling of 3d point clouds with object affordance for robot manipulation," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2014, pp. 5578–5584.

[134] L. Hinkle and E. Olson, "Predicting object functionality using physical simulations," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pp. 2784–2790.

[135] L.-F. Yu, N. Duncan, and S.-K. Yeung, "Fill and transfer: A simple physics-based approach for containability reasoning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 711–719.

[136] A. Myers, C. L. Teo, C. Fermüller, and Y. Aloimonos, "Affordance detection of tool parts from geometric features," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 1374–1381.

[137] P. Abelha, F. Guerin, and M. Schoeler, "A model-based approach to finding substitute tools in 3d vision data," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, IEEE, 2016, pp. 2471–2478.

[138] P. Abelha and F. Guerin, "Learning how a tool affords by simulating 3d models from the web," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 4923–4929.

[139] S. Rezapour Lakani and J. Rodríguez-Sánchez Antonio J.and Piater, "Towards affordance detection for robot manipulation using affordance for parts and parts for affordance," *Autonomous Robots*, Jul. 2018.

[140] A. Nguyen, D. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis, "Detecting object affordances with convolutional neural networks," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, IEEE, 2016, pp. 2765–2770.

[141] L. Porzi, S. R. Buló, A. Penate-Sanchez, E. Ricci, and F. Moreno-Noguer, "Learning depth-aware deep representations for robotic perception," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 468–475, Apr. 2017.

[142] J. Sawatzky, A. Srikantha, and J. Gall, “Weakly supervised affordance detection,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, pp. 5197–5206.

[143] A. Nguyen, D. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis, “Object-based affordances detection with convolutional neural networks and dense conditional random fields,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 5908–5915.

[144] T. Do, A. Nguyen, and I. Reid, “Affordancenet: An end-to-end deep learning approach for object affordance detection,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1–5.

[145] C. Ye, Y. Yang, R. Mao, C. Fermüller, and Y. Aloimonos, “What can i do around here? deep functional scene understanding for cognitive robots,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 4604–4611.

[146] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, “Sun database: Large-scale scene recognition from abbey to zoo,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2010, pp. 3485–3492.

[147] C. Desai and D. Ramanan, “Predicting functional regions on objects,” in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pp. 968–975.

[148] 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.

[149] Y. Zhu, Y. Zhao, and S. Zhu, “Understanding tools: Task-oriented object modeling, learning and recognition,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 2855–2864.

[150] A. Srikantha and J. Gall, “Discovering object classes from activities,” in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., ser. Lecture Notes in Computer Science. Springer International Publishing, 2014, vol. 8694, ch. 27, pp. 415–430.

[151] S. Thermos, G. T. Papadopoulos, P. Daras, and G. Potamianos, “Deep affordance-grounded sensorimotor object recognition,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017.

[152] K. Fang, T. Wu, D. Yang, S. Savarese, and J. J. Lim, “Demo2vec: Reasoning object affordances from online videos,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2018, pp. 2139–2147.

[153] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, “Structural-rnn: Deep learning on spatio-temporal graphs,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016.

[154] H. Grabner, J. Gall, and L. Van Gool, “What makes a chair a chair?” In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 1529–1536.

[155] A. Gupta, S. Satkin, A. A. Efros, and M. Hebert, “From 3d scene geometry to human workspace,” in *CVPR 2011*, Jun. 2011, pp. 1961–1968.

[156] V. Delaitre, D. F. Fouhey, I. Laptev, J. Sivic, A. Gupta, and A. A. Efros, “Scene semantics from long-term observation of people,” in *Computer Vision – ECCV 2012*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 284–298.

[157] D. F. Fouhey, V. Delaitre, A. Gupta, A. A. Efros, I. Laptev, and J. Sivic, “People watching: Human actions as a cue for single view geometry,” in *Computer Vision – ECCV 2012*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 732–745.

[158] Y. Jiang, M. Lim, and A. Saxena, “Learning object arrangements in 3d scenes using human context,” in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, Omnipress, 2012, pp. 907–914.

[159] Y. Jiang and A. Saxena, “Hallucinating humans for learning robotic placement of objects,” in *Experimental Robotics*, Springer, 2013, pp. 921–937.

[160] Y. Jiang, H. Koppula, and A. Saxena, *Hallucinated humans as the hidden context for labeling 3d scenes*, Conference Paper, 2013.

[161] Y. Zhu, C. Jiang, Y. Zhao, D. Terzopoulos, and S. Zhu, “Inferring forces and learning human utilities from videos,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 3823–3833.

[162] X. Wang, R. Girdhar, and A. Gupta, “Binge watching: Scaling affordance learning from sitcoms,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017.

[163] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.

[164] C. Chuang, J. Li, A. Torralba, and S. Fidler, “Learning to act properly: Predicting and explaining affordances from images,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2018, pp. 975–983.

[165] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, “Gated graph sequence neural networks,” *arXiv preprint arXiv:1511.05493*, 2015.

[166] L. Piyathilaka and S. Kodagoda, “Active visual object search using affordance-map in real world: A human-centric approach,” in *2014 13th International Conference on Control Automation Robotics Vision (ICARCV)*, Dec. 2014, pp. 1427–1432.

[167] ——, “Affordance-map: Mapping human context in 3d scenes using cost-sensitive svm and virtual human models,” in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec. 2015, pp. 2035–2040.

[168] Y. Zhao and S. Zhu, “Scene parsing by integrating function, geometry and appearance models,” in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2013, pp. 3119–3126.

[169] S. Qi, S. Huang, P. Wei, and S. Zhu, “Predicting human activities using stochastic grammar,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017, pp. 1173–1181.

[170] S. Qi, Y. Zhu, S. Huang, C. Jiang, and S. Zhu, “Human-centric indoor scene synthesis using stochastic grammar,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2018, pp. 5899–5908.

[171] A. Roy and S. Todorovic, “A multi-scale cnn for affordance segmentation in rgb images,” in *European Conference on Computer Vision*, Springer, 2016, pp. 186–201.

[172] P. K. Nathan Silberman Derek Hoiem and R. Fergus, “Indoor segmentation and support inference from rgbd images,” in *ECCV*, 2012.

[173] Y. Zhu, A. Fathi, and L. Fei-Fei, “Reasoning about object affordances in a knowledge base representation,” in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., ser. Lecture Notes in Computer Science. Springer International Publishing, 2014, vol. 8690, ch. 27, pp. 408–424.

[174] Y. W. Chao, Z. Wang, R. Mihalcea, and J. Deng, “Mining semantic affordances of visual object categories,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 4259–4267.

[175] G. Patterson and J. Hays, “Sun attribute database: Discovering, annotating, and recognizing scene attributes,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2012, pp. 2751–2758.

[176] N. Mitra, M. Wand, H. (Zhang, D. Cohen-Or, V. Kim, and Q.-X. Huang, “Structure-aware shape processing,” in *SIGGRAPH Asia 2013 Courses*, ser. SA ’13, Hong Kong, Hong Kong: ACM, 2013, 1:1–1:20.

[177] K. Xu, V. G. Kim, Q. Huang, and E. Kalogerakis, “Data-driven shape analysis and processing,” *Computer Graphics Forum*, vol. 36, no. 1, pp. 101–132, 2017. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12790`.

[178] R. Hu, M. Savva, and O. van Kaick, “Functionality representations and applications for shape analysis,” in *Computer Graphics Forum*, Wiley Online Library, vol. 37, 2018, pp. 603–624.

[179] N. Gelfand and L. J. Guibas, “Shape segmentation using local slippage analysis,” in *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, ACM, 2004, pp. 214–223.

[180] M. E. Yumer and L. B. Kara, “Co-constrained handles for deformation in shape collections,” *ACM Trans. Graph.*, vol. 33, no. 6, 187:1–187:11, Nov. 2014.

[181] H. Laga, M. Mortara, and M. Spagnuolo, “Geometry and context for semantic correspondences and functionality recognition in man-made 3d shapes,” *ACM Trans. Graph.*, vol. 32, no. 5, 150:1–150:16, Oct. 2013.

[182] P. Merrell, E. Schkufza, Z. Li, M. Agrawala, and V. Koltun, “Interactive furniture layout using interior design guidelines,” in *ACM transactions on graphics (TOG)*, ACM, vol. 30, 2011, p. 87.

[183] K. Xu, J. Stewart, and E. Fiume, “Constraint-based automatic placement for scene composition,” in *Proceedings - Graphics Interface*, Jun. 2002, pp. 25–34.

[184] M. Fisher, D. Ritchie, M. Savva, T. Funkhouser, and P. Hanrahan, “Example-based synthesis of 3d object arrangements,” *ACM Transactions on Graphics (TOG)*, vol. 31, no. 6, p. 135, 2012.

[185] M. Fisher, M. Savva, and P. Hanrahan, “Characterizing structural relationships in scenes using graph kernels,” *ACM transactions on graphics (TOG)*, vol. 30, no. 4, p. 34, 2011.

[186] X. Zhao, H. Wang, and T. Komura, “Indexing 3d scenes using the interaction bisector surface,” *ACM Trans. Graph.*, vol. 33, pp. 1–14, 2014.

[187] W. Massey, *A Basic Course in Algebraic Topology*, ser. Graduate Texts in Mathematics. Springer New York, 1991.

[188] R. Hu, C. Zhu, O. van Kaick, L. Liu, A. Shamir, and H. Zhang, “Interaction context (icon): Towards a geometric functionality descriptor,” *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, p. 83, 2015.

[189] X. Zhao, R. Hu, P. Guerrero, N. J. Mitra, and T. Komura, “Relationship templates for creating scene variations,” *ACM Trans. Graph.*, vol. 35, no. 6, 207:1–207:13, 2016.

[190] H. J. Wolfson and I. Rigoutsos, “Geometric hashing: An overview,” *IEEE computational science and engineering*, vol. 4, no. 4, pp. 10–21, 1997.

[191] S. Pirk, V. Krs, K. Hu, S. D. Rajasekaran, H. Kang, Y. Yoshiyasu, B. Benes, and L. J. Guibas, “Understanding and exploiting object interaction landscapes,” *ACM Trans. Graph.*, vol. 36, no. 4, Jun. 2017.

[192] M. Savva, A. X. Chang, P. Hanrahan, M. Fisher, and M. Niessner, “Scenegrok: Inferring action maps in 3d environments,” *ACM Trans. Graph.*, vol. 33, no. 6, 212:1–212:10, 2014.

[193] V. G. Kim, S. Chaudhuri, L. Guibas, and T. Funkhouser, “Shape2pose: Human-centric shape analysis,” *ACM Trans. Graph.*, vol. 33, no. 4, 120:1–120:12, Jul. 2014.

[194] M. Savva, A. X. Chang, P. Hanrahan, M. Fisher, and M. Nießner, “PiGraphs: Learning Interaction Snapshots from Observations,” *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, 2016.

[195] X. Zhao, M. G. Choi, and T. Komura, “Character-object interaction retrieval using the interaction bisector surface,” in *Computer Graphics Forum*, Wiley Online Library, vol. 36, 2017, pp. 119–129.

[196] K. Q. Brown, “Voronoi diagrams from convex hulls,” *Information Processing Letters*, vol. 9, no. 5, pp. 223–228, 1979.

[197] C. B. Barber, D. P. Dobkin, D. P. Dobkin, and H. Huhdanpaa, “The quickhull algorithm for convex hulls,” *ACM Transactions on Mathematical Software (TOMS)*, vol. 22, no. 4, pp. 469–483, 1996.

[198] F. P. Preparata and M. I. Shamos, *Computational geometry: an introduction*. Springer Science & Business Media, 2012.

[199] S. Suri. (Feb. 2019). Linking voronoi diagram, delaunay triangulation and convex hulls through the lifting transform, [Online]. Available: `https://www.cs.ucsb.edu/~suri/cs235/Dual.pdf`.

[200] S. Tsogkas and S. Dickinson, “Amat: Medial axis transform for natural images,” in *Computer Vision (ICCV), 2017 IEEE International Conference on*, IEEE, 2017, pp. 2727–2736.

[201] C. Funk, S. Lee, M. R. Oswald, S. Tsogkas, W. Shen, A. Cohen, S. Dickinson, and Y. Liu, “2017 iccv challenge: Detecting symmetry in the wild,” in *Computer Vision Workshop (ICCVW), 2017 IEEE International Conference on*, IEEE, 2017, pp. 1692–1701.

[202] S. Stolpner, P. Kry, and K. Siddiqi, “Medial spheres for shape approximation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 6, pp. 1234–1240, Jun. 2012.

[203] R. Marie, O. Labbani-Igbida, and E. M. Mouaddib, “The delta medial axis: A fast and robust algorithm for filtered skeleton extraction,” *Pattern Recognition*, vol. 56, pp. 26–39, 2016.

[204] J. Denny, E. Greco, S. Thomas, and N. M. Amato, “Marrt: Medial axis biased rapidly-exploring random trees,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 90–97.

[205] G. Liu and J.-M. Lien, "Fast medial-axis approximation via max-margin pushing," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, Citeseer, 2015, pp. 3262–3267.

[206] F. Sun, Y. Choi, Y. Yu, and W. Wang, "Medial meshes – a compact and accurate representation of medial axis transform," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 3, pp. 1278–1290, Mar. 2016.

[207] M. Peternell, "Geometric properties of bisector surfaces," *Graphical Models*, vol. 62, no. 3, pp. 202–236, 2000.

[208] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[209] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, Feb. 1992.

[210] S. Li and A. Calway, "Rgbd relocalisation using pairwise geometry and concise key point sets," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 6374–6379.

[211] Q.-Y. Zhou and V. Koltun, "Dense scene reconstruction with points of interest," *ACM Trans. Graph.*, vol. 32, no. 4, 112:1–112:8, Jul. 2013.

[212] S. Choi, Q.-Y. Zhou, S. Miller, and V. Koltun, "A large dataset of object scans," *arXiv:1602.02481*, 2016.

[213] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. Dollar, "Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set," *Robotics Automation Magazine, IEEE*, vol. 22, no. 3, pp. 36–52, Sep. 2015.

[214] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with cuda," *Queue*, vol. 6, no. 2, pp. 40–53, Mar. 2008.

[215] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 1–4.

[216] R. A. Bradley and M. E. Terry, "Rank analysis of incomplete block designs: I. the method of paired comparisons," *Biometrika*, vol. 39, no. 3/4, pp. 324–345, 1952.

[217] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[218] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in *Intelligent Robots International Conference on (IROS)*, 2015.

[219] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Computer Vision and Pattern Recognition*, 2015.

[220] D. Z. Wang and I. Posner, "Voting for voting in online point cloud object detection.," in *Robotics: Science and Systems*, vol. 1, 2015.

[221] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view cnns for object classification on 3d data," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5648–5656.

[222] Y. Li, S. Pirk, H. Su, C. R. Qi, and L. J. Guibas, "Fpnn: Field probing neural networks for 3d data," in *Advances in Neural Information Processing Systems*, 2016, pp. 307–315.

[223] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "Generative and discriminative voxel modeling with convolutional neural networks," *CoRR*, vol. abs/1608.04236, 2016. arXiv: 1608.04236.

[224] R. Klokov and V. S. Lempitsky, "Escape from cells: Deep kd-networks for the recognition of 3d point cloud models," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 863–872.

[225] G. Riegler, A. O. Ulusoy, and A. Geiger, "Octnet: Learning deep 3d representations at high resolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[226] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *arXiv preprint arXiv:1612.00593*, 2016.

[227] L. Yi, H. Su, X. Guo, and L. J. Guibas, "Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation.," in *CVPR*, 2017, pp. 6584–6592.

[228] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *arXiv preprint arXiv:1706.02413*, 2017.

[229] T. Le and Y. Duan, “Pointgrid: A deep network for 3d shape understanding,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018.

[230] Q. Morgan, C. Ken, G. Brian P., F. Josh, F. Tully, L. Jeremy, W. Rob, and N. Andrew Y., “Ros: An open-source robot operating system,” in *IEEE International Conference on Robotics and Automation Workshops*, 2009.

[231] R. T. Azuma, “A survey of augmented reality,” *Presence: Teleoperators & Virtual Environments*, vol. 6, no. 4, pp. 355–385, 1997.

[232] F. Zhou, H. B.-L. Duh, and M. Billinghurst, “Trends in augmented reality tracking, interaction and display: A review of ten years of ismar,” in *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, ser. ISMAR ’08, Washington, DC, USA: IEEE Computer Society, 2008, pp. 193–202.

[233] K. Kim, M. Billinghurst, G. Bruder, H. B.-L. Duh, and G. F. Welch, “Revisiting trends in augmented reality research: A review of the 2nd decade of ismar (2008–2017),” *IEEE transactions on visualization and computer graphics*, vol. 24, no. 11, pp. 2947–2962, 2018.

[234] S. J. Henderson and S. Feiner, “Opportunistic controls: Leveraging natural affordances as tangible user interfaces for augmented reality,” in *Proceedings of the 2008 ACM Symposium on Virtual Reality Software and Technology*, ser. VRST ’08, Bordeaux, France: ACM, 2008, pp. 211–218.

[235] M. Geuss, J. Stefanucci, S. Creem-Regehr, and W. B. Thompson, “Can i pass?: Using affordances to measure perceived size in virtual environments,” in *Proceedings of the 7th Symposium on Applied Perception in Graphics and Visualization*, ser. APGV ’10, Los Angeles, California: ACM, 2010, pp. 61–64.

[236] G. Pointon, C. Thompson, S. Creem-Regehr, J. Stefanucci, M. Joshi, R. Paris, and B. Bodenheimer, “Judging action capabilities in augmented reality,” in *Proceedings of the 15th ACM Symposium on Applied Perception*, ser. SAP ’18, Vancouver, British Columbia, Canada: ACM, 2018, 6:1–6:8.

[237] G. Pointon, C. Thompson, S. Creem-Regehr, J. Stefanucci, and B. Bodenheimer, “Affordances as a measure of perceptual fidelity in augmented reality,” in *2018 IEEE VR 2018 Workshop on Perceptual and Cognitive Issues in AR (PERCAR)*, 2018, pp. 1–6.

[238] S.-w. Leigh and P. Maes, “Aftermath: Visualizing consequences of actions through augmented reality,” in *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA ’15, Seoul, Republic of Korea: ACM, 2015, pp. 941–946.

[239] S. Lin, H. F. Cheng, W. Li, Z. Huang, P. Hui, and C. Peylo, “Ubii: Physical world interaction through augmented reality.,” *IEEE Trans. Mob. Comput.*, vol. 16, no. 3, pp. 872–885, 2017.

[240] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, “Slam++: Simultaneous localisation and mapping at the level of objects,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 1352–1359.

[241] E. Marchand, H. Uchiyama, and F. Spindler, “Pose estimation for augmented reality: A hands-on survey,” *IEEE transactions on visualization and computer graphics*, vol. 22, no. 12, pp. 2633–2651, 2016.

[242] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, “Kinectfusion: Real-time dense surface mapping and tracking,” in *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, IEEE, 2011, pp. 127–136.

[243] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, “OctoMap: An efficient probabilistic 3D mapping framework based on octrees,” *Autonomous Robots*, 2013.

[244] R. Bajcsy, Y. Aloimonos, and J. K. Tsotsos, “Revisiting active perception,” *Autonomous Robots*, vol. 42, no. 2, pp. 177–196, 2018.

[245] M. Pizzoli, C. Forster, and D. Scaramuzza, “Remode: Probabilistic, monocular dense reconstruction in real time,” in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, IEEE, 2014, pp. 2609–2616.

[246] J. Li, R. Klein, and A. Yao, “A two-streamed network for estimating fine-scaled depth maps from single rgb images,” in *Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy*, 2017, pp. 22–29.

[247] F. Ma and S. Karaman, “Sparse-to-dense: Depth prediction from sparse depth samples and a single image,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1–8.

[248] M. Lapin, M. Hein, and B. Schiele, “Analysis and optimization of loss functions for multiclass, top-k, and multilabel classification,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 7, pp. 1533–1554, 2018.