Developing and Analyzing Intuitive Modes for Interactive Object Modeling

Alexander Kasper
Institute for Computer Science and Engineering
University of Karlsruhe, Germany
akasper@ira.uka.de

Regine Becher
Institute for Computer Science and Engineering
University of Karlsruhe, Germany
becher@ira.uka.de

Peter Steinhaus
Institute for Computer Science and Engineering
University of Karlsruhe, Germany
steinhaus@ira.uka.de

Rüdiger Dillmann
Institute for Computer Science and Engineering
University of Karlsruhe, Germany
dillmann@ira.uka.de

ABSTRACT

In this paper we present two approaches for intuitive interactive modelling of special object attributes by use of specific sensoric hardware. After a brief overview over the state of the art in interactive, intuitive object modeling, we motivate the modeling task by deriving the different object attributes that shall be modeled from an analysis of important interactions with objects. As an example domain, we chose the setting of a service robot in a kitchen. Tasks from this domain were used to derive important basic actions from which in turn the necessary object attributes were inferred.

In the main section of the paper, two of the derived attributes are presented, each with an intuitive interactive modeling method. The object attributes to be modeled are stable object positions and movement restrictions for objects. Both of the intuitive interaction methods were evaluated with a group of test persons and the results are discussed. The paper ends with conclusions on the discussed results and a preview of future work in this area, in particular of potential applications.

Categories and Subject Descriptors

H.5.2 [**User Interfaces**]: Ergonomics; H.5.2 [**User Interfaces**]: Evaluation/methodology; H.5.2 [**User Interfaces**]: Haptic I/O; H.5.2 [**User Interfaces**]: *complexity measures, performance measures*

General Terms

Human Factors, Performance, Experimentation

ICMI'07, November 12-15, 2007, Nagoya, Aichi, Japan.

1. INTRODUCTION

When humans interact with their environment they rely on a vast amount of background knowledge about the various objects they encounter. They acquire this knowledge by learning the specific attributes and functionalities of different sorts of objects. If a robot shall be able to interact with its environment in a similar way, it will need similar background knowledge about its surroundings and about the existing objects. As robots currently do not have learning abilities comparable to those of humans, their knowledge about objects must be modeled and acquired in a different way.

The most convenient way would be to have a system that can learn attributes and aspects of a given object autonomously. However, learning object knowledge is a very complex task and the current results in this area are not satisfying. In particular, some forms of object information like functional, semantic or user-specific knowledge are difficult to obtain without human help. Thus, we explored a human-supervised modeling approach where we make use of the human background knowledge. In this approach, the *model* of an object is created interactively with the aid of the human user, and the actual parameterization process is supported by automatically generated information. Such a combination allows for both a relatively fast and a very accurate model generation.

Important object attributes and sensible ways to identify their values interactively or automatically have been investigated in a systematical way in the course of our work. For this, the domain "‘household"’ was chosen, and important objects and actions in this domain have been identified and relevant attributes of objects derived. Based on these results, we selected appropriate interaction modes or ways of automatic processing to gather the relevant object attribute values, depending on the type of the required information. Two such ways of interactively gathering object information are presented and evaluated in detail in the course of this paper: the determination of stable positions e.g. for place operations and of movement restrictions e.g. for carrying objects.

The paper is structured as follows: After a general overview over the current research in interactive object modeling, a brief introduction into the underlying structure of the object models is given and the results of the systematical study of the domain "‘household"’ are briefly presented, in particular the important object attributes for typical tasks in household environments. We then describe the methods and evaluation results for the object attributes "‘stable positions"’ and "‘movement restrictions"’. The paper closes with conclusions and an outlook onto future work in this area.

2. RELATED WORK

In the field of human-computer interfaces, there has been a wide variety of research over the last decades. From the first pointing devices to recent implementations of gesture and speech recognition, many approaches have been studied. The developed interfaces can be divided into several groups: haptic interfaces, acoustic and visual communication. At first, we will give a brief overview over the different typical interaction modes of these groups, while concentrating on works using the same hardware than we used here afterwards. Finally, we will conclude this section with a brief paragraph on interactive object modeling itself.

One category of interfaces using visual communication are systems that use the user's eye gaze. Jacob ([5]) used such a system to interpret user intention for interaction with the system. He therefore segmented the data into tokens that represented several eye gaze states which were then interpreted similarly to the tokens delivered by a common keyboard. A quite similar approach to give disabled patients a means for steering a computer system was taken by Hutchinson et al. in [4]. They developed a special user interface organized in a tree structure that was customized to take input of an eye gaze tracking system.

A category of interfaces spanning both visual and haptic communication are the hand and gesture tracking and recognition systems. Murakami and Taguchi present a system that can recognise sign language in [7]. Using neural networks and data gloves they train their system to be able to distinguish many different finger configurations. A similar task is solved by Hardenberg et al. in [17] where they replace the data glove with a camera and recognise the user's hand and fingers by means of computer vision.

Speech could be a promising candidate for intuitive human computer interaction. While there have been great advances in the field of speech recognition, in [14] Shneiderman displays its limits as a natural interface. He states that speaking and thinking for most people are so deeply entwined, that the use of a speech recognition system can worsen the quality of a user interface. Nonetheless is speech a versatile communication channel whose use has its own challenges and difficulties, explored further in [1].

Haptic interfaces are characterized by interfaces employing special hardware, from simple mechanic devices to high complex mechatronic hardware, to bridge the gap between human and machine. A very simple example which is widespread these days is the generic computer mouse. The translation of the device is mapped directly to the translation of the pointer on the graphical user interface. A far more complex haptic interface, communicating from the user to the machine and backwards, is the *Pantograph* introduced by Ramstein and Hayward in [12]. The device is programmable to emulate different surfaces via force feedback to the user and receive input through finger motion. This allows for a bidirectional communication between user and machine.

In our experiments we used the InterSense Inertia Cube2 and the Polhemus Fastrak, both of which are widespread as interactive devices. A 6 DOF input device for single or double handed use is developed by Simon and Doulis in [15]. The device is composed of a 6 DOF sensor, measuring 3 DOF forces and 3 DOF torques, and a 3 DOF inertial sensor (the InterSense InertiaCube). This allows for a variety of different input and is aimed at a virtual reality environment. Two other works in this field, which employ the Polhemus Fastrak, come from Poupyrev ([11]) and Poston ([10]). Both use the sensor to track the user's hand and apply this data to represent the user's hand, or an equivalent tool in the virtual space. Another interesting usage of the InertiaCube can be found in [9], where Petridis et. al. integrated the device into a replica of a museum exhibit. A digitized version of the original exhibit is then shown on a screen and the visitors can rotate and manipulate this visualization via the replica.

However, the vast majority of these works concentrate on making an interface that is applicable to a wide range of tasks and do not focus on a highly specialised modeling process. Though, especially this can yield, if applicable, very good results, which we will show in the rest of this paper.

3. LEARNING OBJECT MODELS FOR SERVICE ROBOTS IN HOUSEHOLD ENVIRONMENTS

The goal of our research is to create an object model hierarchy for a humanoid robot. This hierarchy should be created by a human user with little programming knowledge, supported by the modeling system. As an application example, we use a typical kitchen environment. The next two sections describe first the internal structure of the object models in more detail and then lead to the deduction of the different modeling tasks that need to be solved in such environments by intuitive user interaction.

3.1 The Object Model Hierarchy

In order to represent all the different objects that can be found in everyday environments, the object model needs to be very flexible and versatile. To achieve this, the proposed model consists of four main parts: object classes, object instances, features and attributes. In this concept, object classes consist of features (and potentially additional attributes), whereas features in turn consist of one or more attributes. Attributes are low-level descriptions of object properties, such as geometry, weight, texture, etc. Features describe higher level properties of objects which combine different attributes, e.g. the feature *is container* which implies attributes like *filling state*, *content type*, etc.

On the top end of the hierarchy, object classes represent complete families of objects, such as cups, plates, forks or chairs. Objects of the same object classes share characteristic features and attributes. By setting special default values for the attributes of the object classes, sub-classes like e.g. *wooden chair* can be specified. Finally, object instances represent objects in a real world scene by instancing the appropriate object class and thus, setting situation and object specific values for its attributes. A more detailed explanation of this approach can be found in [3] and [2].

Based on this object model concept, two questions need to be answered to create a model for a real world scene: first, which attributes need to be modeled to describe the objects in the scene properly? And second, how can appropriate default values for these attributes be set by the human user in an intuitive way? The next section answers the first question whereas the second part of this paper describes a possible answer to the second question for two exemplaric attributes.

3.2 From Tasks to Object Attributes

To create meaningful object classes to represent real world objects, the core attributes that are common to all objects of the domain in question need to be identified and modeled. The identification of these attributes was achieved through a careful analysis of possible interactions with the observed objects. This analysis consisted of three steps: the identification of the potential interaction tasks, the separation of the tasks into basic actions and finally the derivation of the attributes which are necessary to execute these basic actions.

The setting of service robots in a kitchen was taken as exemplary domain in our investigation. The analysis therefore concentrates on interaction tasks which service robots will be able to carry out in the near future. In the following two subsections, the derivation of basic actions from these tasks and of required object attributes from actions are presented.

3.2.1 Deriving Basic Actions

For the chosen domain, we identified several important tasks: For the fundamental recognition and localization of objects, the encompassing task is perception. As fig. 1 exemplifies, perception tasks can be subdivided into three basic actions: classification, identification and localization. Fig. 1 also shows that other tasks like pick & place, too, partly employ the same basic actions, but rely on additional actions like approach, grasp, move etc.

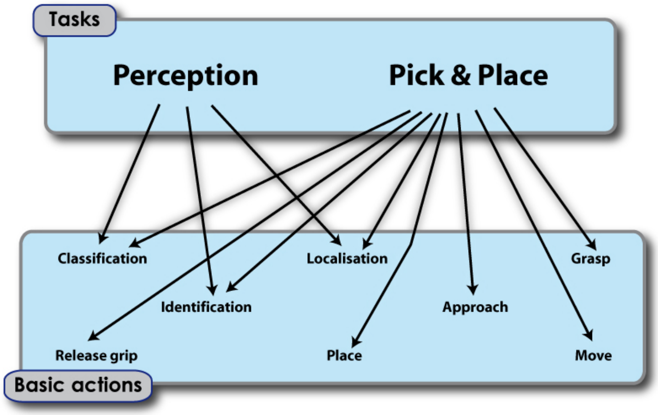


Figure 1: Exemplary tasks and corresponding basic actions

In the same way, the remaining tasks of open/close, fill/empty and utilize were broken down into several basic actions. The analysis revealed that many of the basic actions are part of more than one task. The aforementioned classification, identification and localization, for example, are integral parts of each of the analyzed tasks.

3.2.2 Deriving Required Attributes

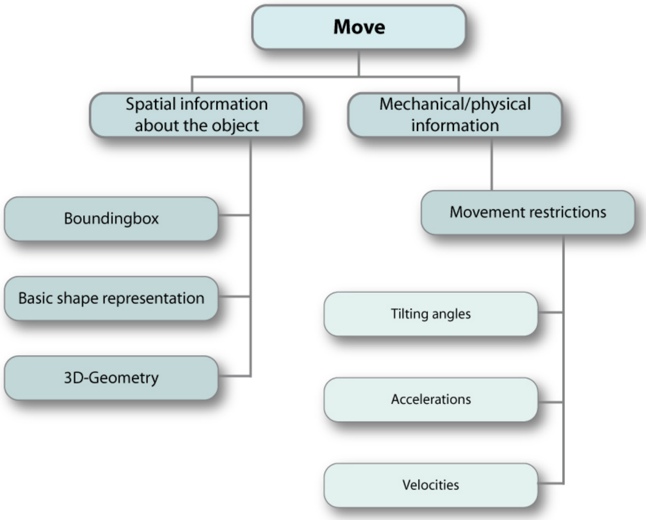


Figure 2: The process of action analysis (here: movement of an object)

Now that the basic actions are known, the useful attributes required to perform these actions can be derived. Figure 2 shows this process at the example of moving an object. In this case spatial information is needed e.g. to avoid collisions. This can be represented for example as a bounding box, in form of a basic shape representation or through highly detailed 3D geometry. When moving the object, mechanical and physical information is also crucial. Most important here are the movement restrictions, i.e. tilting angles, maximum accelerations and maximum velocities. In this fashion all of the basic actions, derived from the set of possible interactions in the environment, were analyzed and thus necessary object attributes extracted.

The most important resulting attributes for this domain are: movement restrictions, basic shapes, weight, bounding box, main axes, stable positions, grasp forces, grasp contact points, deformability, environmental conditions, risk potential, texture, colour graph, 3D geometry, type of locking mechanism, closability, graph of potential usages, container type and filling capacity. These attributes require different approaches to achieve intuitive, fast and exact object modeling. The next part of this paper focuses on two of these attributes, namely stable positions and movement restrictions, and the suggested methods for interactively modeling them.

4. STABLE POSITIONS

This object property is important for pick and place operations. If a robot shall be able to place an object securely at a specific place, it needs to know in which orientations the object can be placed down for it to rest balanced on a surface. These orientations can be represented through planes at the object surface that are appropriately positioned and oriented relatively to the object's 3D geometry. A cup for example can be placed securely on its bottom, which can be represented by a plane that is parallel to it. For simple objects finding these planes can be retrieved automatically,

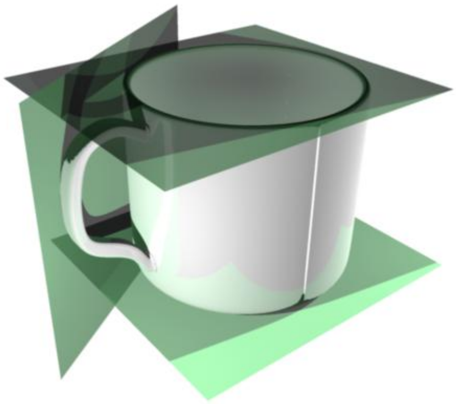


Figure 3: Potential stable positions of a cup

but with growing complexity autonomous retrieving quickly reaches its limits. Additionally, not all potentially stable positions are always desired, and some stable positions may depend on the context. E.g., a full cup should only be put upright, but an empty cup in the dishwasher should be put upside down. It is therefore more efficient and more reliable to use a human's cognitive capabilities for this task and to support him with some automatic functions. To make that task as agreeable as possible for the human user, we have investigated a specific method of intuitive interaction for this modeling task.

4.1 Methods

Our idea for modeling stable positions is for the user to imitate the action of placing the object on a flat surface in a virtual environment. Since defining the stable planes of an object only makes sense if they are correlated directly to the 3D geometry of the object, the latter is obviously a good starting point for the modeling process. In our current system, a 3D point cloud representation of the object of interest is therefore required. This can be retrieved by a laser scanner for example. This representation is not only used as the base for correlating the stable planes to the object geometry, but is also a very good visualization for the user during the modeling process. To achieve this, we use the point cloud representation to visualize the object, so that the user can directly see the results of his manipulations.

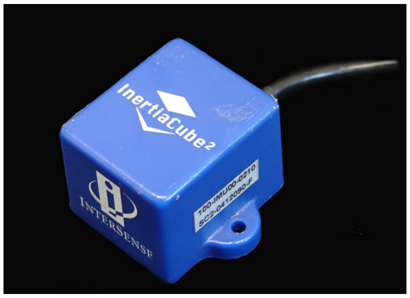


Figure 4: InterSense InertiaCube2

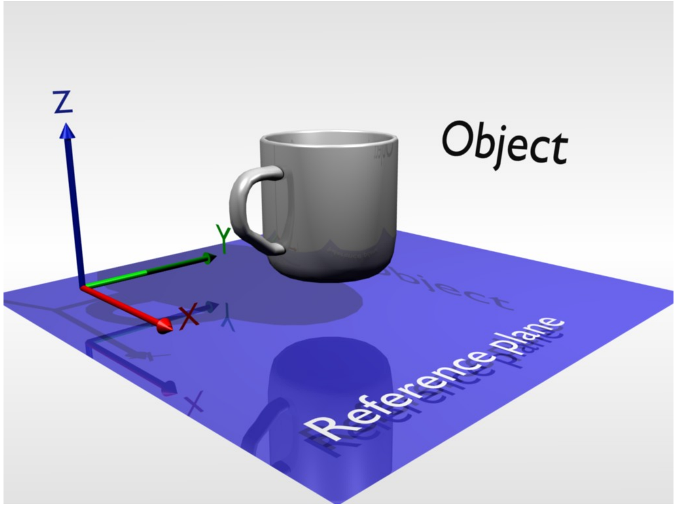
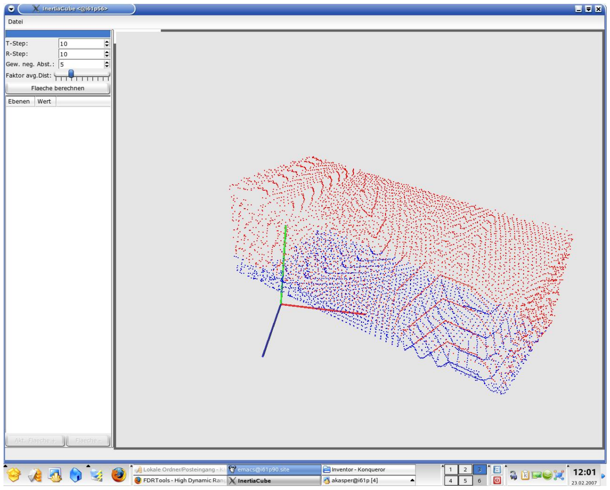


Figure 5: Relations between object, reference plane and coordinate system

As a means to manipulate the object orientation we use a special input device, the InterSense InertiaCube2 (fig. 4). This device is a 3 DOF sensor which can measure rotations. The sensor's orientation is mapped directly onto the point cloud representation, so that the user is able to rotate the object in the graphical user interface by rotating the sensor. In this way the user can "place" the object on a surface in the same orientations that he would in the real world.

The surface is in our system a plane parallel to the x-y-plane (see fig. 5), below the point cloud of the object. We visualize the orientation of the object relative to this plane by coloring the points of the point cloud that are within a certain distance of said plane (cf. fig. 6). If the object is rotated with the InertiaCube, the selected points change dynamically, thus showing the object's current orientation. At the same time the current set of selected points is the basis for an automatic calculation of a plane of best fit representing a stable position. The user can start such a calculation by pressing a button when the object has the desired orientation.



T-Step: 10
R-Step: 10
Gew. neg. Abst.: 5

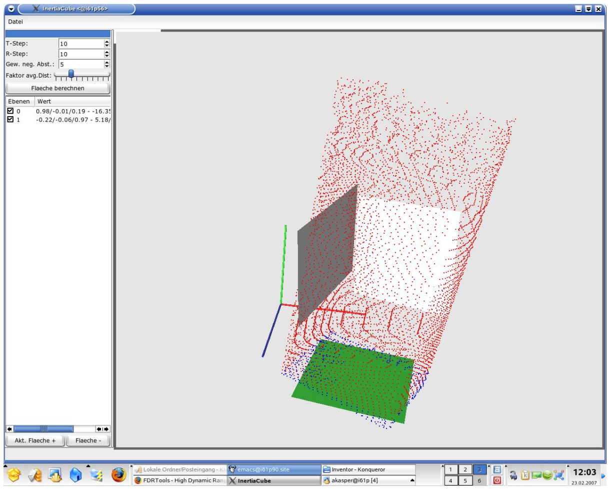
Figure 6: Point cloud with selected points

The calculation of this plane is done using a gradient descent. First, the center of gravity ($\vec{P}_{COG}$) of the selected points (M_{sel}) is calculated as in equation 1.

$$\vec{P}_{COG} = \frac{1}{k} \cdot \sum_{i=1}^{k} (P_i), \forall P_i \in M_{sel} \tag{1}$$

The starting plane for the gradient descent is created, using this calculated point and the normal vector of the virtual "deposit surface" $(0,0,1)^T$. We then calculate the distance of every point of the selection to this new surface and use the squared sum of it to rate the surface. A minimal rating is the goal of the gradient descent algorithm.

After the calculation of the newly found stable position, the plane is visualized in the scene together with the point cloud and, if appropriate, other existing planes (cf. fig. 7). The user can then judge if the calculated plane is a good enough approximation and may either keep it or make a new calculation with a slightly different point set. The modeled stable positions can then be exported for further use.



T-Step: 10
R-Step: 10
Gew. neg. Abst.: 5
Faktor avg.Dist:

Ebenen	Wert
☑ 0	0.98/-0.01/0.19 - -16.35
☑ 1	-0.22/-0.06/0.97 - 5.18/

Figure 7: Calculated planes (white) and current plane (green)

4.2 Evaluation

The interactive modeling process was evaluated with a small group of test persons. In total there were 8 subjects, four of which were instructed in-depth in the use of the program, whereas the other four had to explore the program by themselves and were only instructed on the background and purpose of the modeling. Subjects were given a sample point cloud of a box and the task of modeling all 6 stable planes of the object. This task had to be solved three times by each subject. Recorded data for evaluation was: time for task completion, found plane normals and a questionnaire filled out by each subject. The latter included 6 questions, covering three areas of interest:

- Graphical user interface
    - Interface content (satisfaction with the number and kind of presented information)
    - Interface structure (clearness)
- Hardware
    - Technology/ergonomics (satisfaction with the hardware interface in general)
    - Incorporation (eligibility of the hardware in this scenario)
- Intuitivity
    - Graphical user interface (usability/ease of use)
    - Hardware (usability/ease of use)

The subjects were asked to answer these questions on a scale from 1 (very good) to 6 (very bad). They were also given the opportunity to add comments and problems in text form.

Results.
The subjects were instructed to repeat the given task three times. This allows us to estimate on how strong the influence of being used to the process would be on the time needed to complete the task. Both groups showed here that their performance in this area considerably increased with more repetitions. Members of the instructed group, for example, needed 8 minutes in the first run on average, while dropping to 3 minutes in the third run. While this drop was expected in both groups, the total amount of time needed surprisingly doesn't differ much between groups. This indicates that the user interface allows for uninstructed usage. Although the number of subjects is too small for a well-founded statement, it clearly indicates that the presented interface is quite intuitive. It was also rated very well from the subjects.

Here, the applied hardware - the InterSense InertiaCube2 - did achieve both the best and the worst score. The best, with an average of 1.5 in terms of intuitivity/ease of use for the special application, and the worst, with an average of 2.875 for its ergonomics. The explanation for the latter can be found in the comments of the subjects. Almost all of them stated, that the InertiaCube was indeed very intuitive for the task at hand, but the fact that they had to hold it perfectly still at the moment they wanted to calculate a plane was seen as the major drawback of the process. Some other technical solution will have to be found here.

Evaluation of the data gathered from the plane extraction yields a positive image of this modeling method. In figure 8 the extracted normals can be seen, which coincide mostly among each other and with the expected plane normals of a cuboid.

	Group A	**Group B**	**Average**
GUI			
Content	1.75	2.25	2.00
Structuring	1.50	2.25	1.875
Hardware			
Ergonomics	2.75	3.00	2.875
Integration	2.00	2.50	2.25
Intuitivity			
GUI	1.25	2.00	1.625
Hardware	1.50	1.50	1.50

Table 1: Overall scores of the modeling process "stable planes" (1=very good, 6=very bad)

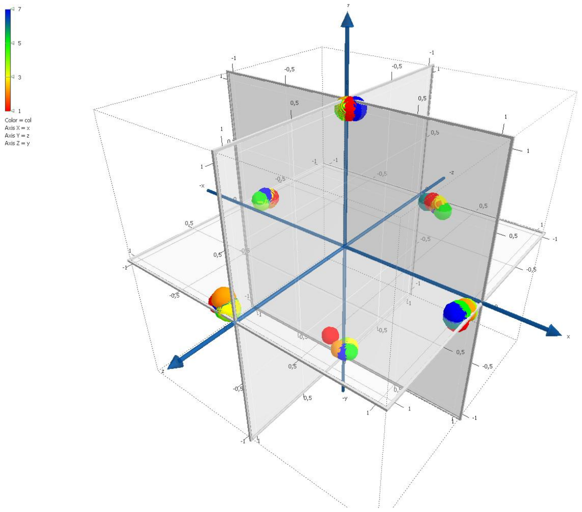


Figure 8: Experimental results: plane normals generated by subjects

		Pass [mins]		
Group	**Candidate**	**1**	**2**	**3**
A	1	20	11	8
A	2	8	6	4
A	3	3	6	4
A	4	6	3	3
	avg.	9.25	6.5	4.5
B	1	11	7	2
B	2	5	3	3
B	3	6	2	2
B	4	6	5	3
	avg.	7	4.25	2.25

Table 2: Time needed for modeling stable planes

Table 2 shows the gathered data for the group of 8 subjects for completing the task. Group A contained the subjects that got an in-depth introduction to the program whereas group B consisted out of the subjects that had to explore the application by themselves. Interestingly, the latter needed less time in average than the first group. This is due to the fact that the instructed subjects used more of the features the interface offered, while the uninstructed subjects didn't discover all of these. The overall discrepancies in task completion time are due to the fact that subjects were only told to do "as accurate as possible", which some took more serious than others. Additionally, some people had general difficulties with navigating a 3D scene on a computer display, while most others did not.

In conclusion, the proposed modeling process can be described as very successful. All of the subjects were able to complete the task with acceptable accuracy. The goal of intuitivity and ease of use was mostly reached, even in this prototypical stage. Some adjustments to the setup need to be made especially to further improve the ergonomics and the GUI. One possibility would be to use a wireless InertiaCube mounted into a sphere which can then be rolled around on the desk with the fingers like a big trackball. The GUI would benefit from a new layout of the buttons.

5. MOVEMENT RESTRICTIONS

Equally important for pick and place operations as the stable positions are *movement restrictions* related to an object. This object property encompasses the most important aspects of object movement: how fast can an object be moved relative to its main axes, what are its highest possible accelerations and what are the ranges for rotations around its axes? The resulting data are maximum values for velocity, acceleration and rotation for all 6 DOFs of the object.

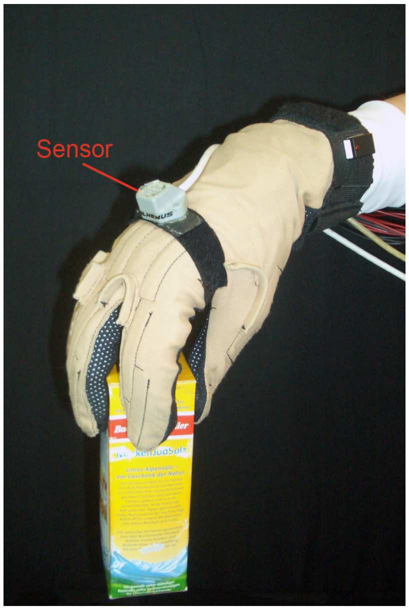


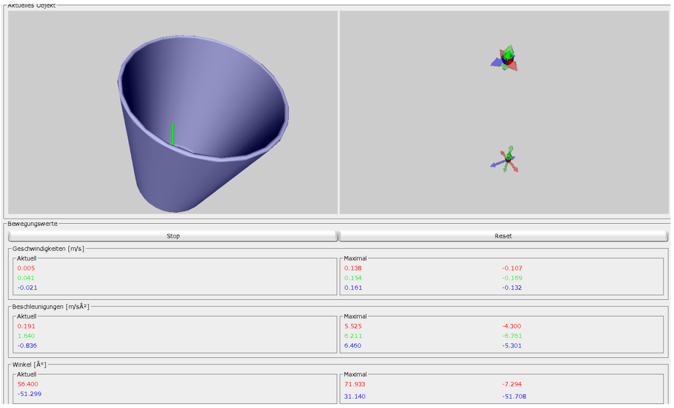
Figure 9: Polhemus Fastrak mounted on glove

5.1 Methods

To model this object property, again we aimed to imitate a natural task for a human with an object. The relevant data is than to be gathered from observing this action. In this case we used a Polhemus Fastrak sensor attached to a glove to track the motion and orientation of the user's hand, which in most instances equals the object motion. For modeling movement restrictions intuitively, the user is instructed to make typical movements with the object of interest. These movements are recorded by the system and the resulting maximum velocities, accelerations and tilting angles are displayed immediately to the user in a simulation. That way, the user has good control of the ongoing modeling process and the recorded data.

The application itself is built in the fashion of a *wizard*. The process is divided in 4 steps: selection of the object of interest (a point cloud for visualization), selection of the values to track, demonstration of the movements, and finally display of the gathered data. Step three, which is of particular interest, is shown in fig. 10. For demonstrating, the user needs to grab the object, then start the recording of the data. Calculated speeds, accelerations and angles are displayed immediately.

The calculation of the required values is straightforward. The global sensor values have to be converted into values relative to the local object coordinate system. First, the initial rotation of the sensor when the user grabs the object



```
Geschwindigkeiten [m/s]
Aktuell          Maximal
0.005            0.138      -0.107
0.041            0.154      -0.169
-0.021           0.161      -0.132
Beschleunigungen [m/sÂ²]
Aktuell          Maximal
0.191            5.525      -4.300
1.640            6.211      -6.761
-0.836           6.460      -5.301
Winkel [Â°]
Aktuell          Maximal
56.400           71.933     -7.294
-51.299          31.140     -51.708
```

Figure 10: Recording movements with the object

needs to be reversed. Since the sensor records with a frequency of approximately 4 Hz, the velocities are calculated based on the last and the current set of data. To achieve object-relative data, the last set needs to be converted into the current coordinate system via a rotation of the translation vector. This vector is now divided through the time elapsed between the two measurements to get the velocity. Based on this result, it is easy to determine the according accelerations. The calculation of the maximum angles of rotation is only slightly more complex, for the sensor outputs only Euler angles in pitch-yaw-roll convention relative to the global coordinate system. These angles need to be converted and then set against the initial rotation, resulting in the relative pitch, yaw and roll angles.

5.2 Evaluation

The modeling process was evaluated by the same group of test persons who tested the modeling of the stable object positions. Again, 4 of the 8 subjects got in-depth instructions of the user interface while the other 4 had to find out its usage on their own. Equivalent questionnaires were given to the subjects for quantitative and qualitative analysis. A comparision of the generated movement restrictions would not yield useful results in this case, because the subjects were completely free in the choice of demonstrated movements.

The main focus of the evaluation was therefore on the acceptance and usability of the interface. In these areas, the presented modeling process scored even better than the modeling of the stable positions. Table 3 lists the gathered data in detail. Again, no significant discrepancy between instructed (group A) and uninstructed (group B) subjects could be found.

Subjects particularly rated the wizard-like structure of the application as positive, as it made it easy to set the required parameters and as it guided the users through the modeling process.

As can be seen from table 4, the time for task completion in this case is rather short. Subjects needed the most time for the first run, which is due to them reading the instructions provided in the graphical user interface. These instructions were known in the second and third pass, so only the time for the demonstration affected the results here. Since there was no template given for the movement demonstration, measured times here only reflect the effect of users getting used to the program and can't really be the base for any assumptions on the absolute time needed for this modeling process.

	Group A	Group B	Average
GUI			
Content	1.25	1.75	1.5
Structuring	1.25	1.5	1.375
Hardware			
Ergonomics	1.25	1.75	1.5
Integration	1.00	1.75	1.375
Intuitivity			
GUI	1.00	1.5	1.25
Hardware	1.00	1.25	1.125

Table 3: Overall scores of the modeling process "movement restrictions" (1=very good, 6=very bad)

		Pass [mins]		
Group	**Candidate**	**1**	**2**	**3**
A	1	4	1	1
A	2	3	1	1
A	3	4	1	1
A	4	4	1	1
	avg.	3.75	2	2
B	1	5	1	1
B	2	4	1	1
B	3	4	1	1
B	4	2	1	1
	avg.	7	4.25	2.25

Table 4: Time needed for modeling movement restrictions

The developed input mode and GUI showed themselves to be very successful. Subjects greatly appreciated the wizard-like structuring and the accurate way of demonstrating movement with the object. Improvement to the system could be made by providing a short animation of the desired movement demonstration as a guide to the user, which would also allow for a better comparability of different user's results.

6. CONCLUSIONS

We presented two approaches for intuitive interactive modeling of special object attributes by use of specific sensoric hardware. Motivated by the task of modeling background knowledge for a humanoid robot, important object attributes were derived. Two of these were chosen for exemplatory implementation, namely stable object positions and movement restrictions for objects. For each of the two, an appropriate methods for interactive modeling was explored, focusing on intuitivity, highest possible grade of interactivity and ease of use. The evaluation with a group of test persons indicates that the chosen methods incorporate all of these features to a high degree.

In future work, we plan to implement additional modeling methods for most of the other derived object attributes. They require different interaction approaches to satisfy the same standards in intuitivity, interaction and ease of use.

Furthermore these new methods and specifically adapted versions of the existing approaches shall be used in different areas as well. Currently the application to model movement restrictions is altered to accomodate the needs which arise with robot grasp planning. Sensor data shall be used to record trajectories and accelerations as input to an automatic grasp planning mechanism.

7. ACKNOWLEDGEMENTS

This work was funded as part of the German Collaborative Research Center 588 “Humanoid Robots – Learning and Cooperating Multimodal Robots” by the Deutsche Forschungsgemeinschaft (DFG), and as part of the research project DESIRE by the German Federal Ministry of Education and Research (BMBF) under grant no. 01/ME01E.

8. REFERENCES

[1] J. Allen, D. Byron, M. Dzikovska, G. Ferguson, and L. Galescu. Towards conversational human-computer interaction, 2001.

[2] R. Becher, I. Boesnach, P. Steinhaus, and R. Dillmann. From subjects to objects and back – combining human motions and object properties to understand user actions. In *Proceedings of the Human-Centered Robotic Systems (HCRS)*, München, Germany, Oktober 2006.

[3] R. Becher, P. Steinhaus, R. Zöllner, and R. Dillmann. Design and implementation of an interactive object modelling system. In *Robotik/ISR*, München, Germany, Mai 2006.

[4] T. E. Hutchinson, K. W. Jr., W. Martin, and K. Reichert. Human-computer interaction using eye-gaze input. *IEEE Transactions on Systems, Man and Cybernetics*, 19:1527–1534, November/Dezember 1989.

[5] R. J. Jacob. The use of eye movements in human-computer interaction techniques: What you look at is what you get. *ACM Transactions on Information Systems*, 9:152–169, April 1991.

[6] T. H. Massie and J. Salisbury. The phantom haptic interface: A device for probing virtual objects. In *Proceedings of the ASME Winter Annual Meeting*. SensAble Technologies, November 1994.

[7] K. Murakami and H. Taguchi. Gesture recognition using recurrent neural networks. In *CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 237–242, New York, NY, USA, 1991. ACM Press.

[8] B. A. Myers. A brief history of human computer interaction technology. *ACM interactions*, 5(2):44–54, March 1998.

[9] P. Petridis, K. Mania, D. Pletinckx, and M. White. Usability evaluation of the epoch multimodal user interface: designing 3d tangible interactions. In *VRST '06: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 116–122, New York, NY, USA, 2006. ACM Press.

[10] T. Poston and L. Serra. Dextrous virtual work. *Commun. ACM*, 39(5):37–45, 1996.

[11] I. Poupyrev, M. Billinghurst, S. Weghorst, and T. Ichikawa. The go-go interaction technique: non-linear mapping for direct manipulation in vr. In *UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 79–80, New York, NY, USA, 1996. ACM Press.

[12] C. Ramstein and V. Hayward. The pantograph: a large workspace haptic device for a multi-modal human-computer interaction. *Conference Companion CHI 94*, April 1994.

[13] J. Rehg and T. Kanade. Digiteyes: Vision-based hand tracking for human-computer interaction. In *Proceedings of the workshop on Motion of Non-Rigid and Articulated Bodies*, pages 16–24, November 1994.

[14] B. Shneiderman. The limits of speech recognition. *Communications of the ACM*, 43(9):63–65, 2000.

[15] A. Simon and M. Doulis. Noyo: 6dof elastic rate control for virtual environments. In *VRST '04: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 178–181, New York, NY, USA, 2004. ACM Press.

[16] R. Steffan, T. Kuhlen, and F. Broicher. Design of multimodal feedback mechanisms for interactive 3d object manipulation. In *HCI (1)*, pages 461–465, 1999.

[17] C. von Hardenberg and F. Bérard. Bare-hand human-computer interaction. In *PUI '01: Proceedings of the 2001 workshop on Perceptive user interfaces*, pages 1–8, New York, NY, USA, 2001. ACM Press.