Towards Robot-Centric Conceptual Knowledge Acquisition

Georg Jäger[1*], Christian A. Mueller[2*], Madhura Thosar[1*], Sebastian Zug[1] and Andreas Birk[2]

***Abstract*— Robots require knowledge about objects in order to efficiently perform various household tasks involving objects. The existing knowledge bases for robots acquire symbolic knowledge about objects from manually-coded external common sense knowledge bases such as ConceptNet, WordNet etc. The problem with such approaches is the discrepancy between human-centric symbolic knowledge and robot-centric object perception due to its limited perception capabilities. Ultimately, significant portion of knowledge in the knowledge base remains ungrounded into robot's perception. To overcome this discrepancy, we propose an approach to enable robots to generate robot-centric symbolic knowledge about objects from their own sensory data, thus, allowing them to assemble their own conceptual understanding of objects. With this goal in mind, the presented paper elaborates on the work-in-progress of the proposed approach followed by the preliminary results.**

I. Motivation

Baber [1] postulated that a deliberation for tool selection in humans or animals alike is facilitated by conceptual knowledge about objects, especially, knowledge about their physical and functional properties and relationship between them. The conceptual knowledge about household objects is desired in a service robot too when performing various household tasks, for instance, selecting an appropriate tool for a given task, selecting a substitute for a missing tool required in some task or action selection for using objects. Since the demand for such conceptual knowledge about objects has been increasing, the development of such knowledge bases has been undertaken by the researchers around the world. In [2], we reviewed the following existing knowledge bases developed for service robotics: KNOWROB [3], MLN-KB [4], NMKB [5], OMICS [6], OMRKF [7], ORO [8], OUR-K [9], PEIS-KB [10], and RoboBrain [11]; with respect to various criteria corresponding to the categories *knowledge acquisition*, *knowledge representation*, and *knowledge grounding*.

Knowledge grounding, a.k.a. symbol grounding or perceptual anchoring, which describes a mapping from abstract symbols to representative sensory data, is of special interest for robots. It closes the gap between symbolic reasoning, which enables abstract decision making, and interpreting their sensory data, which is imperative for understanding a robot's environment. In many existing knowledge bases

[1]Georg Jäger, Madhura Thosar and Sebastian Zug are with Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Germany gjaeger@ovgu.de, thosar@iks.cs.ovgu.de, zug@ivs.cs.uni-magdeburg.de

[2]Christian A. Mueller and Andreas Birk are with the Robotics Group, Computer Science & Electrical Engineering Department, Jacobs University Bremen, Germany, {chr.mueller,a.birk}@jacobs-university.de

[*]These authors contributed equally to this work and share first authorship.

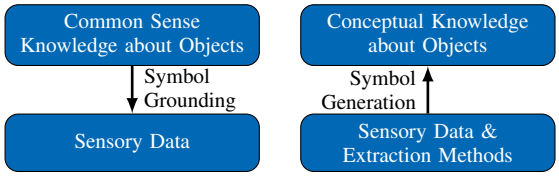


Fig. 1. Symbol grounding approach comparison: the typical approach to grounding knowledge vs. proposed approach to grounding the knowledge

(see [2] for details), symbols are defined by humans and are derived from common sense knowledge bases such as WordNet (KnowRob, MLN-KB, OMICS, RoboBrain), Cyc (PEIS-KB) OpenCyc (KnowRob, ORO, RoboBrain) (see left side of Fig. 1). By employing sensors, a mapping from these symbols to sensory data is constructed. However, this imposes a human perspective on a robot's sensory data. Humans select sensors and dictate their interpretation for grounding the symbols. This can only work with unambiguous sensory data and complete knowledge about all relevant object categories.

In contrast, we propose a bottom-up approach to symbol grounding, a robot-centric symbol generation approach, see right side of Fig. 1. Our current research work is aimed at creating a multi-layered dataset that can be used to build robot-centric conceptual knowledge about household objects. Instead of predefining the symbols and grounding them afterwards, we use sensory data and robot-centric extraction methods to generate qualitative data for each object property. Afterwards, symbols are generated from this qualitative data in an unsupervised manner and thereby inherently grounded in the robots sensory data. Moreover, this approach enables a robot to build up its own individualized understanding about objects that does not rely on completeness.

The conceptual knowledge considered in this work primarily involves properties of the objects. The properties considered are divided into *physical* and *functional* properties where physical properties describe the physicality of the objects such as *rigidity*, *weight*, *hollowness* while the functional properties ascribe the (functional) abilities or affordances to the objects such as *containment*, *blockage*, *movability*.

The remainder of this paper is structured as follows: we initially define the considered physical and functional properties and introduce the property acquisition in Sec. II. Using the presented definitions and acquisition methods, we further elaborate on the architecture of our framework for the generation of robot-centric knowledge in Sec. II-C. In Sec. IV we present our preliminary results. Finally, we conclude our work and discuss possible future work (Sec. V).

II. Property Acquisition

As a prerequisite to the generation of a robot-centric knowledge base, we present definitions of the considered object properties in Sec. II-A. These are general and not specialized towards a robotic platform as they represent the humans' perspective. In contrast, in Sec. II-B we define methods for acquiring scalar representations of the defined object properties for a robotic platform. Their implementations are embedded into an extensible framework for data aggregation, which we will briefly introduce in Sec. II-C.

A. Property Definition

Overall, we consider ten core object properties. We start with six physical properties. These form the basis from which the remaining four functional properties, which we will define afterwards, emerge from.

1) Physical Properties: As a selection of core physical properties linked to the physicality of an object we have considered ***size***, ***hollowness***, ***flatness***, ***rigidity***, ***roughness***, and ***weight***. This selection is inspired from the discussion on the design of tools offered by Baber in [12] where it is stated that, among others, the properties such as ***shape***, ***size***, ***rigidity***, ***roughness***, and ***weight*** play a significant role in the design of a tool. In the following, we discuss briefly the property definitions which state how they are to be measured.

Size of an object is described by its spatial dimensionality in form of *length*, *width* and *height*. ***Flatness***, on the contrary, describes a particular aspect of an object's shape. ***Flatness*** is defined as the ratio between the area of an object's greatest plane and its overall surface area. For instance, a sheet of paper has maximal ***flatness*** while a ball has minimal ***flatness***. ***Hollowness*** focuses on another aspect of an object's shape. It is the amount of visible cavity or empty space within an object's enclosed volume. ***Weight*** of an object is borrowed from physics: the object's ***weight*** is the force acting on its mass within a gravitational field. Similar to gravity, forces acting on objects might deform it depending on its ***rigidity***. Consequently, we define ***rigidity*** as the degree of deformation caused by a force operating vertically on an object. The last physical property to be defined is ***roughness***. It provides a feedback about the object's surface. Therefore, we simplify the physical idea of friction and define ***roughness*** as an object's resistance to slide.

2) Functional Properties: Opposed to physical properties, functional properties describe the functional capabilities or affordances of objects. It is proposed in [13] that functional properties do not exist in isolation, rather certain physical properties are required to enable them. The proposed system follows the same suit where each functional property is defined in terms of the associated physical properties

A basic functional property is ***support***. It describes an object's capability to ***support***, i.e. to carry another object. Therefore, an object is attributed with ***support***, if other objects can be stably placed on top of the supporting object. The ***containment*** property extends this idea. An object is attributed with property ***containment*** if it can encompass another object to a certain degree. Finally, we also consider

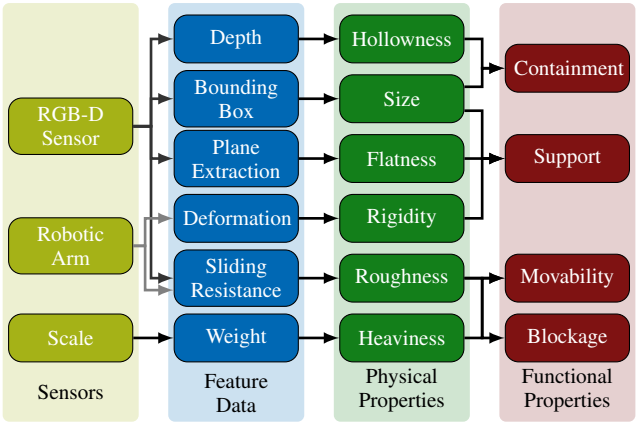


Fig. 2. Proposed property hierarchy and their dependencies.

movability, which describes the required effort to move an object, and ***blockage***, which describes an object's capability to stop the movement of another object.

B. Property Extraction

Though the property definitions in Sec. II-A are formulated from a human perspective, we aim at enabling a robot to assemble its own understanding about objects. Hence, we have devised the extraction methods allowing a robot to interpret its sensory data for generating scalar representations of object properties. The different levels of abstractions, starting with the sensory data and ending with functional properties, are shown in Fig. 2. This, however, requires to take the available sensors and actuators into account to ensure observability of all properties. While this means that the presented methods are tuned towards our robotic platform (a Kuka youBot [14] (see Fig 4(d)) and a Asus Xtion Pro depth sensor [15] (see Fig. 3(a))), they are adoptable to other robotic platforms as we use common hardware.

Across all methods, we assume that the object for which the property shall be measured, is placed in its most natural position. For instance, a cup is most commonly placed in such a way, that its opening points upwards.

1) Physical Properties: The ***size*** of an object is extracted from point clouds of an RGB-D sensor by segmenting an object from the scene which then is used to estimate the bounding box. The length, width, and height of the bounding box are then used to measure the ***size***.

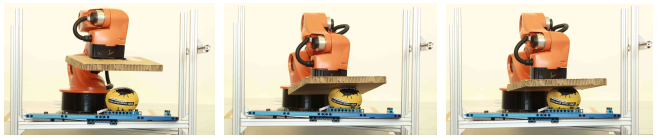
Additionally, the segmented object point cloud enables the robot to extract the object's ***flatness*** value. By observing an object from above, an object's greatest top-level plane is extracted using RANSAC (RAndom SAmple Consensus). The size of this plane, that is, the number of points corresponding to this plane is divided by the number of points representing the observed object to get a scalar measure of its ***flatness***.

Similar to ***flatness***, ***hollowness*** is also focused on an object's shape. According to its definition, an object's enclosed, but not filled volume defines its hollowness. For the sake of simplicity, we measure the internal depth and height of an

(a) Multi-camera setup (b) Side RGB-D camera

(c) Top RGB-D camera (d) Top RGB-D camera

Fig. 3. Experimental setup consisting of a two camera combination, for acquiring physical properties such as ***hollowness***, ***size***, ***roughness***.



(a) Rigidity extraction at time t=1 (b) Rigidity extraction at time t=2 (c) Rigidity extraction at time t=3

(d) Roughness extraction at time t=1 (e) Roughness extraction at time t=2 (f) Roughness extract. at time t=3

Fig. 4. Experimental setup consisting of a camera-manipulator combination, for acquiring physical properties such as ***rigidity*** and ***roughness*** which are required to acquire functional properties such as ***support*** and ***movability***.

object, which resembles the enclosed volume, and use their ratio as ***hollowness*** value. To measure depth and height, we employ marker detection, which is robust and insusceptible to noise. We place one marker inside (or on top) of the considered object and another marker right next to the object which functions as a global spatial reference (see samples in Fig. 3(b)- 3(d)).

In contrast, measuring the ***weight*** of an object is straight forward. Using a scale with a resolution of 1g, the actual ***weight*** of each object can be measured directly. While this requires additional hardware, a robot could easily try to lift an object and calculate its ***weight*** by converting the efforts observed during the process in each of its joints (we are currently working on replacing the method).

The ***rigidity*** property requires a more sophisticated measuring process. Following its definition, we use a robotic arm with a planar end-effector to vertically exert a force onto an object until predefined efforts in the arm's joints are exceeded, see Fig. 4(a)-(c). Using the joints' positions at the first contact with the object and when the efforts are exceeded, the vertical movement of the arm during the experiment is calculated. For rigid objects, this distance is zero while it is increased continuously for non-rigid objects.

Similar to ***rigidity***, ***roughness*** requires interaction to measure an object's resistance to sliding. Therefore, the robotic arm is used to act as a ramp on which the considered object is placed, see Fig. 4(d)-(f). Starting horizontally, with an angle of 0°, the ramp's angle is increased and thereby causes the force pulling the object down the ramp to be increased too. As soon as the object starts sliding, which is detected based on marker detection, the movement is stopped. In this position, the ramp's angle is a measure of the object's ***roughness***.

2) Functional Properties: As functional properties are enabled by an object's physical attributes, we define their extraction methods on the basis of an object's physical properties. Corresponding to its definition, ***support*** requires to consider three aspects of an object. Firstly, the considered object needs to be rigid. Secondly, for carrying another object, the sizes of both need to match. Thirdly, the object's shape needs to be sufficiently flat in order to enable the placing of another object on top of it. Consequently, for measuring support, we consider ***rigidity***, ***size***, and ***flatness***.

Similarly, the ***containment*** property requires to consider two aspects. In order to contain something, an object needs to be hollow. On the other hand, it's ***size*** itself needs to be respected when considering whether it can contain another object. Thus, the value of an object's ***containment*** property is formed by combining its ***hollowness*** and ***size***.

Extracting an object's ***movability*** is based on a robot's primary ways of moving objects: either by lifting or pushing. For both, an object's ***weight*** is important. Additionally, when pushing an object, its sliding resistance, that is, its ***roughness*** (see Fig. 4), needs to be considered.

Finally, assessing an object's ***blockage***, can be derived from its ***movability***. According to its definition, ***blockage*** states to which degree an object is able to stop another object's movement. Thus, the object itself needs to be not movable by the other object, which is the inverse of its ***movability***.

The described hierarchy of object properties as well as their dependency on feature and sensory data is illustrated in Fig. 2.

C. Framework for Dataset Acquisition

Using the extraction methods of Sec. II-B, we present our ROS-based framework with which our robotic platform gathers data about objects to build up its individualized knowledge. A schematic overview on the framework is given by Fig. 5.

Although different software platforms for operating robots exist (e.g. Fawkes [16] or Orocos [17]), ROS (Robot Operating System) [18] became a quasi-standard. Given the amount

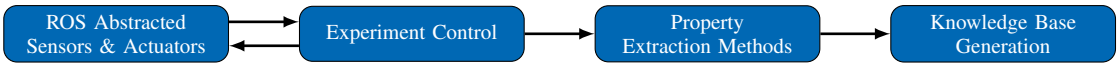


Fig. 5. Data flow within the data set creation framework.

TABLE I

HUMAN-PREDEFINED OBJECT CLASSES, NUMBER OF INSTANCES IN EACH CLASS AND THEIR NUMERIC LABELS USED IN THE PLOTS IN FIG. 6 AND 7

Class	Plastic Box	Paper Plate	Steel Cup	Ceramic Bowl	Plastic Cup	Paper Box	Ceramic Plate	Ball	Metal Box	Paper Cup	Marker	Plastic Bowl	Ceramic Cup	Sponge	Marble Plank	Ceramic Glass	Book	$\sum$
Label	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
#	1	1	1	3	3	9	6	1	3	1	3	3	3	2	1	1	4	46

of supported hardware components, building on top of this middleware enables other researchers to reproduce our results and adapt our framework according to their specific hardware (which is essential for building an robot-centric knowledge base).

Consequently, the interface for operating sensors and actuators is provided to our framework by ROS. This interface is used by different experiments for observing and interacting with objects to acquire the necessary sensory data. Together both blocks (*ROS Abstracted Sensors & Actuators* and *Experiment Control*) form a control loop which generates feature data (see Fig. 2).

To provide extensibility and comparability along with our framework, versatile experiments can be defined by either adding independent ROS nodes or extending existing experiments. The generated feature data is further processed by property extraction methods to calculate the values of each property for the object currently under consideration. Again, extendability and comparability are facilitated by running each extraction method as an independent ROS node and therefore by providing a *plug-and-play* interface.

The data generated by the extraction methods resembles the scalar representations of an objects properties. Therefore, physical as well as functional properties of objects are available and used by the knowledge base generation process to generate a symbolic representation.

III. KNOWLEDGE BASE

Using the framework described in the previous section, we can employ our robotic platform to gather scalar data about an object's properties. However, this data can not be used for symbolic reasoning yet. To facilitate this application, a knowledge base needs to be generated. We briefly describe this step in this section.

The knowledge base primarily consists of two layers: knowledge about object instances and knowledge about object classes. The primary input to the knowledge base is the data about the physical properties of the objects where each object instance is represented in terms of its physical properties as well as its functional properties. The data about the properties of the objects is processed by the knowledge base module in two stages: *sub-categorization* and *conceptualization*. In the sub-categorization process, the non-symbolic continuous data of each property is transformed into symbolic data using a clustering algorithm such as K-means. The cluster representation of the numerical values of the property data can also be seen as a symbolic qualitative measure representing each cluster. Consequently, the number of clusters describes the granularity with which each property can qualitatively be represented. In case of a high number of clusters, an object is described in finer detail. Complementary, a lower number of clusters suggest a general description of an object. For instance, the numerical data about the ***rigidity*** of the object instances of *Ceramic Cup*, when clustered into three clusters, can be represented as Rigidity=$\{soft, medium, rigid\}$ (see Table II). At the end of the *sub-categorization* process, each object is represented in terms of the qualitative measures for each property.

TABLE II

THE SUB-CATEGORIZATION PROCESS WHICH GENERATES THE SYMBOLIC KNOWLEDGE ABOUT OBJECT INSTANCES.

Object		Sensory Data	Discretized Data
Class	Instance	Rigidity	Rigidity
Ceramic Cup	ceramic_cup_1	0.76	*soft*
	ceramic_cup_2	3.17	*medium*
	ceramic_cup_3	7.69	*rigid*

The conceptualization process gathers the knowledge about all the instances of an object class and represents the knowledge about an object class. Initially, the knowledge about objects is represented using *bivariate joint frequency distribution* of the qualitative measures of the properties in the object instances. Next, conceptual knowledge about objects is calculated as a sample proportion of the frequency of the properties across the instances of a class.

The conceptual knowledge about instances and object classes is represented in JSON format in order to allow the users of the knowledge base to adapt the suitable representation formalism for their application.

IV. PRELIMINARY RESULTS

In the endeavor of enabling a robot-centric conceptual knowledge acquisition, we introduced physical and functional properties of objects in the previous section and presented a framework implementing the envisioned process. Since this is a work in progress, we subsequently present preliminary results of the proposed dataset. For the initial experiments, we primarily focused on the evaluation of the discrimination of the object instances with regard to the acquired properties. Given the acquired properties of each instance, we represented each instance in vector form, i.e.

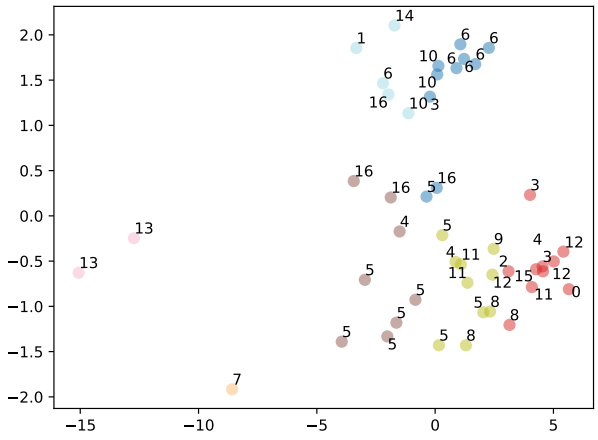


Fig. 6. The instances are represented by the physical properties (reduced to two dimensional space [19]) and are clustered using K-means. Note: instances are colored w.r.t. to cluster assignment; cluster colors are randomly selected; instances are annotated with a numeric label according to Table I.

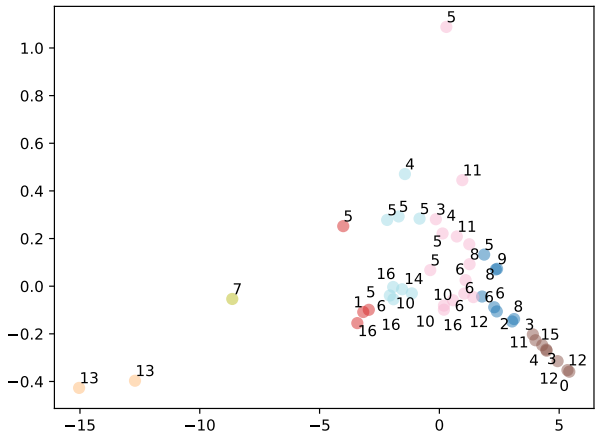


Fig. 7. The instances are represented by support property (reduced to two dimensional space [19]) and are clustered using K-means. Note: instances are colored w.r.t. to cluster assignment; cluster colors are randomly selected; instances are annotated with a numeric label according to Table I.

each vector element represents a specific physical property value. For the preliminary results, we acquired the physical properties of 46 objects in total which span across 16 object classes. The number of instances considered for each class is stated in the Table I.

The objective of this experiment is to examine whether the physical property measurements of the object instances convey the physical similarity between different objects. For this experiment, each object instance was represented in terms of its physical properties. The dimensionality of the instance data is initially reduced to two dimensions (see Fig. 6) using Isomap embedding technique [19]. Next, the reduced data is split into seven clusters (half of the total number of actual object categories) using K-means clustering. The results are depicted as a scatter plot in Fig. 6. In the plot, an object instance is represented as a point which is colored according to its cluster assignment. Furthermore, each point is attributed with a numeric label according to its class label, see Table I. The clusters group together the instances which are physically similar, e.g., in the red cluster the instances of *steel cup*, *ceramic bowl*, *plastic cup*, *plastic box*, *plastic bowl*, and *ceramic cup* are physically similar according to the given set-up.

Similar experiments were conducted to evaluate the functional similarities between the objects. Due to the lack of space, in Fig. 7, we have illustrated the similarity between objects with respect to ***support*** functional property. Accordingly, the instances of the object classes *plastic bowl*, *plastic cup*, *ceramic bowl*, *ceramic glass* and *ceramic cup* represented by a brown cluster have the similar degree of ***support***.

A. Limitations

While designing the property extraction methods and the framework, we aimed for approaches that do not require sophisticated hardware and software components. Therefore, we use a marker-based instead of a generic object tracking in our experiments. This mitigates the limitations that might be caused by complex approaches and facilitates robots to perform necessary experiments in an automated way. However, some limitations remain, which we discuss in this subsection.

One of the limitations is caused by mismatches between the definition of an object's property and the corresponding extraction method. For instance, instead of directly measuring the enclosed volume of an object to determine its ***hollowness***, we consider its internal depth and height. Consequently, the extraction method will calculate the same value for an object with a spherical hollowness as it calculates for an object with a cylindrical hollowness of the same depth although the enclosed volume is different. A similar mismatch can occur when calculating the ***size*** of an object, as we consider only its visual bounding box.

Another category of limitations are caused by the design of extraction methods. To measure the deformability of an object for extracting its ***rigidity***, it is placed on a planar, rigid surface. This surface itself provides ***rigidity*** to objects (e.g. for a sheet of paper) and thereby causes incorrect measurements. Similarly, ***roughness*** values for spherical objects are incorrect as these roll down the ramp instead of sliding.

Besides these methodological limitations, the employed hardware components impose limitations too. For instance, due to its size, the youBot's robotic arm does not allow extracting ***rigidity*** values for objects with a width greater than 20cm. This ultimately limits the objects that can be analyzed.

V. Current State and Future Direction

Standard datasets of the robotics community are generally created under supervision, one-dimensional, and discrete, i.e. an unary human-predefined label is given to an object sample; e.g. a point cloud is labeled as a *mug*. The presented work focuses on a framework for generating a dataset from a robot-centric perspective by gathering continuous conceptual object knowledge such as the functional property ***movability***.

We defined a set of object properties and their interrelations. Therein we distinguish between physical and functional properties. We show that these properties can be organized in a hierarchical bottom-up manner from low-level ones acquired from sensory data to high-level ones acquired from lower-level properties. Given this basis, we proposed acquisition procedures for each property. Eventually, we have introduced a framework consisting of property definitions and acquisition procedures and a corpus of 46 objects.

In our preliminary experiments we could show the discrimination of instances according to their physical and the functional property ***support***. These observed results encourage us to continue on our goal of creating a robot-centric conceptual knowledge base.

Therefore, we focus on extending the dataset with additional instances as well as classes in order to further investigate object understanding as such given the gained conceptual knowledge. Furthermore we aim to mitigate the discussed limitations. Considering the physical properties of ***flatness***, ***hollowness***, and ***size***, we plan for introducing 3D models of objects for generating more robust property values. Instead of directly processing noisy point clouds, a 3D model of an object will be created before extracting the respective properties.

While not considered in this early phase, failure modeling [20] and detection will be applied to enable failure-aware applications and to further mitigate the effects of sensor failures. Run-time approaches, such as the validity concept, were successfully applied to depth measurements of RGB-D sensors as well as to low-level distance sensors [21] and are specifically design to track sensor failures while propagating through a processing chain.

Finally, in the endeavor of enabling robots to perform the extraction methods autonomously, we plan for replacing the scale for measuring the objects weight. Instead, using the effort observations within the robotic arm can be used to determine the weight of an object while lifting it.

References

[1] C. Baber, "Introduction", in *Cognition and Tool Use*, 2003, pp. 1–15.

[2] M. Thosar, S. Zug, A. M. Skaria, and A. Jain, "A Review of Knowledge Bases for Service Robots in Household Environments", in *6th International Workshop on Artificial Intelligence and Cognition*, 2018.

[3] M. Tenorth and M. Beetz, "KNOWROB- Knowledge Processing for Autonomous Personal Robots", in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 4261–4266.

[4] Y. Zhu, A. Fathi, and L. Fei-Fei, "Reasoning About Object Affordance in a Knowledge Based Representation", *European Conference on Computer Vision*, pp. 408–424, 2014.

[5] L. A. Pineda, A. Rodríguez, G. Fuentes, C. Rascón, and I. Meza, "A light non-monotonic knowledge-base for service robots", *Intelligent Service Robotics*, vol. 10, pp. 159–171, 2017.

[6] R. Gupta and M. J. Kochenderfer, "Common Sense Data Acquisition for Indoor Mobile Robots", in *Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence*, 2004, pp. 605–610.

[7] I. H. Suh, G. H. Lim, W. Hwang, H. Suh, J. H. Choi, and Y. T. Park, "Ontology-based multi-layered robot knowledge framework (OMRKF) for robot intelligence", *IEEE International Conference on Intelligent Robots and Systems*, pp. 429–436, 2007.

[8] S. Lemaignan, R. Ros, L. Mösenlechner, R. Alami, and M. Beetz, "ORO, a knowledge management platform for cognitive architectures in robotics", *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, pp. 3548–3553, 2010.

[9] G. H. Lim, I. H. Suh, and H. Suh, "Ontology-based unified robot knowledge for service robots in indoor environments", *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, vol. 41, pp. 492–509, 2011.

[10] M. Daoutis, S. Coradeshi, and A. Loutfi, "Grounding commonsense knowledge in intelligent systems", *Journal of Ambient Intelligence and Smart Environments*, vol. 1, pp. 311–321, 2009.

[11] A. Saxena, A. Jain, O. Sener, A. Jami, D. K. Misra, and H. S. Koppula, "RoboBrain: Large-Scale Knowledge Engine for Robots", *ArXiv*, pp. 1–11, 2014.

[12] C. Baber, "The Design of Tools", in *Cognition and Tool Use*, 2003, pp. 69–80.

[13] ——, "Working With Tools", in *Cognition and Tool Use*, 2003, pp. 51–68.

[14] R. Bischoff, U. Huggenberger, and E. Prassler, "Kuka youbot - a mobile manipulator for research and education", in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 1–4.

[15] D. M. Swoboda, "A comprehensive characterization of the asus xtion pro depth sensor", 2014.

[16] T. Niemueller, A. Ferrein, D. Beck, and G. Lakemeyer, "Design principles of the component-based robot software framework fawkes", pp. 300–311, 2010.

[17] P. Soetens, "A software framework for real-time and distributed robot and machine control", *Doktorarbeit, Katholieke Universiteit Leuven, Belgien*, 2006.

[18] A. Koubâa, *Robot operating system (ros): The complete reference*. 2017, vol. 2.

[19] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction", *Science*, vol. 290, p. 2319, 2000.

[20] G. Jäger, S. Zug, and A. Casimiro, "Generic sensor failure modeling for cooperative systems", *Sensors*, vol. 18, 2018.

[21] J. Höbel, G. Jäger, S. Zug, and A. Wendemuth, "Towards a sensor failure-dependent performance adaptation using the validity concept", pp. 270–286, 2017.