Robot Google^{TM}: a Framework for Robot Knowledge Sharing

Xiuyi Fan and Thomas C. Henderson
School of Computing
University of Utah
Salt Lake City, UT, 84112 USA
Email: {fan, tch}@cs.utah.edu

Abstract— **Knowledge representation is a traditional field in artificial intelligence. Researchers have developed various ways to represent and share information among intelligent agents. Agents that share resources, data, information, and knowledge perform better than agents working alone. However, previous research also reveals that sharing knowledge among a large number of entities in an open environment is a problem yet to be solved. Intelligent robots are designed and produced by different manufactures. They have various physical attributes, use different knowledge representations and have different needs. In this research, we pose robot knowledge sharing as an activity to be developed in an open environment - the World Wide Web. Just as search engines like Google provide enormous power for information exchange and sharing for humans, we believe a searching mechanism designed for intelligent agents can provide a robust approach for sharing knowledge among robots. We have developed knowledge representation for robots that allows Internet access and a knowledge organization and search indexing engine that performs knowledge retrieval.**

I. INTRODUCTION

In the past when we needed to know something, we would look it up in an encyclopedia or find a book on the subject. Nowadays, we turn to web search engines, like Google^{TM}[1] or Yahoo^{TM}[2], and are given pointers to a large amount of information, and we usually find what we're looking for relatively quickly and easily. In this research, we develop similar capabilities for physical robots, including humanoid robots, which act in the world and must know a great deal about it. This includes robot butlers, surgeons, drivers, hospital orderlies, homecare nurses, etc. Thus, when a robot encounters an unfamiliar or unknown object in its environment, or when it needs to know how to perform a particular task with or on an object (e.g., clean it), it will be able to query a Robot Google in order to get pointers to relevant information available in the world wide web.

Humans achieve knowledge sharing mainly through natural language: queries are words that are matched to document content. For robots, it is not clear how to achieve this, and the question arises as to what representations best facilitate robot knowledge sharing. It is quite clear though, both the content of the knowledge representation and its format impact the effectiveness of this sharing. We would like to build our framework on solid ground so that knowledge representations derived from it allow common usage. Our framework provides a convenient yet unambiguous way to support the representation. Robots that use our framework may communicate with each other usefully.

[1]GOOGLE is a trademark of Google Inc.
[2]Yahoo is a trademark of Yahoo Inc.

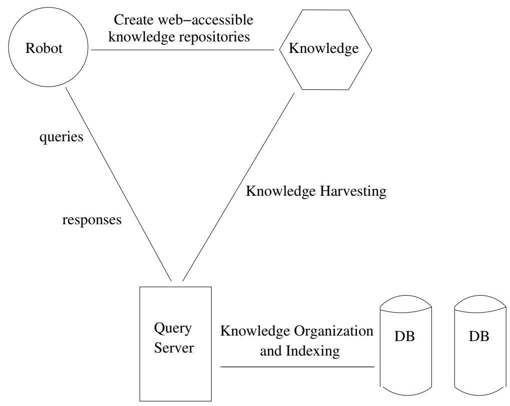

Fig. 1. The Robot Google Framework.

The developed framework for our solution is shown in Figure 1. In this figure, each participant robot creates web accessible knowledge repositories; the Robot Google server harvests knowledge from each of the participant robots and then creates and organizes efficient indexes into the database. Participant robots query the Robot Google server for knowledge and receive URLs pointing to other robots' knowledge. Robots, in this view, act as agents [1], [2], [3], [4], [5], and we assume their ability to generate the necessary knowledge structures; this is not an issue of investigation here.

Imagine a scenario like the following. A kitchen robot works in a kitchen. It is told to clean all kitchenware. After successfully cleaning a few plates, forks and spoons, the robot notices that there is a pair of wooden sticks. The robot is confused by this pair of wooden sticks as it has never seen this before. It does not know what to do with them. By asking a nearby human, the robot learns that this pair of wooden sticks is called "chopsticks." The robot then formulates a query and sends it to the Robot Google server though its on-board Internet connection. It wants to know if these chopsticks need to be cleaned (as it is told to clean all kitchenware) and if so, how to clean them. The Robot Google server processes the query, and responds with: please go to www.robot_chopstick.com/info to see more information about chopsticks and please go to www.robot_chopstick.com/clean to see how they can be cleaned. The robot then access these URLs and downloads the needed information. It then cleans the pair of chopsticks successfully.

As a general framework, we propose to consider a robot to be much like its human counterpart sitting at a computer terminal in the following sense. Two major paradigms have been established to support people:

1) Java virtual machine
2) Web-based search engines.

The first provides a means for a particular machine to emulate a universally shared virtual machine, and in effect, to embody another machine through the execution of byte-code. An analogous approach for robots requires the definition and adoption of a universal reference standard for some abstract mechanism, and the local embodiment of that abstract machine in the particular instance of a specific robot (e.g., executing bot-code). There could be a variety of abstract reference machines; e.g., mobile 3-DOF vehicles, 6-DOF robot arm mechnisms, or 44-DOF humanoid linkage mechanisms (e.g., see [6]). Each robot manufacturer would then provide interpreters for such abstract reference models.

The second paradigm, web-based search engines, addresses ways to find and share relevant knowledge. We describe in this paper a specific framework in which all queries and information gathering flow through the search engine (see Figure 1). This follows human use of a search engine and allows knowledge indexing, querying and relevance determination (the latter is possible since requests for URLs must also be submitted through the Robot Google search engine).

A large amount of work need to be done to realize the Robor Google scenario. Questions need to be answered include: (1) How does a robot communicate with Robot Google? (2) How does Robot Google process and answer a query? (3) How does one robot know if information stored on a website applies to its own environment? These are interesting questions that point to intensive research. This work focuses on solving one core problem: build a multi-format data search engine for robot knowledge sharing.

The next section describes some relevant background work. Section III presents our view of robot knowledge. Section IV introduces the Robot Google architecture. We discuss functions and designs of Robot Google components. Section V presents experimental results on our sample data set. We conclude in section VI with future research.

II. RELEVANT WORK

The study of knowledge representation can be traced back to ancient Greece. Epistemology, the study of the nature of knowledge and its justification, was established by Plato in the fifth century B.C. [7]. Since then, the study of knowledge, including its nature, representation, development, etc. has been carried on by philosophers, mathematicians, linguists, and scientists. Most knowledge representation developed today is rooted in various logics. Recently, some computer scientists have expressed belief that grounding knowledge purely in logic, e.g., in symbolic languages, is insufficient for building intelligent agents, e.g., robots. They propose to develop sensor grounded and context-aware knowledge representations for robot [8], [9]. Even though their work is promising, they are still far from providing a comprehensive and satisfactory solution.

Although still in its formative stages, several groups are making progress on sensor-grounded robot knowledge creation. Cohen et al. [10] describe a *natural semantics* approach in which robots learn meanings through their interaction with the environment. In natural semantics, meanings are acquired and maintained by the robot system, and not specified externally by human programmers or knowledge engineers. In this work, a robot is provided with a small number of behaviors (e.g., move, turn, open gripper, etc.), and the robot records sensor data streams. In this way, the robot learns a sensor data based ontology through interaction with the environment, and concepts are related to the sense data.

The Spatial Semantic Hierarchy, which allows bootstrap learning from uninterpreted experience [11], the human developmental theory based robot behavior study [12], the relational representation for procedural task knowledge [13], the cognitive robot [14], [15], etc, are some examples of a group producing sharable robot knowledge.

Another influential work in knowledge sharing in general is the Knowledge Interchange Format, known as KIF [16]. KIF was defined as an ANSI standard by the NCITS T2 committee on Information Interchange and Interpretation in 1998. KIF is a version of typed predicate logic, which differs from the sensor data grounded knowledge sharing. The book by Davies et al. [17] provides a very clear review of methods and tools developed for the human semantic web, which set its aim as to create a universal medium for information exchange by putting documents with computer processable meaning on the World Wide Web. Using the Semantic Web, information can be better organized and more accurately delivered to a human reader.

III. ROBOT KNOWLEDGE

Humans recognize the external world first through sensory organs. Imagine when we visit a museum, there are a number of artifacts on display. Suppose there is an object we do not recognize, but we would like to know what it is. We look at it to see its shape; we lift it to feel its weight; we may smell it to determine its odor; we may tap it with our finger to see how it sounds. With this collected sensory information, we try to associate this new object to some object we already know. We believe that humans recognize an object first by collecting information through sensory organs. Sensory information is the ground for object recognition.

We believe robots can behave similarly, and that the best way for a robot to recognize objects is through its sensor data. Therefore, we position this research towards a sensor data grounded approach. We restrict the scope of robot knowledge to be: string representation of sensor data. Strings give information about the object; this usually includes physical properties of an object and verbal descriptions. For a robot to know an object means to have information about that object stored in its memory.

The problem we need to address is: how do robots share their knowledge with each other, though a knowledge server such as Robot Google? We believe two knowledge transformations can help. The first transformation takes place in robots, where knowledge is transformed from the robot's internal representations, which are probably known only to themselves, to a form such that they are understandable to other parties, i.e., Robot Google and other robots. The second transformation takes place in Robot Google, where knowledge which is represented in the format produced by the first transformation, is transformed into a representation, that can be efficiently indexed. Then, Robot Google is able to effectively answer queries sent by other robots, hence knowledge is shared.

A. The First Transformation

The purpose of the first transformation, from a robot's internal format to an open standard, is to transform knowledge in a systematic way such that an unambiguous, widely-adoptable, format is obtained, while maintaining all information in a *knowledge piece*. Two requirements need to be satisfied. First, the data source, or inputs, of the transformation need to be collected in a standard way. We propose the idea of the *asymmetric spatial-temporal coherence* for objects. When a robot collects information about an object, i.e., measures its properties, we assume the robot does this in a uniform way such that all properties are measured with the least intervention among them. Once all information is collected through a robot's onboard sensors, the robot needs to package them tightly to maintain data integrity. Therefore, it is clear to both the robot, and Robot Google, at a later time, that information about one particular object, or an instance of an object, is collected, not just information about some object from a certain class.

Having a clear distinction between an object instance and an object class is significant to this work for two reasons. First, Robot Google is deeply rooted in the concept of sensor data grounded knowledge. Knowing which instance sensor data refers to is important to all possible higher levels, e.g., semantic level, knowledge structure formation at a later time. Second, it is desirable to support instance-based queries in addition to the general class-based queries. For example, to be able to determine that an image represents a human face is useful (the class-based query), but to be able to detect whose face it is (instance-based query) can be more useful for some applications.

We employ the standard Extensible Markup Language (XML) to represent knowledge as the result of the first transformation, as XML is widely used and accessible. The designed format is flexible enough to capture various types of knowledge while still being parser friendly.

Field	Value
IM1:red	[1.84e-011 -6.63e-009 4.44e-007 3.07e-005 -4.80e-004]
IM1:green	[2.23e-011 -8.72e-009 8.06e-007 8.99e-006 -2.25e-004]
IM1:blue	[2.20e-011 -8.55e-009 7.65e-007 1.37e-005 -4.56e-004]
IM2:red	[1.45e-012 2.51e-009 -9.89e-007 8.74e-005 -2.45e-004]
IM2:green	[3.11e-012 1.82e-009 -9.20e-007 8.84e-005 -3.80e-004]
IM2:blue	[4.23e-012 1.34e-009 -8.71e-007 8.94e-005 -5.51e-004]
IM1:edge	[-1.24e-010 6.39e-008 -1.07e-005 6.47e-004 -0.0063]
IM2:edge	[-1.37e-010 6.99e-008 -1.16e-005 6.97e-004 -0.0070]
LSI:	[-0.0509 0.2674 0.2571 0.4403]
Dim:	[6.0190 1.6906 11.3570 0.4998]
Weight:	0.4244
Filename:	'knife2a.jpg'
Description:	'Knife with black handle'

TABLE I

AN OBJECT EXAMPLE CONTAINS IMAGES, TEXT DESCRIPTIONS, DIMENSIONAL DATA AND WEIGHT MEASURE

B. The Second Transformation

The purpose of the second transformation is to convert the easy-to-communicate XML file into something that is easy to index. Hence we can build the search engine efficiently. We take the vector space approach.

Every knowledge piece in our system can be divided into three parts: text data, sensor data and meta data. Text data are provided by humans. This includes the name, function, use and possible other related descriptions of an object. Sensor data represents physical properties of an object. They are recorded by numerical values. For instance, the weight measure of an object is recorded with a single numerical value, given a standard unit is used; the shape of an object can be recorded by a histogram of the direction of the object's edges, where a histogram is represented by a vector. Meta data is recorded when the object is measured by sensors. It contains information about collected sensor data. For instance, the location of where the object is encountered, the time of when the object is encountered, the type/band/model of the sensor used to collect this data, etc.

Knowing the curse of dimensionality[18], we explored a few methods to reduce lengths of vectors, i.e., convert a high dimension vector to a lower dimension one, into especially histograms, including, Fourier coefficient representation, polynomial coefficient representation, statistics representation and moment representations. Our results show that representing a 256-bin color histogram by coefficients of a fourth order polynomial provides a reasonable compromise between data accuracy and vector length reduction; hence it is adopted in this research. Text information is converted into vector space using the latent semantic indexing (LSI) mechanism [19], [20]. For instance, a description such as "Small metal bowl" can be represented as $[-0.39, -0.12, -0.21, 0.08]$. Meta data can be represented by numerical values as well. For instance, time can be unambiguously represented by a UNIX time string; locations can be represented by GPS coordinates and sensor type can be represented by an index of the sensor in a sensor database or Logical Sensor System [21], [22]. Table I shows an example of an object represented in this vector space.

IV. KNOWLEDGE SEARCH ENGINE

As described by Frieden and Kuntz [23], the three main tasks of a search engine are to (1) match query keywords with related material on the web, (2) rank web documents according to relevance, and (3) provide pointers to the documents. Essentially, a search engine needs to be able to gather data from online sources, categorize and store them into a repository, and provide data access to its users. In the first generation robot search engine, i.e., Robot Google, we do not foresee a major role for web crawlers. Even if web pages exist, the meta data is not available to determine what pages to download, what is of interest in them (e.g., there are no words to count and no lexicon to help define any semantics), no popularity measure, and no standard places to find things (e.g., specific sites, in homepage, etc.). Thus, robots must register with the system and provide direct meta data and links to files. We focus on creating a robot search engine architecture that is:

- **Flexible.** Robot Google is not designed for any particular data type. We emphasize the *sensor data grounded* knowledge sharing model, in which virtually all types of sensor data can be supported.
- **Scalable.** Robot Google sets no limit on the number of attributes or properties of its supported data types. Both query templates and items stored in Robot Google can contain as many properties as needed.
- **Efficient.** Robot Google supports indexing on a large amount of data. The retrieval speed is not determined by the number of items stored in its database.

To accomplish these three tasks, we divide Robot Google into four groups of components: a query processor, indexing structures, a cross analyzer and a response formalizer (see Fig 2).

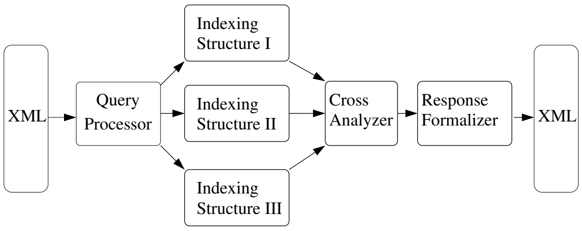

Fig. 2. The Robot Google Architecture.

A. Query Processor

Robot Google supports query-by-example [18]. The query processor is the first component in a Robot Google query workflow. It takes queries, in the format of XML, and translates them into arrays of vectors and sends each vector to a corresponding indexing structure to find matches. The query processor converts data stored in XML to vectors, and computes various derived features from raw data stored in the XML file, such as color and edge histograms and their low dimension representations from images. It also generates vector representations from text data.

B. Indexing Structures

The second group of components are indexing structures. They take inputs from the query processor in the form of vectors, and produce ordered lists of items. They sort items using measures between the query template and objects stored in Robot Google and return the list.

Currently, eleven indexing structures exist in Robot Google. Six of them are built for color histograms (each object contains two images, and there are three color channels in an image); two of them are built for edge histograms; one of them is built for text data produced from an LSI process; one of them is built for dimensional information of the object, i.e., length, height, width, and the cube root of the product of the three; the last indexing structure is built for the weight measure of one object. Currently, k-d trees have been used to index all fields except the weight measure, where a binary tree is used. In all k-d trees, branch dimension is selected in the round-robin fashion, starts from the left most entry/first dimension in a vector. This is due to the fact that eight histograms in one object are approximated by polynomials in which high order terms contribute more to the shape of the polynomial. Text data are processed by the LSI process, where at its core is a singular value decomposition (SVD), which has the same property that high order terms capture more information than low order ones. All indexing components return fifteen items for each query, except the text indexing and dimensional indexing component, in which thirty items are returned.

There are a few advantages for separating the indexing structure into distinct subclasses. First, this design allows Robot Google to support incomplete queries, i.e., queries with missing attribute fields, quite easily. For instance, if a query template contains only one image instead of two, indexing structures built for color and edge histograms for the missing image would not be used. All other Robot Google components would work the same as before. Second, it brings great flexibility to Robot Google as each indexing structure can be added, removed or modified without changing other components. New indexing structures can be added to support new data types and existing indexing structures can be removed if they are found to be not effective for retrieval. Different data types may require different types of indexing structures. If an existing structure is found to be less than ideal, it can be modified or replaced. Third, it brings more parallelism into a query process workflow. Each indexing structure processes one object attribute. Thus there is no intervention between any two indexing structures. A parallel process reduces search time for queries.

C. Cross Analyzer

The cross analyzer takes item lists from each indexing component, and cross analyzes them to produce a single sorted list. This list is produced based on a weighted summation of distances between a query template and items stored in the Robot Google database.

The cross analyzer first creates a list contains all items generated by indexing structures, without duplication. It then computes a distance measure from the query template to every item in the list. The distance measure is a weighted L_1 norm, which can be expressed as, the overall distance $D(A,B)$ between two objects A and B is equal to:

$$D(A,B) = \sum_{i=1}^{k} |w_i d_i(A_i, B_i)|$$

Where k equals the number of item attributes presented in the query; w_i is the weight coefficient of the ith component; and d_i is the distance between ith components in the two objects. All $d_i(A_i, B_i)$ are computed using an L_1 distance measure, where

$$d_i(A_i, B_i) = \sum_{j=1}^{k} |A_{i,j} - B_{i,j}|$$

For all image histograms represented by polynomial coefficients, k equals 5; for the text data and the dimensional data fields, k equals 4; and for the singleton weight measure, k equals 1. We have evaluated a set of distance measures for this computation, including L_2, L_∞, Mahalanobis, and Kullback-Leibler [18], [24]. L_1 gives the best compromise between computation efficiency and accuracy.

The cross analyzer provides Robot Google a simple control on item rankings through weight coefficients w_i. Since each indexing structure processes one item attribute, if for some reason, a certain attribute needs to be emphasized over others, the weight coefficient associated with this attribute can be adjusted accordingly. If we consider each item as a point in some high dimensional space, it can be viewed that weight coefficients can be used to dynamically enlarge or shrink the space in different dimensions. For instance, if we want to ignore a certain attribute in a query sample, we put a small weight coefficient for this attribute. This is equivalent to shrinking the space in dimensions used to represent this attribute. When a space is shrunk, the coordination of a point and the distance between two points no longer matter as all points are crowded together in this space.

We are at the very beginning stage to develop a systematic approach for computing weight coefficients w_i. Currently, a static analysis approach is taken. We design experiments for various data conditions and query types. In each experiment we evaluate Robot Google performance using the standard information retrieval measures: precision and recall [25]. We then search for weight coefficients that maximize these measures. The searching algorithm we have implemented is an n-dimensional binary search, which is a good compromise between simplicity and performance.

The cross analyzer controls the number of items returned by Robot Google. Since the cross analyzer sorts the item list returned by the indexing structures, two commonly used search types: k-Nearest-Neighbor search, i.e., using distance measure D, find k objects that are closest to the query and within-Distance (α-cut) search, i.e., using distance measure D, find all objects which are within α to the query template, are effortlessly supported.

D. Response Formalizer

The response formalizer is the last component in the Robot Google query workflow. It takes inputs, which are sorted item lists, from the cross analyzer, transforms them into a format which can be easily understood by robots, and then packages them into files. Snapshots of items, which contains basic information such as dimensional measure, and URLs which point to sources of items are both included. Containing snapshots in the returned file help robots to filter out unwanted results and find interesting information faster.

V. EXPERIMENT

To evaluate the performance of Robot Google, the common measures of precision and recall are used. Precision in information retrieval is defined as:

$$precision = \frac{|relevantdocument \bigcap retrieveddocuments|}{|retrieveddocuments|}$$

It is a measure of the percentage of results that are desired in the total retrieved list. Recall in information retrieval is defined as:

$$recall = \frac{|relevantdocument \bigcap retrieveddocuments|}{|relevantdocuments|}$$

It is the percentage of desired retrieved results in the entire database.

A. Object Knowledge

In this section, Robot Google performance on object data are examined. Sixteen objects, including four bowls, one cup, two forks, three knives, two plates, and four spoons are selected as data. Two images are taken of each object: a top view, and a side view. All images are taken against a white background and manually segmented. Images are stored in JPEG format in the size of 640x480 pixels. For each image, histograms of its color in RGB channels are computed, respectively, with 256 bins for each channel. The Sobel edge detection algorithm is applied to compute the distribution of edge orientations, and a histogram with 256 bins is obtained. To simulate the uncertainty of the real world, twenty-nine duplications of each object are created by adding 20% random noise from a Gaussian distribution to each object. The total sample size reaches 480.

Six groups of performance evaluation are presented. In tests groups one through three, relevant documents are defined as items which are generated from the same master copy by adding Gaussian noise. Group one shows the result where all documents returned from Robot Google are considered. Group two shows the result where only the top 30 most relevant items are considered. Test three shows the result where optimal weight coefficients are applied to rank returned items. Groups four though six use same settings, except "relevant documents" are defined as items which are in the same object class, e.g., fork, bowl, etc. Each group contains four tests. Test one examines Robot Google under optimal conditions, in which both query templates and objects stored in Robot Google contain complete information. Test two tests performance with incomplete queries, in which one image is missing in query templates. Test three tests performance with an incomplete data record. One third of Robot Google records miss one image in their record. Test four tests performance with both incomplete queries and incomplete records. Query templates miss one image and one third of Robot Google records miss the other image. Each test is composed of 200 runs. In each run, one object is randomly selected from the database as the query template. Average precision and recall are computed.

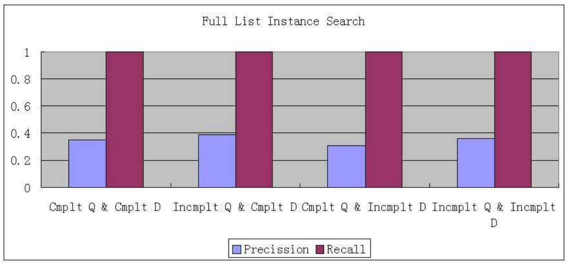

Fig. 3. Performance Test Group 1. All items returned from Robot Google are taken into consideration. Items derived from the same instance are relevant. Tests range from complete query & complete database to incomplete query & incomplete database.

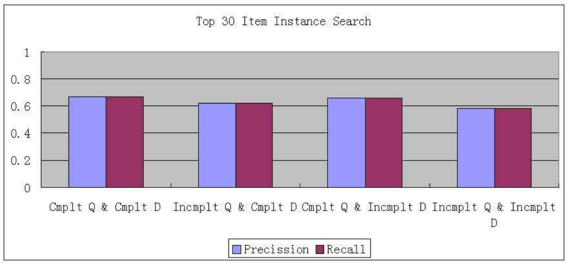

Fig. 4. Performance Test Group 2. Top 30 items returned from Robot Google are taken into consideration. Items derived from the same instance are relevant. Tests range from complete query & complete database to incomplete query & incomplete database.

Many observations have been made during our performance evaluation. In terms of precision, Robot Google shows good performance when the top items from returned list are considered. (Even though it is not shown here, when the top 10 items are evaluated, precision reaches 100% uniformly for every test.) In general, performance drops when queries

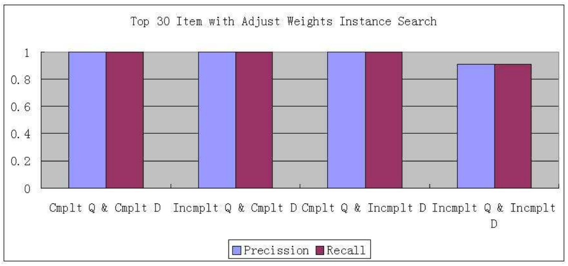

Fig. 5. Performance Test Group 3. Top 30 items returned from Robot Google are taken into consideration. Optimal weight coefficients are applied for ranking. Items derived from the same instance are relevant. Tests range from complete query & complete database to incomplete query & incomplete database.

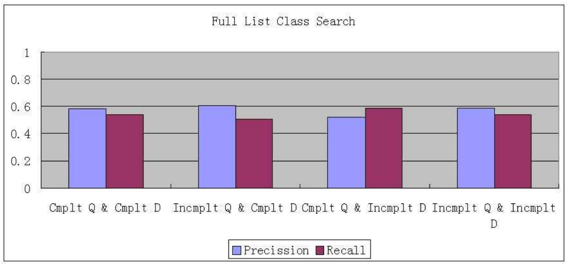

Fig. 6. Performance Test Group 4. All items returned from Robot Google are taken into consideration. Items belong to the same object class are relevant. Tests range from complete query & complete database to incomplete query & incomplete database.

are incomplete or database records are incomplete. However, we also have discovered that certain features are more useful than others when retrievals are concerned. For samples from our dataset, images and text descriptions are better classifiers than dimensional and weight measures. Hence missing images or text descriptions in a query shows more negative impact to retrieval performance than missing dimensional measures. The weight coefficient adjustment in the cross analyzer is a very effective tool for improving retrieval performance. It identifies attributes that have more classifying power, hence prevents or reduces interference from other attributes. To achieve a uniformly high performance on precision, a set of weight coefficients should be precomputed, based on the type of queries, i.e., attributes that are missing in a query.

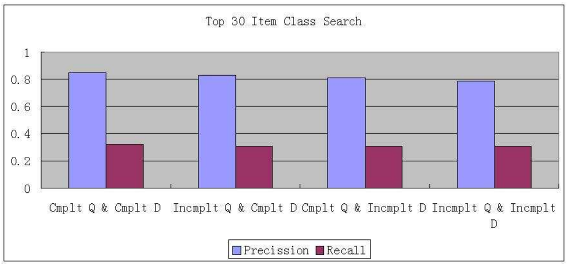

Fig. 7. Performance Test Group 5. Top 30 items returned from Robot Google are taken into consideration. Items belong to the same object class are relevant. Tests range from complete query & complete database to incomplete query & incomplete database.

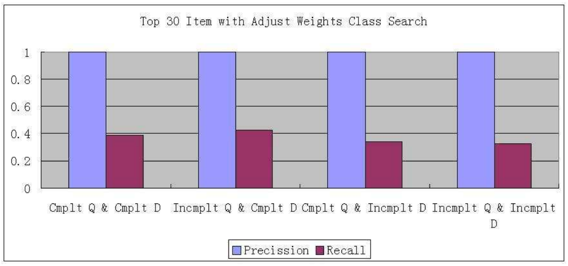

Fig. 8. Performance Test Group 6. Top 30 items returned from Robot Google are taken into consideration. Optimal weight coefficients are applied for ranking. Items belong to the same object class are relevant. Tests range from complete query & complete database to incomplete query & incomplete database.

B. Activity Knowledge

In addition to object recognition, human activity recognition is another field Robot Google can be used. Collaborating with Prof. Dillmann's humanoid robot research group at the University of Karlsruhe, we have obtained data generated by the *VooDoo human motion capture system* [6], [26], which gathers data of the human configuration over time, resulting in 3D trajectories for every modeled limb and joint angle of the human body. These data contain 8 activities with each activity performed multiple times, resulting 120 instances of activities. Since all instances are performed by a human experimenter, recorded lengths of instances range from 41 frames to 151 frames. These 120 instances are stored into an activity database.

Unlike the Feed Forward Neural Network (FFNN) classification model used in [6], Robot Google takes a simpler approach. In *VooDoo*, the human body is represented by 19 4-by-4 transformation matrices, where each matrix describes the state of a limb joint. In each transformation matrix, the upper left 3-by-3 sub-matrix describes the rotation of the joint, the right most column describes the movement of the joint. (See [26] for a complete discussion of the *VooDoo* system representation.) We exploit two of these matrices: one that describes the trunk of the body transformation and the other that describes the right forearm transformation, from each activity instance frame. The motion description is based on six values from each of the two transforms: 3 three diagonal elements of the rotation matrix and the three translation components. This results in 12 feature vectors. We then approximate the trajectory of every feature field across frames of an activity instance by a fourth order polynomial and index every instance of a trajectory into a k-d-tree. Therefore, in contrast to the eleven index structures developed for object knowledge sharing, a twelve index structure approach is adopted for activity recognition.

Activity	Correct
Hold Out Hand	91.0%
Hold Out Object	95.5%
Put Object On Table	89.9%
Read Book	73.6%
Sitting	89.9%
Standing	86.3%
Take Object From Table	77.7%
Typing On Laptop	100%

TABLE II

RESULTS FOR ACTIVITY RECOGNITION EXPERIMENTS.

We randomly select query templates from the activity database. Since the purpose of activity recognition is to identify human activities, Nearest-Neighbor search is more appropriate than k-Nearest-Neighbor search or α-cut search described in previous sections. We then limit the number of returned activities for each search to be 2 (since every search always returns the query template itself as the first activity) and define the classification as correct if the second returned item is the same activity as the query template. Results are presented in Table II. [6] indicates their FFNS approach reaches average correctness ranges from 53.1% to 100% for various activities; the Robot Google approach is comparable.

VI. CONCLUSION

This paper introduces the novel idea of developing robot virtual machines and robot search engines for robot knowledge sharing and discusses the architecture of the robot search engine we have developed. As a proof of the concept system, it demonstrates the merit of taking a sensor data grounded approach and using a flexible architecture. Robot Google shows promising performance in our object knowledge sharing experiments, where the search precision reaches 90+% for the items selected as most relevant. We have also examined Robot Google's performance with robot activity knowledge, in which Robot Google is used as an activity recognizer. In these experiments, Robot Google demonstrates comparable results to activity recognizers built by our colleagues at the University of Karlsruhe, in which a neural network approach is used.

Future research includes:

- development of virtual reference models for robots
- the study of Robot Google's performance on implemented, distributed robot systems
- investigation of a relevance feedback based approach for ranking returned items
- evaluation of the feasibility of porting this framework to intelligent software agent systems
- engagement of the robotics community into semantic robot web activity.

VII. ACKNOWLEDGMENTS

We would like to thank Prof. Rüdiger Dillmann's group at the University of Karlsruhe for sharing data with us, and in particular, Steffen Knoop and Pedram Azad.

REFERENCES

[1] J. Ferber, *Multi-Agent Systems An Introduction to Distributed Arificial Intelligence*. Harlow, England: Addison-Wesley, 1999.

[2] P. Maes, *Designing Autonomous Agents Theory and Practice from Biology to Engineering and Back*. Cambridge, MA: The MIT Press, 1990.

[3] V. S. Subrahmanian, P. Bonatti, J. Dix, T. Eiter, S. Kraus, F. Ozcan, and R. Ross, *Heterogeneous Agent Systems*. Cambridge, MA: The MIT Press, 2000.

[4] G. Weiss, *Multiagent Systems A modern Approach to Distributed Artificial Intelligence*. Cambridge, MA: The MIT Press, 1999.

[5] M. Wooldridge, *Reasoning About Rational Agents*. Cambridge, MA: The MIT Press, 2000.

[6] S. Knoop, S. Brannstrom, S. Vacek, and R. Dillmann, "Extraction, evaluation and selection of motion features for human activity recognition," in *Proceedings of the International Conference on Robotics and Automation*, Rome, Italy, 2007.

[7] J. F. Sowa, *Knowledge Representation Logical, Philosophical, and Computational Foundations*. Pacific Grove, CA: Brooks Cole Publishing Co., 2000.

[8] D. Roy, "Grounding words in perception and action: computational insights," *Trends in Cognitive Sciences*, vol. 9, no. 8, pp. 389–396, 2005.

[9] ——, "Semiotic schemas: a framework for grounding language in action and perception," *Artif. Intell.*, vol. 167, no. 1-2, pp. 170–205, 2005.

[10] P. Cohen and C. Beal, "Natural semantics for a mobile robot," University of Massachusetts, Department of Computer Science 2000-59, 2000. [Online]. Available: http://www-eksl.cs.umass.edu/papers/cohen-ECCS99.pdf

[11] B. Kuipers, P. Beeson, J. Modayil, and J. Provost, "Learning from experience in the ssh," in *AAAI Spring Symposium Series: Learning Grounded Representations*, Stanford, CA, March 2001.

[12] R. Grupen and M. Huber, "A framework for the development of robot behavior," in *2005 AAAI Spring Symposium Series: Developmental Robotics*, Stanford, CA, March 2005.

[13] S. Hart, R. Grupen, and D. Jensen, "A relational representation for procedural task knowledge," in *Proceedings of the AAAI Conference*, Pittsburgh, PA, 2005.

[14] R. Becher, P. Steinhaus, R. Zollner, and R. Dillmann, "Design and implementations of an interactive object modeling system," in *IRS 2006 37th International Symposium on Robotics*, Munich, 2006.

[15] R. Dillmann and T. Asfour, "Perception, action & cognition through learning of object-action complexes," University of Karlsruhe, Germany, Tech. Rep., Project FP6-2004-IST-4-27657 2005.

[16] M. Genesereth, "Knowledge interchange format," in *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference*, Cambridge, MA, 1991, pp. 389–396.

[17] J. Davies, D. Fensel, and F. V. Harmelen, *Towards the Semantic Web Ontology-driven Knowledge Management*. West Sussex, England: John Wiley and Sons LTD, 2003.

[18] V. Castelli and L. Bergman, *Image Databases: Search and Retrieval of Digital Images*. New York, NY: John Wiley & Sons, 2002.

[19] S. Sclaroff, M. L. Cascia, S. Sethi, and L. Taycher, "Unifying textual and visual cues for content-based image retrieval on the world wide web," *Computer Vision and Image Understanding*, vol. 75, no. 1-2, pp. 86–98, July 1999. [Online]. Available: http://dx.doi.org/10.1006/cviu.1999.0765

[20] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society of Information Science*, vol. 41, no. 6, pp. 391–407, 1990. [Online]. Available: http://citeseer.ist.psu.edu/deerwester90indexing.html

[21] M. Dekhil and T. C. Henderson, "Instrumented logical sensors systems," *International Journal of Robotics Research*, vol. 17, no. 4, pp. 402–417, 1998.

[22] T. Henderson and E. Shilcrat, "Logical sensor systems," *Journal of Robotic Systems*, vol. 1, no. 2, pp. 169–193, 1984.

[23] N. Fielden and L. Kunt, *Search Engine Handbook*. Jefferson, NC: McFarland and Company, Inc., 2002.

[24] J. Puzicha, Y. Rubner, C. Tomasi, and J. M. Buhmann, "Empirical evaluation of dissimilarity measures for color and texture," in *ICCV (2)*, 1999, pp. 1165–1172. [Online]. Available: citeseer.ist.psu.edu/puzicha99empirical.html

[25] Wikipedia, "Information retrieval — wikipedia, the free encyclopedia," 2007, [Online; accessed 4-March-2007].

[26] S. Knoop, S. Vacek, and R. Dillmann, "Sensor fusion for 3d human body tracking with an articulated 3d body model," in *Proceedings of the 2006 IEEE International Conference on Robotics and Automation*, Orlando, Florida, 2006.