Self-Valuing Learning and Generalization with Application in Visually Guided Grasping of Complex Objects[*]

Jianwei ZHANG and Bernd RÖSSLER

Technical Aspects of Multimodal Systems (TAMS),
Department of Computer Science, University of Hamburg, D-22527 Hamburg, Germany

We present a self-valuing learning technique which is capable of learning how to grasp unfamiliar objects and generalize the learned abilities. The learning system consists of two components which distinguish between local and global quality criteria for grasp points. The local criteria are not object-specific while the global criteria cover physical properties of each object. In this case we present a generalization method of the learning parameters based on a tree distance model for the medial axis transformations. The system is self-valuing, i.e. it rates its actions by evaluating sensory information and the usage of image processing techniques. This learning system has been implemented in a real robot assembly system equipped with hand-cameras and force/torque sensors. Both the theory and the experiments have shown it ability to grasp a wide range of objects and to apply pre-learned knowledge to new objects.

1. Introduction

Robot learning is an area of interest to many scientific fields such as robotics, artificial intelligence, neural and cognitive science. The challenges of robot learning are to bypass the problems faced by classical programming techniques leading either to solving of toy problems (no scale up) or to the pre-wired design of most skills and behaviors without any possibility to learn from new experiences and to generalize. New applications emerge like service robots, pet robots or humanoid robots. The success of these new applications will mainly lay in their ability to learn from their environment and to interact with humans.

While reinforcement learning is usually a time-consuming process, Programming by Demonstration provides a straightforward and fast approach to implement sensor-based tasks. For both approaches, however, the learned skills need appropriate representations for generalization and possibilities for further improvements by the robot itself. With this background, our work aims at developing a self-valuing scheme and representation methods in the context of visually guided grasping.

In a wide range of robotic systems grasping is a basic skill that is crucial to manipulation tasks and interaction with the environment. In most industrial applications the problem of grasping is solved via *teaching-by-doing* or static programs. However, when thinking of recent research fields, e.g. service robots or humanoids, aspects of sensor-based grasping will play a very important role. New techniques are required for the robots to operate in uncharted and unknown

[*]Journal of Robotics and Autonomous Systems, 47: 117–127, 2004.

territories. For this purpose, elements of human learning abilities, e.g. the human potential of generalization, are helpful when constructing a robotic grasping system.

2. Related Research

A lot of work has been done in the field of robot grasping. [1] gives a brief overview of the field over the last two decades. The commonly used analytical approaches try to compute optimal grips according to special heuristics (e.g. [2] and [3]). In these cases either a fully specified model of the object and its mass distribution is known or the center of area of the object, extracted via image processing, is used to approximate the real center of gravity. The first case is very difficult to obtain via external sensors and without any a priori knowledge. A complete 3D-representation of the object via image processing is required and additionally features like the material of the object need to be examined. However, a hidden internal inhomogeneous mass distribution can never be discovered with such an approach. The latter case of using the center of the object's area is certainly only an approximation. This would work if the center of gravity coincides with the object's center of area, but would not work in case of inhomogeneity either.
Among several attempts to handle the problem of learning how to grasp, [4] presents a system that learns how to grasp objects with a parallel-jaw gripper. Two main subproblems are learned: to choose grasping points and to predict the quality of a given grasp. The disadvantage of this system is that only *local* criteria are used to store grasping configurations. Without *global* criteria it is for example impossible to learn how to grasp an object whose center of gravity does not coincide with the center of its image area. Without self-valuing learning techniques it is impossible to handle the real physical properties of an object. [5] presents a learning system for visually guided grasping, constructed of two learners. This system is not self-valuing, i.e. the optimal grasp point has to be given to the learner initially. Therefore, the two learners cannot adapt to new objects either. In [6] a system is developed that performs skillful grasping with a dextrous robot hand by emulating infants' growth processes. [7] described a programming system by demonstration using the segmentation of breakpoints of the motion observations and producing the robot commands to replicate the observed task. They did not consider the representation of objects to be grasped nor discussed how the learned mappings can be generalized and adapted to new objects.

3. Learning Scheme

To construct a robotic learning system, it is useful to investigate elements of human learning abilities. Our work is based on the idea that when intending to grasp an unfamiliar object with two fingers, one can mainly consider two kinds of quality criteria on how to choose optimal grasp points. These two kinds are further referred to as *local quality criteria* and *global quality criteria*, or *local/global criteria* for short. Together they form the design basis for the underlying learning system.

3.1. Local and Global Quality Criteria

Local quality criteria: The local quality criteria are mostly independent of a special shape and therefore of global aspects like the distribution of an object's mass. Therefore, they are valid for any kind of object. Local quality criteria are considered first when one decides to grasp an unfamiliar object. Such a criterion can be estimated for example by the sliding friction between an object and the fingers of the gripper.

Global quality criteria: Global quality criteria, by contrast to the local ones, are closely connected to a special object and therefore seldom significant for other objects. They are applied after the local criteria to find the optimal grasp point. These criteria consider aspects like the distribution of mass of an object, e.g. the torque at the gripper when grasping an object.

Technically speaking, the local criteria define a virtual axis on which a possibly good grasp point may be found with the help of the global criteria. For example, the grasp configuration computed in Fig. 1(b) could be determined from the search axis proposed by grasp point 3 in Fig. 1(a). In the learning process these criteria are repeatedly considered one after the other for a finite number of steps until a good grasp point is found. The number of steps varies with the skill of the learner and the structure of the object. For a familiar type of object, the global and local criteria are considered in only one step. In fact, since the same local criteria can be applied to any kind of object, as mentioned above, they are fully learned prior to the global criteria.

3.2. Optimality Conditions

A grasp point is considered to be optimal, if the local quality criteria are met in the following fashion:

- the fingers can cover the object at this grasp point, and
- no sliding friction occurs between the fingers and the object.

It is considered to be optimal according to the global quality criteria if:

- no torque occurs between the fingers grasping the object, and
- the grasp is stable, i.e. the object does not slip between or out of the fingers.

Some sample grasp configurations are shown in Fig. 1.

3.3. Higher Level Criteria

An additional and *higher level criterion* for human grasps is the purpose of the grip, i.e. its importance for carrying out further operations, e.g. grasping a cup at its bail in order to drink something or a sledge at its handle to drive a nail into a wall[2]. Other higher level criteria are for example the material or surface of an object. To consider these criteria additional sensors or sophisticated image processing techniques need to be integrated, which is beyond the scope of this work. Our objective is to emulate the abilities of young children who just wants to learn to get hold of an object as good as possible.

4. Two-Learner System

The quality criteria mentioned above suggest a system consisting of two learners, one for the local and the other for the global quality criteria. The states for the first learner only consist

[2]For this higher level criterion, aspects of optimality like reducing torque must possibly be shelved.

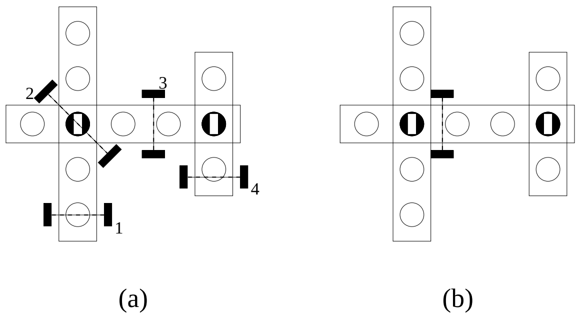


Figure 1. Optimal grasp points in terms of the local and global quality criteria. (a): some example grasp configurations which are optimal according to the local quality criteria. (b): the grasp point is optimal according to both criteria.

of the local features $s = (f_{l_1}, \ldots, f_{l_m})$. The learner tries to map them to actions consisting of a rotational component $a = \phi$. The second learner tries to map states of global features $s = (f_{g_1}, \ldots, f_{g_n})$ to actions of translational components: $a = (x, y)$. Because the local criteria are mainly covered by the relative orientation of the gripper, the responsible learner is called *orientation learner*. The global criteria are affected by the position of the grasp point in the object and therefore the proper learner is further referred to as *position learner*. These two learners operate right after each other (Algorithm 1).

Algorithm 1 Algorithm for learning an optimal grasp point

```
choose an initial grasp point configuration
$steps \leftarrow 0$
**repeat**
  $steps \leftarrow steps + 1$
  **repeat**
    learn with the orientation learner
  **until** [the grasp point is optimal according to orientation OR number of episodes exceeds a given value]
  **repeat**
    learn with the position learner
  **until** [the grasp point is optimal according to the position in the object OR number of episodes exceeds a given value]
**until** [the optimal grasp point is found OR $steps > steps_{max}$]
```

The key element for a good performance of such a learning system is the choice of adequate features which reflect the local and global quality criteria. The choice depends mainly on the kind of gripper and other available sensors used in the system. The local and global features are shown in Fig. 2. The first component of the state vector of the orientation learner is the length L of the grasp-line. With this feature, good grasp points are distinguished from those which are inadequate because the gripper cannot cover the object. The remaining features are the corresponding angles $\Theta_1, \ldots, \Theta_4$ between the grasp-line and the flanking straight line segments gained by a simple contour tracking process. The features for the position learner are the distance D between the center of the grasp-line and the center of area of the object's image as well as the torque T around the normal vector $\vec{n}$ of the gripper. Thanks to the learner separation, the local criteria need not be learned for every new object. Therefore, the orientation learner is a universal learner, which means that the same learner can be used for every object. E.g., this learner can learn to grasp objects at opposite parallel or concave edges.

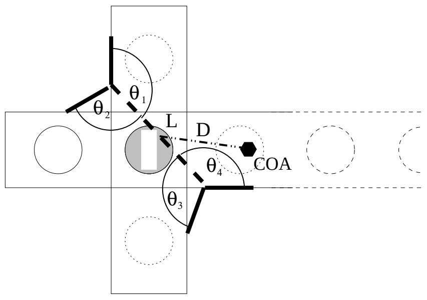


Figure 2. State coding for the learners. The orientation learner uses the length L and the angles $\Theta_1, \ldots, \Theta_4$ while the position learner integrates the distance D between the center of the grasp-line and the center of area of the object's image.

In principle, a two-learner system design was first introduced in [5]. The new aspect of this work is a different usage of the two learners. As described above, this design was chosen with respect to the local and global criteria and their generalization properties.

4.1. Self-Valuation

The presented system is self-valuing, a method to automatically gain an estimation for the learning algorithms. Self-valuation is done via a force/torque sensor and the hand-camera at the gripper of the robot. It is important to mention that no optimal grasp point is pre-known and the system finds its own grasp points taking into account the quality criteria.

4.1.1. Orientation Learner

The best estimation of a good grasp, determined by the orientation learner, is obtained by the second optimality condition, i.e. no sliding friction at the fingers of the gripper. When an object slips between or out of the fingers at the moment of closing the gripper, the selected grasp configuration was not optimal according to the local criteria. Some existing systems (e.g. [2]) try to determine the friction occurring within the gripper analytically, i.e. by computing the friction cone via geometrical features. Here, several grasp configurations are tried out with the real robot that values the success or failure of the performed grasps – like humans who do not analytically compute their optimal grips, but learn by success and failure. Because the gripper used in this work does not slip like human fingers over the object's surface, sliding friction appears either as a rotation or as a displacement of the object itself. The valuation of the orientation learner is basically gained by image processing. A penalty for self valuation is computed as follows:

$$P = \begin{cases} -(\Theta_{\text{diff}} + \mathcal{D}_{\text{diff}}) & \text{if the grip was successful} \\ -\mathcal{P}_{\text{const}} & \text{otherwise} \end{cases}$$

where Θ_{diff} is the angle between the initial and the least inertia axis after the performed grasp, D_{diff} the displacement of the center of area and P_{const} a high constant penalty which is greater than the highest estimated changes in orientation and position together. The first two pictures of Fig. 3 show a grasp configuration which results in a rotation of the object itself. If a grasp has totally failed and therefore the first optimality condition cannot be met[3], a pre-defined penalty is given.

4.1.2. Position Learner

While the valuation technique for the orientation learner is primarily based on processing images from the camera sensors, the self-valuation of the position learner is primarily gained via a force/torque sensor. The two points for optimality of the global quality criteria, mentioned

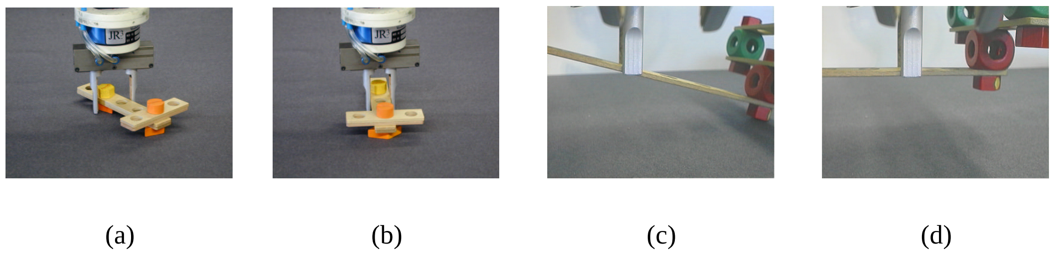


Figure 3. Sample grips that are suboptimal according to the quality criteria.

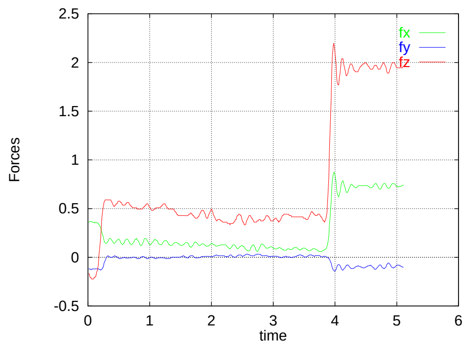
in Section 3.2, are taken into account for self-valuation in the following fashion:

Stable grasp: A grip is unstable, for example, if the gripper is not strong enough to fix the object at a given position. Such a situation is shown in Fig. 3(c). This lift-up movement of the manipulator results in forces shown in Fig. 4(a). Nearly during the whole lift-up movement, the force in the direction of the approach vector of the gripper is approximately constant. At the moment when the object loses contact with the table (in this example at $4s$) the force rises to a higher value. These profiles can be analyzed and used for valuation of the learner, e.g. this situation is valued with a pre-defined high penalty to express that such grips are not desired.

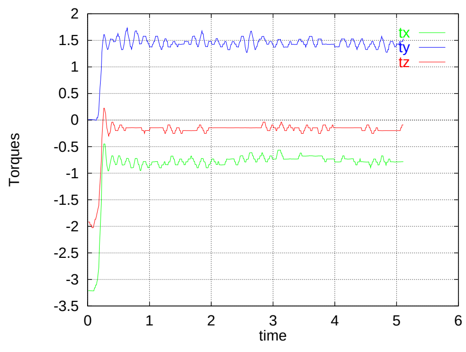
[3]This occurs either when the orientation of the gripper does not permit covering the object or the object slips out of the fingers while closing them.

When an object slips out of the fingers of the gripper the force in the direction of the approach vector of the gripper suddenly reduces to zero and the grasp can be considered a failure. Such a grasp is totally undesirable. Therefore, a constant high penalty is given to prevent the system from effecting such grips in the future.

Reducing torque: The goal of the position learner is to reduce torque within the fingers of the gripper. Fig. 3(d) shows an example of a grip that produces high torque. The torque profiles are shown in Fig. 4(b). Immediately after the beginning of the lift-up process, the torque around the normal vector of the gripper rises to a value considerably bigger than zero and stays constant while the object is being held. Torques are computed, and their negative values are directly used in the position learner. Here, no constant penalty is given because a grip with large torque is not necessarily bad, i.e. the system must have the possibility to distinguish between grasp points with different torques and choose the best among them.



(a)



(b)

Figure 4. Force profiles when the grip is not stable as shown in Fig. 3(c) and torque profiles of the grip in Fig. 3(d).

5. Implementation of Learning

The learning scheme in our system is based on the *Sarsa* algorithm [8]. The general update formula computes the difference between the current and the next prediction of cumulative reward and updates the action-value function Q by a fraction of this difference as follows:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right]$$

As seen above, our system must handle continuous states and actions, i.e. angles, torques, etc. In such a situation we cannot provide a single value Q for every state-action pair but rather have to use a function approximator. Such a function approximator is of the form $Q_\omega(s, a)$, where

$\omega = (\omega(1), \omega(2), \ldots, \omega(n))^T$ is a set of adjustable weights. The update of the current estimate of Q is performed by modifying the weights according to the following rule:

(1) $\quad \Delta\omega_t = \alpha \left[r_{t+1} + \gamma Q_{t+1} - Q_t \right] \nabla_{\omega_t} Q_t$

A general advantage of function approximators is that they are able to *generalize*. The system is able to estimate the expected return of state-action pairs that were never visited before. Although a function approximator can deal with continuous state and action spaces, it may not be able to accurately represent Q for the entire state and action space due to its finite resources. We employ the B-spline function approximator [9] for the Q-function which is a natural generalization of coarse coding to continuously-valued features.

5.1. Approximating the Action-Value Function

In the following we define $\vec{x}$ as the concatenation of the current state $s = (s_1, \ldots, s_l)$ and the taken action $a = (a_1, \ldots, a_m)$, that is: $\vec{x} = (s_1, \ldots, s_l, a_1, \ldots, a_m)$. The output for the B-spline function approximator which is the prediction of Q is computed by:

$$\begin{aligned} Q(\vec{x}) &= \frac{\sum_{i_1=1}^{l_1} \cdots \sum_{i_n=1}^{l_n} \left(c_{i_1,\ldots,i_n} \prod_{j=1}^{n} N_{i_j,k_j}^{j}(x_j) \right)}{\sum_{i_1=1}^{l_1} \cdots \sum_{i_n=1}^{l_n} \prod_{j=1}^{n} N_{i_j,k_j}^{j}(x_j)} \\ &= \sum_{i_1=1}^{l_1} \cdots \sum_{i_m=1}^{l_n} \left(c_{i_1,\ldots,i_n} \prod_{j=1}^{n} N_{i_j,k_j}^{j}(x_j) \right) \end{aligned}$$

Because the introduced weights ω of Q here correspond to the control vertices $c_{i_1,i_2,\ldots,i_n}$ of the B-spline function approximator, the gradient of Q with respect to ω from Eqn. (1) is:

$$\nabla_\omega Q = \nabla_{c_{i_1,i_2,\ldots,i_n}} Q$$

Now the learning update from Eqn. (1) turns into the following formula:

$$\Delta c_{i_1,i_2,\ldots,i_n} = \alpha \left[r_{t+1} + \gamma Q_{t+1} - Q_t \right] \prod_{j=1}^{n} N_{i_j,k_j}^{j}(x_j)$$

Based on this, the control vertices are updated online after each grasping trial of the system.

5.2. Accumulating Trails

A practical problem to be considered is that the system will learn a path from an initial state, i.e initial grasping configuration, up to a final state, i.e. a successful grip. To overcome this side effect in systems where the goal state itself is the most important outcome and not the path to the goal state, we propose an approach to increase the performance of such a learning system, called *accumulating trails*. When a learning system learns a type of path from an initial state to a final state, i.e. by applying a set of actions $a_0, \ldots a_n$ to s and its successors, it is sometimes possible to get to the same goal state by applying a set of actions $a'_0\ {}_m\ .\ a'$ to the state s and its successors, where $m < n$. That is to say, one would reach the goal state $n - m$ steps earlier.

Let ψ denote the function applying an action a to a state s, denoted $\psi : \mathcal{A} \to (\mathcal{S} \to \mathcal{S})$, where $\mathcal{A}, \mathcal{S}$ are the total sets of actions and states, respectively. The outcome of this function, when applied to an action, is a function on the state space $\mathcal{S}$ called *action execution function*.

Using the definition above, each learning episode can be considered as a composition of functions ($A : \mathcal{S} \to \mathcal{S}$)

$$A(s) = \psi(a_n) \circ \psi(a_{n-1}) \circ \cdots \circ \psi(a_0)(s)$$

where s is the starting state of the episode and a_i is the action applied in time step i. This function composition is further referred to as *sequence*.

A sequence B of action executions $\psi(b_m) \circ \cdots \circ \psi(b_0)$ is called a *sub-sequence* of sequence $A = \psi(a_n) \circ \cdots \circ \psi(a_0)$, if $A(s) = B(s)$:

$$\psi(a_n) \circ \cdots \circ \psi(a_0)(s) = \psi(b_m) \circ \cdots \circ \psi(b_0)(s),\ m \leq n$$

where s is the starting state. Then, sequence A is called *substitutable* through B. The sub-sequence B always produces the same resulting state as the sequence A. That means, if we start in state s it makes no difference whether we “follow” sequence B or sequence A. The state at the end of the sequence is always the same. If a sequence A is not substitutable by any other sequence B, it is called *final*. When the agent’s intention is to reach the goal states as soon as possible, as for example in this work[4], the learning algorithm should converge to a situation of purely final sequences.

An accumulation function on action executions is defined as: $\oplus : (\mathcal{S} \to \mathcal{S}) \times (\mathcal{S} \to \mathcal{S}) \to (\mathcal{S} \to \mathcal{S})$. A sequence $A = \psi(a_n) \circ \cdots \circ \psi(a_0)$ of action executions is *accumulatable*, if

$$\psi(a_n) \oplus \psi(a_{n-1}) \oplus \cdots \oplus \psi(a_0) = B,$$

where B is the subsequence of A. The accumulation function describes how to combine action executions to produce shorter sequences. This function has to be defined according to the learning system one wants to develop. The accumulation is defined for action executions and not solely for actions, because it depends on the states if such an accumulation can be performed. In some situations the accumulation function is defined as follows:

$$\psi(a_k) \oplus \psi(a_l) = \psi(a_k \diamond a_l), \tag{2}$$

where $\diamond$ is a function $\diamond : \mathcal{A} \times \mathcal{A} \to \mathcal{A}$.

In most situations, the accumulation function must include a kind of model of the environment and this is only possible by also taking into account the states rather than only the actions as suggested by Eqn. (2). The agent must “know” in which situation it is possible to accumulate action executions and in which situation it is not. However, for some tasks Eqn. (2) is an intuitive and sufficient definition.

As an example, for application within the orientation learner the accumulation function $\diamond$ is defined as:

$$a_k \diamond a_l = \begin{cases} a_k + a_l & \text{if} \quad -90 \leq a_k + a_l \leq 90 \\ a_k + a_l + 180 & \text{if} \quad a_k + a_l < -90 \\ a_k + a_l - 180 & \text{if} \quad a_k + a_l > 90 \end{cases}$$

assuming that the actions of the orientation learner are rotational movements in the interval $[-90, \ldots, 90]$.

[4]It is desirable to find an optimal grasp point as soon as possible.

6. Generalization

The orientation learner is fully applicable to any kind of object, i.e. it provides a total generalization potential. In other work, where the learning process is not divided into two separate learners, the generalization is only partial. This results in slower learning phases for new objects. Propositions like “grasping at parallel edges is always good” cannot be made by such systems at all. Here, once the orientation learner has learned several grasping situations, it can be used with any kind of new object the robot is faced with.

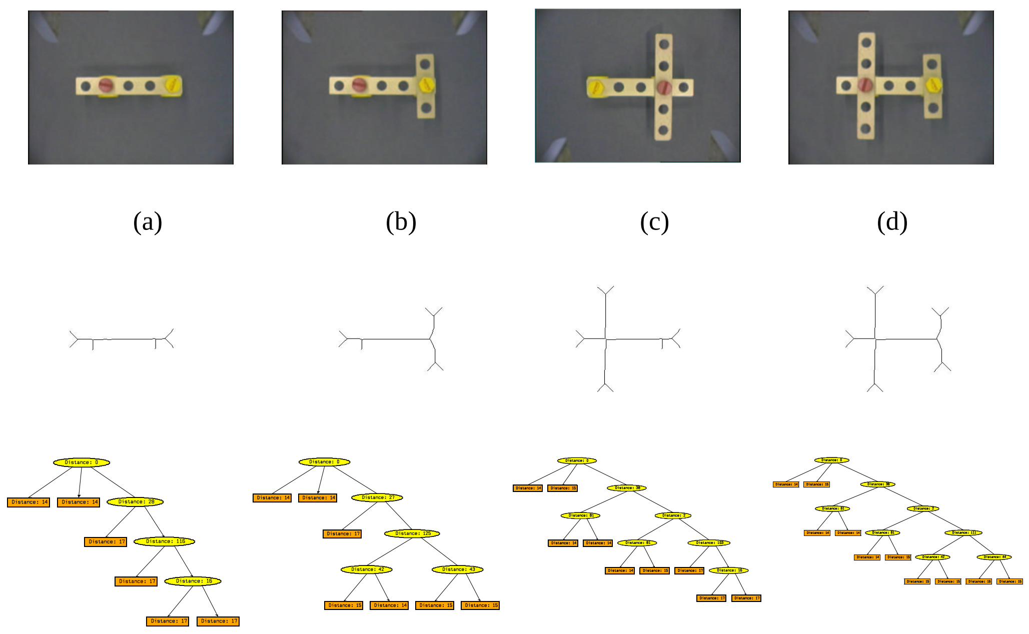


Figure 5. Different objects, their medial axes and the resulting tree coding. At each node the length of the branch leading to that node is stored.

By comparison, the position learner is more complicated. Because of different shapes of objects the global positions of the learned grasp points cannot be applied to every object. In most cases it is a better choice to initialize the learning parameters by values of a pre-learned similar object than a random initialization.

Three main subproblems must be solved:

A. In which situation can a pre-learned position learner be fully adopted to a new object?
B. When can a pre-learned position learner be used as a basis for a new object?
C. When must a completely new position learner be initiated?

6.1. Tree Distance as a Measure

Assuming a distance measure on the objects to grasp, we can improve our initial learning parameters as follows: Let $\mathcal{L}_s = \{(o_i, l_i) | i = 1 \ldots n\}$ be the set of tuples of n pre-stored objects

o_i in tree notation, together with their stored position learners l_i, and $dist(o_i, o_k)$ the distance of the trees according to a distance measure. Then,

A a pre-learned position learner l' of an object o' can be fully adopted to a new object o, if $\forall(o_i, l_i) \in \mathcal{L}_s \backslash (o', l')$:

(3) $\quad dist(o', o) \leq dist(o_i, o) \leq D_{min};$

B a pre-learned position learner l' of an object o' can be used as basis for a new object o , if $\forall(o_i, l_i) \in \mathcal{L}_s \backslash (o', l')$:

(4) $\quad D_{max} \geq dist(o', o) \leq dist(o_i, o) > D_{min};$

C a completely new position learner is initiated for a new object o, if $\forall(o_i, l_i) \in \mathcal{L}_s$

(5) $\quad dist(o_i, o) > D_{max},$

where D_{max} and D_{min} are adequate thresholds for accepting and refusing an object to be equal, respectively.

Simple distances on the objects' pixel data, like the *Hamming Distance,* are susceptible to noisy data and moreover do not consider the object structure. We restrict our observations to object structures that can be represented as a hierarchical relation. A simple tree encoding is used for the objects. These are rooted ordered trees, i.e. there exists an ancestor relation of nodes and the order of sibling nodes matters. In order to obtain a tree out of an object's image pixel data a medial axis transformation [10] is performed. The resulting graph is analyzed by a contour tracking process which transforms the medial axes into the corresponding trees. Fig. 5 gives examples of this process. The nodes of the shown trees contain the length information about the corresponding parts of the medial axis.

6.2. Computational Results

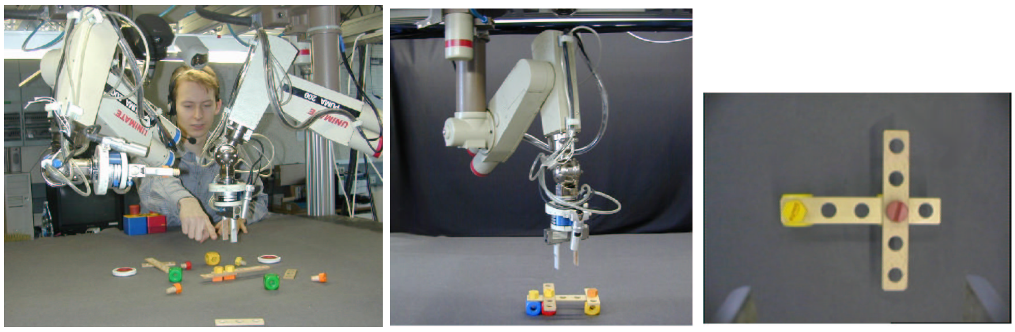
Distance models on rooted ordered trees based on editing operations are proposed in [11,12]. We used the latter for our experiments. For instance, we learn to grasp the object shown in Fig. 5(a) without any a priori knowledge. Accordingly, a tree comparison with the object in Fig. 5(b) leads to Eqn. (4). For the object in Fig. 5(c) no object of sufficient similarity is stored in the database which leads to Eqn. (5). Finally, Eqn. (4) again suggests to grasp the object in Fig. 5(d) using the parameters of the object in Fig. 5(c).

In this manner one can determine the "most similar" object out of the set of pre-learned aggregates for a new object. If the similarity is strong enough, i.e. Eqn. (3) is met, the objects are treated the same and the same position learner is used.

7. Experiments and Results

The physical set-up of this system consists of the following components:

Main actuator and sensors: In our integrated experimental scenario, a number of assembly parts must be recognized, manipulated and built together to construct the model aircraft [13]. In each of these steps, a human communicator instructs the robot, which implies that the interaction between them plays an important role in the whole process (Fig. 6(a)). The 6-DOF PUMA-260 manipulators are installed overhead in a stationary assembly cell. On the wrist of the manipulator, a robot gripper with integrated force/torque sensor and "self-viewing" hand-eye system (local sensors) is mounted (Fig. 6(b), (c)). The robot is controlled by RCCL (*Robot Control C Library*).



(a) The scenario of programming by gestures and instructions

(b) The robot arm learning to grasp

(c) A view of the hand-camera

Figure 6. The experimental setup (a), (b) and a sample view of the hand-camera (c).

Objects: Most kinds of objects are constructed from *Baufix* elements, wooden toys for children containing parts like screws, ledges and cubes. Therefore, these objects are also referred to as *aggregates*. An advantage of these parts is that one can very quickly construct new aggregates that can be tested with the system.

To get a uniform and matchable view of the objects the system learns to grasp, the manipulator initially moves over the object so that the x-axis of the camera's coordinate system appears parallel to the axis of least inertia of the object and the center of area on the right side of the image. The center of the object's bounding box coincides with the center of the image. An additional tool-transformation is performed so that the camera is moved towards the working surface. Several objects were used to test the performance of the whole system. Some of them are shown in Fig. 7. The robot has found a good and stable grasp point for each object that fulfills the optimality conditions given above, often near the object's center of gravity. Two special results of a grasping operation are shown in Fig. 8. In Fig. 8(a) the manipulator has grasped the object at a point different from the center of area but near the center of mass of the object. Fig. 8(b) shows a successful grasp at a convex edge of a different object. To show the generalization ability of the orientation learner, it was first applied to a new object for a defined number of epoches. Thereafter, the same learner was used on a different object to show that the average steps until the goal state decrease much faster. The result is shown in Fig. 9. In the second

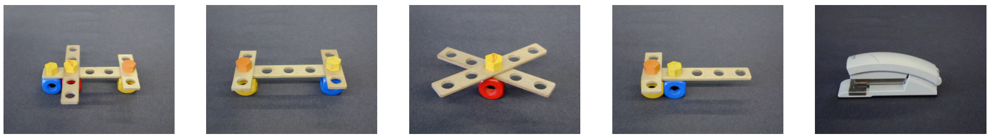


Figure 7. Sample objects for grasp learning.

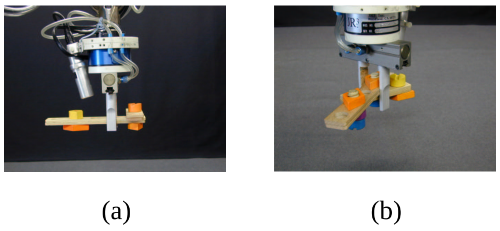


Figure 8. Successfully performed grasping operations.

part of the experiment the orientation learner did not start at a value of three where the initially performed orientation learner ended. This is due to the fact that in the first cycle a simple ledge was used and the learner still did not converge while in the second cycle a more complex object was used. However, one can see that in the second cycle the orientation learner was quicker. Only new states that do not occur on the simple ledge have to be learned additionally.

8. Discussion and Ongoing Work

We presented a self-valuing learning system that is capable of grasping various kinds of objects. Our system consists of two learners based on local and global quality criteria. While the orientation learner is applicable to arbitrary objects and therefore fully generalizes between them, the position learner is mostly dependent on a special object and its physical properties. The generalization of the position learner is accomplished by a tree distance model on the shape of the object. The system shows the ability to grasp several kinds of objects and to generalize the learned faculties to new ones.

In our ongoing work, the system will be extended to handle grips in 3D. Therefore the tree coding for generalization has to be extended to represent the 3D-structure of an object. With additional sensors, e.g. a stereo camera vision system, the robot should examine the objects and place the grips from different orientations in space. We are also adapting the presented system to a multi-fingered robot hand. In this case a grasp point no longer consists of only two contact points on the object's surface. The threefingered hand, as shown in Fig. 10, can e.g. perform a full 7 point form closure grasp. Furthermore, the possible actions of the learners are more challenging with a multi-fingered hand. The different fingers can move independently to a certain extent and apply different forces. However, the basic principle of two learners, based

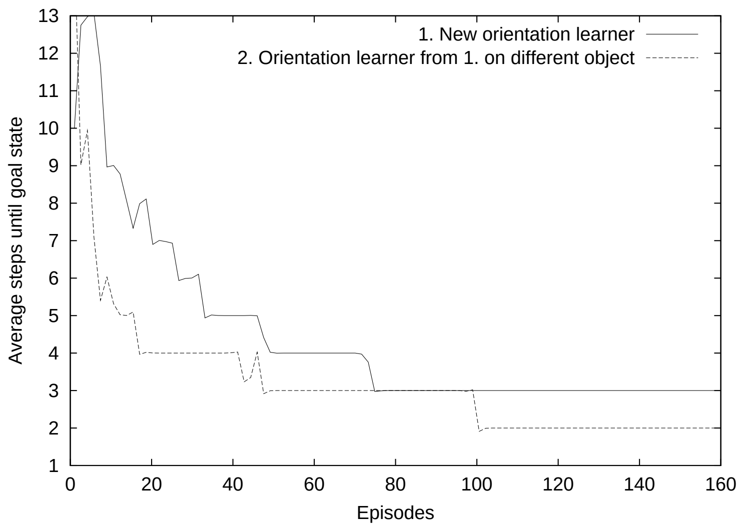


Figure 9. Generalization of the orientation learner.

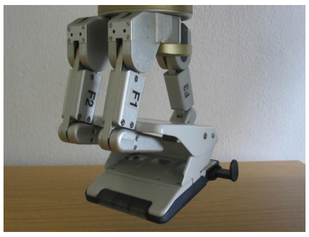


Figure 10. Sample grip with a multi-fingered hand.

on local and global criteria, and the self-valuing approach can be maintained.

REFERENCES

1. A. Bicchi, V. Kumar, Robotic grasping and contact: A review, in: Proceedings of the IEEE International Conference on Robotics and Automation, 2000, pp. 348–353.
2. G. Smith, E. Lee, K. Goldberg, K. Boehringer, J. Craig, Computing parallel-jaw grips, in: Proceedings of the IEEE International Conference on Robotics and Automation, 1999, pp. 1897–1903.
3. P. J. S. A. Morales, G. Recatalá, A. P. del Pobil, Heuristic vision-based computation of planar antipodal grasps on unknown objects, in: Proceedings of the IEEE International Conference on Robotics and Automation, 2001, pp. 583–588.
4. I. Kamon, T. Flash, S. Edelman, Learning visually guided grasping: A test case in sensorimotor learning, IEEE Transactions on System, Man and Cybernetics 28 (3) (1998) 266–

276.

5. J. Zhang, G. Brinkschröder, A. Knoll, Visuelles reinforcement-lernen zur feinpositionierung eines roboterarms ”uber kompakte zustandskodierung, in: Tagungsband *Autonome Mobile Robotersysteme (im Druck)*, München, 1999.
6. J. Coelho, J. Piater, R. Grupen, Developing haptic and visual perceptual categories for reaching and grasping with a humanoid robot (2000).
7. S. Kang, K. Ikeuchi, Toward automatic robot instruction from perception - temporal segmentation of tasks from human hand motion, IEEE Trans on Robotics and Automation 11 (5) (1995) 670 – 681.
8. S. P. Singh, R. S. Sutton, Reinforcement learning with replacing eligibility traces, Machine Learning 22 (1–3) (1996) 123–158.
9. J. Zhang, A. Knoll, Constructing fuzzy controllers with B-spline models - principles and applications, International Journal of Intelligent Systems 13 (2/3) (1998) 257–285.
10. H. Blum, A transformation for extracting new descriptors of shape, in: W. Wathen-Dunn (Ed.), Models for the Perception of Speech and Visual Form, M.I.T. Press, Cambridge, MA, 1967, pp. 362–380.
11. K. C. Tai, The tree-to-tree correction problem, J. Assoc. Comput. Mach. 26 (3) (1979) 422–433.
12. L. W. T. Jiang, K. Zhang, Alignment of Trees - An Alternative to Tree Edit, Theoretical Computer Science 143 (1) (1995) 137–148.
13. J. Zhang, A. Knoll, A two-arm situated artificial communicator for interactive assembly, IEEE Transactions on Industrial Electronics 50 (4) (2003) 651–658.