Classifier Independent Viewpoint Selection for 3–D Object Recognition

F. Deinzer[1,⋆], J. Denzler[1,2], H. Niemann[1]

[1] Lehrstuhl für Musterkennung, Universität Erlangen–Nürnberg
[2] Computer Science Department, University of Rochester, USA
deinzer@informatik.uni-erlangen.de

Abstract 3–D object recognition has been tackled by passive approaches in the past. This means that based on one image a decision for a certain class and pose must be made or the image must be rejected. This neglects the fact that some other views might exist, which allow for a more reliable classification. This situation especially arises if certain views of or between objects are ambiguous.
In this paper we present a classifier independent approach to solve the problem of choosing optimals views (viewpoint selection) for 3–D object recognition. We formally define the selection of additional views as an optimization problem and we show how to use reinforcement learning for continuous viewpoint training and selection without user interaction. The main focus lies on the automatic configuration of the system, the classifier independent approach and the continuous representation of the 3–D space.
The experimental results show that this approach is well suited to distinguish and recognize similar looking objects in 3–D by taking a minimum amount of views.

1 Motivation

The results of 3–D Object classification and localization depend – as matter of course – strongly on the images which have been taken of the object. Based on ambiguities between objects in the data set some views might result in better some other in worse results. For difficult data sets usually more than one view is necessary to decide reliably for a certain object class. Problems with ambiguous views can especially be observed for objects in real world applications.

Viewpoint selection tackles exactly the problem of finding a sequence of optimal views to increase classification and localization results by avoiding ambiguous views or sequentially ruling out possible object hypotheses. The optimality is not only defined with respect to the recognition rate but also with respect to the number of views necessary to get reliable results. The number of views should be as small as possible to delimit viewpoint selection from randomly taking a large number of images.

In this paper a novel approach for viewpoint selection based on reinforcement learning is presented. The approach shows the following properties: first, the sequence of best views is learned automatically in a training step, where no user interaction is necessary. Second, the approach is classifier independent, so that an arbitrary classifier can be used. This makes it applicable for a very wide range of applications. Third, the possible viewpoints are continuous in 3–D, so that a discretization of the viewpoint space is avoided, like it has been done before, for example in the work of [2]. Actually, our approach not only allows to avoid ambiguous views. Such ambiguous views are presented in Figure 1. Since it is classifier independent, views which are difficult for a certain classifier can also be detected.

⋆ This work was partially funded by the German Science Foundation (DFG) under grant SFB 603/TP B2. Only the authors are responsible for the content.

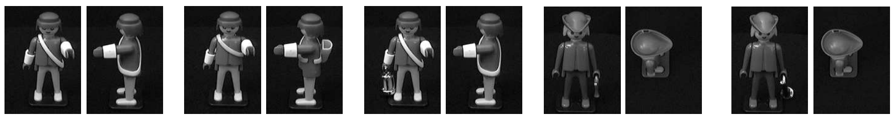
Object o1: band Object o2: quiver Object o3: lamp Object o4: gun Object o5: trumpet

Figure 1. Examples for ambiguities between objects.

Viewpoint selection has been investigated in the past in several applications. Examples are 3–D reconstruction [8] or optimal segmentation of image data [6]. In object recognition also some active approaches have already been discussed. In [2] different frameworks for handling uncertainty and decision making (probabilistic, possibilistic and Dempster–Shafer theory) have been compared with respect to viewpoint selection. But this approach can only handle discrete positions and viewpoints. The work of [11] presents an active recognition approach for which a camera must be moved around the object. But this approach is not really a viewpoint selection. The active part is the selection of a certain area of the image for feature selection. The selected part is also called receptive field [11]. Compared to our approach, no camera movement is performed neither during training nor during testing. Thus, the modeling of viewpoints in continuous 3–D space is also avoided. The work of [5, 4] tackles the viewpoint selection problem from a knowledge based point of view. They use Bayesian networks to decide for the next view to be taken. Therefore, the approach is dedicated to special recognition algorithms and to certain types of objects, for which the Bayesian network has been manually constructed. In other words, the approach is not classifier independent and cannot be applied without user interaction. Finally, an approach for 2–D viewpoint selection has been presented in [3]. In contrast to the paper presented here, the degree of freedom for the viewpoint selection is one, since the viewpoints were chosen by rotating a turntable. Also, only synthetic images have been used.

In the following we will present a formal statement of the problem and the goals of viewpoint selection in Section 2. In Section 3 we show how reinforcement learning can be used to solve the problem stated in Section 2. Also an extension of the normal discrete reinforcement learning is presented which makes it possible to model a continuous viewpoint space. Thus viewpoint selection can be defined as a continuous optimization problem. The experimental environment and results are presented and discussed in Section 4. The paper concludes with a summary and an outlook to future work in Section 5.

2 Viewpoint Selection in 3–D Object Recognition

The goal of this work is to provide a solution to the problem of optimal viewpoint selection for 3–D object recognition without making a priori assumptions about the objects and the classifier. The problem is to determine the next view of an object given a certain decision about the class and the estimated pose of that object. The problem can also be seen as the determination of a function, which maps a class and pose decision to a new viewpoint. Of course, this function should be estimated automatically during a training step. The estimation must be done by defining a criterion, which measures how useful it is to choose a certain view given a classification and localization result. Additionally, the function should take uncertainty into account in the recognition process as well as in the viewpoint selection. The latter one is important, since new views are usually taken by moving a robot arm and the final position of the robot arm will always be error–prone. Last not least, the function should be classifier independent and should handle continuous viewpoint and object pose spaces as well.

A straight forward way to formalizing the problem is given by looking at Figure 2. A closed loop between sensing $\boldsymbol{s}_t$ and acting $\boldsymbol{a}_t$ can be seen. The chosen *action* $\boldsymbol{a}_t \in \mathbb{R}^2$ corresponds to the executed camera movement, the sensed *state* $\boldsymbol{s}_t \in \{1, 2, \ldots, k\} \times \mathbb{R}^2$ is class number and pose, returned by the classifier. The pose is modeled in our work as the viewing position $(\alpha\ \beta)^T$ on a sphere. Additionally, the classifier returns a so called *reward* r_t, which measures the quality of the chosen viewpoint. For a viewpoint, where a correct decision for exactly one object class and pose is possible, the reward should have a large value. A small value will indicate that the view is ambiguous and no reliable classification and pose estimation is possible. It is worth noting that the reward might also include costs for the camera movement, so that large movements of the camera are punished. In our paper we neglect costs for camera movement for the time being.

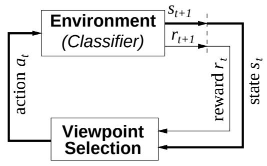

Figure 2. Principles of reinforcement learning.

At time t during the decision process, i.e. the selection of a sequence of viewpoints, the goal will be to maximize the accumulated and weighted future reward, called the *return* R_t

$$R_t = \sum_{n=0}^{\infty} \gamma^n r_{t+n+1} \text{ with weight } \gamma \in [0;1] \qquad (1)$$

The weight γ defines, how much influence a future reward at time $t + n + 1$ will have on the overall return R_t. Of course, the future rewards cannot be observed at time step t. Thus, the following function, called the *action–value function* $Q(\boldsymbol{s}, \boldsymbol{a})$

$$Q(\boldsymbol{s}, \boldsymbol{a}) = E\left\{R_t | \boldsymbol{s}_t = \boldsymbol{s}, \boldsymbol{a}_t = \boldsymbol{a}\right\} \qquad (2)$$

is defined, which describes the expected return, when starting at time step t in state $\boldsymbol{s}$ with action $\boldsymbol{a}$. In other words, the function $Q(\boldsymbol{s}, \boldsymbol{a})$ models the expected quality of the chosen camera movement $\boldsymbol{a}$ for the future, if the classifier has returned class and pose $\boldsymbol{s}$ before. This function $Q(\boldsymbol{s}, \boldsymbol{a})$ is one of the key points in reinforcement learning and will be described in the next section. Viewpoint selection can now be defined as a two step approach: First, estimate the function $Q(\boldsymbol{s}, \boldsymbol{a})$ during training. Second, if at any time the classifier returns $\boldsymbol{s}$ as classification result, select that camera movement $\boldsymbol{a}$ which maximizes $Q(\boldsymbol{s}, \boldsymbol{a})$, i.e. the expected accumulated and weighted rewards. The second step is treated by defining a so called *policy* π

$$\pi(\boldsymbol{s}) = \operatorname*{argmax}_{\boldsymbol{a}} Q(\boldsymbol{s}, \boldsymbol{a}) \qquad (3)$$

which returns the best action $\pi(\boldsymbol{s})$ to be performed while being in a state $\boldsymbol{s}$. During training, this deterministic policy is often changed to a randomized one, to make sure that all state/action pairs are evaluated [12].

It is worth noting that this approach defines a classifier independent way of viewpoint selection, since the only classifier dependent component, is the reward r_t. The two step approach described above makes no assumptions about the chosen classifier, unless the classifier must return in some way an estimate of the reliability of its result. For statistical classifier a straight forward way exists to define such a quantity. To give one example, assume the difference between the maximum a posteriori probability and the second maximum. In the case of an Eigenspace approach [7] for classification, also a natural way for defining the reward is possible. A definition can be found in Section 4.

For the estimation and learning of the function $Q(\boldsymbol{s}, \boldsymbol{a})$ reinforcement learning provides a bunch of algorithms and theoretical results on convergence. We will present one algorithm, Monte Carlo learning, in the following section.

3 Reinforcement Learning Applied to Viewpoint Selection

3.1 General Approach

In the previous section viewpoint selection has been defined as an optimization problem. The key issue of course is the estimation of the function $Q(\boldsymbol{s}, \boldsymbol{a})$ which is the basis for the decision process in equation (3). One of the demands defined in Section 1 is that the selection of the most promising view should be learned without any user interaction. Reinforcement learning provides many different algorithms to estimate the action value function based on a trial and error method [12]. Trial and error means that the system itself is responsible for trying certain actions in a certain state. The result of such a trial, i.e. the return R_t, is then used to update the function Q and to improve its policy π (see equation (3)).

In reinforcement learning a series of *episodes* are performed; each episode k consists of a sequence of state/action pairs $(\boldsymbol{s}_t, \boldsymbol{a}_t), t \in \{0, 1, \ldots, T\}$, where the action $\boldsymbol{a}_t = \pi_k(\boldsymbol{s}_t)$ in state $\boldsymbol{s}_t$ results in a new state $\boldsymbol{s}_{t+1}$. A final state $\boldsymbol{s}_T$ is called the terminal state, where a predefined goal is reached and the episode ends. In our case, the terminal state is that state, where classification and localization is possible with high confidence. The definition of high confidence is application dependent. For each episode k the policy $\pi_k(\boldsymbol{s})$, which has been estimated up to episode k, is fixed. During the episode new returns $R_t^{(k)}$ are collected for these state/action pairs $(\boldsymbol{s}_t^k, \boldsymbol{a}_t^k)$, which have been visited at time t during the episode k. After the end of the episode the action–value function is updated. In our case the so called Monte Carlo learning is applied, i.e. the action–value function is updated by

$$\forall(\boldsymbol{s}, \boldsymbol{a}): \quad Q^{\pi_k}(\boldsymbol{s}, \boldsymbol{a}) = E_\pi\{R_t \,|\, \boldsymbol{s}_t = \boldsymbol{s}, \boldsymbol{a}_t = \boldsymbol{a}\} \approx \frac{\sum_{i=1}^{k} \sum_{\{t | \boldsymbol{s}_t^i = \boldsymbol{s}, \boldsymbol{a}_t^i = \boldsymbol{a}\}} R_t^{(i)}}{\sum_{i=1}^{k} |\{t | \boldsymbol{s}_t^i = \boldsymbol{s}, \boldsymbol{a}_t^i = \boldsymbol{a}\}|} \quad (4)$$

In other words, the function Q is estimated by the mean of all collected returns $R_t^{(i)}$ for the state/action pair $(\boldsymbol{s}, \boldsymbol{a})$ for all episodes — which is the fraction on the right hand side of (equation 4).

As a result for the next episode one gets a new decision rule π_{k+1}, which is now computed by maximizing the updated action value function. This procedure is repeated until the action–value function converges to Q^* and as a consequence the final and optimal decision rule π^* it returned.

The reader is referred to a detailed introduction to reinforcement learning [12] for a description of other ways for estimating the function Q. Convergence proofs for several algorithms can be found in [1].

3.2 Function Approximation for Continuous Reinforcement Learning

Most of the algorithms in reinforcement learning treat the states and actions as discrete variables. Of course, in viewpoint selection parts of the state space (the pose of the object) and the action space (the viewpoints of the object) is continuous. The idea of continuous reinforcement learning can be summarized as follows:

1. collect returns for a finite set of state/action pairs $(\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{Q}$ and use them to compute $Q(\boldsymbol{s}, \boldsymbol{a})$ for these state/action pairs by methods of discrete reinforcement learning. $\mathcal{Q}(\boldsymbol{s})$ denotes the set of all state/action pairs $(\boldsymbol{s}', \boldsymbol{a}') \in \mathcal{Q}$ whose state $\boldsymbol{s}'$ has the same estimated class as $\boldsymbol{s}$;
2. approximate $Q(\boldsymbol{s}, \boldsymbol{a})$, which is continuous in its parameters by a function

$$\widehat{Q}(\boldsymbol{s}, \boldsymbol{a}) = \frac{\sum\limits_{(\boldsymbol{s}', \boldsymbol{a}') \in \mathcal{Q}(\boldsymbol{s})} d(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}', \boldsymbol{a}') \cdot Q(\boldsymbol{s}', \boldsymbol{a}')}{\sum\limits_{(\boldsymbol{s}', \boldsymbol{a}') \in \mathcal{Q}(\boldsymbol{s})} d(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}', \boldsymbol{a}')}, \quad (5)$$

which is the weighted average of the values of $Q(\boldsymbol{s}', \boldsymbol{a}')$ of all $(\boldsymbol{s}', \boldsymbol{a}') \in \mathcal{Q}(\boldsymbol{s})$.

This two step approach is called function approximation [12] and has been proposed in [10]. The weight function $d(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}', \boldsymbol{a}')$, which measures some kind of distance, defines how much influence the observed return at $(\boldsymbol{s}', \boldsymbol{a}') \in \mathcal{Q}$ has on an arbitrary state/action pair $(\boldsymbol{s}, \boldsymbol{a})$. Obviously, the function $\widehat{Q}(\boldsymbol{s}, \boldsymbol{a})$ is continuous in $\boldsymbol{s}$ and $\boldsymbol{a}$. The key point for viewpoint selection is the choice of the weights $d(\cdot, \cdot, \cdot, \cdot)$. We have chosen the product form

$$d(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}', \boldsymbol{a}') = K_\varphi(\varphi(\bar{\boldsymbol{s}}, \bar{\boldsymbol{s}}')) \cdot K_\mu(\mu(\boldsymbol{s}, \boldsymbol{s}')) \tag{6}$$

for the weight function, where K_φ and K_μ are two kernel functions, for example Gaussian kernels. The following two functions $\varphi(\cdot, \cdot)$ and $\mu(\cdot, \cdot)$ are defined:

– The function $\varphi(\cdot, \cdot)$ measures the distance between the two *expected destination states* $(\bar{\boldsymbol{s}}, \bar{\boldsymbol{s}}')$ of two state-action pairs $(\boldsymbol{s}, \boldsymbol{a})$ and $(\boldsymbol{s}', \boldsymbol{a}')$. Assuming that the actions $\boldsymbol{a}$ and $\boldsymbol{a}'$ lead to two new destination state $\bar{\boldsymbol{s}}$ and $\bar{\boldsymbol{s}}'$: $\boldsymbol{s} \xrightarrow{\boldsymbol{a}} \bar{\boldsymbol{s}}, \quad \boldsymbol{s}' \xrightarrow{\boldsymbol{a}'} \bar{\boldsymbol{s}}'$.
The closer the two destination states are to each other, the more adaptable is $Q(\boldsymbol{s}', \boldsymbol{a}')$ for the estimation of $\widehat{Q}(\boldsymbol{s}, \boldsymbol{a})$. For calculating the expected destination states we are currently assuming that the pose estimation of state $\boldsymbol{s}$ is correct and that action $\boldsymbol{a}$ is affecting the environment in an ideal way. For example, if the estimated position of $\boldsymbol{s}$ is $200°$ and an action $\boldsymbol{a}$ moves the camera $100°$ the pose of the expected destination state $\bar{\boldsymbol{s}}$ will be $300°$.
– The function $\mu(\cdot, \cdot)$ measures the distance between the two *source states* $\boldsymbol{s}$ and $\boldsymbol{s}'$. The fundamental idea for this distance is that close source states are suitable for using $Q(\boldsymbol{s}', \boldsymbol{a}')$ for the estimation of $\widehat{Q}(\boldsymbol{s}, \boldsymbol{a})$ because close source states imply less external influences as e.g. precision of camera positioning, classification and localization results.

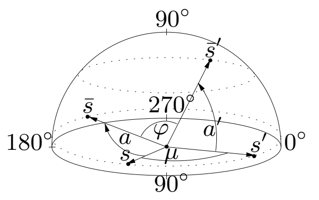

Figure 3. Source states ($\boldsymbol{s}$, $\boldsymbol{s}'$), executed actions ($\boldsymbol{a}$, $\boldsymbol{a}'$), resulting expected destination states ($\bar{\boldsymbol{s}}$, $\bar{\boldsymbol{s}}'$). Also shown are the distances φ and μ.

As pose estimation is not done with absolute coordinates in space, but with angles on the sphere, calculating distances between states can be done by measuring the angle between the vectors on the sphere given by the states (see figure 3):

$$\varphi(\bar{s}, \bar{s}') = \arccos\left(\frac{(\bar{\boldsymbol{s}}, \bar{\boldsymbol{s}}')}{||\bar{\boldsymbol{s}}|| \cdot ||\bar{\boldsymbol{s}}'||}\right) \tag{7}$$

$$\mu(s, s') = \arccos\left(\frac{(\boldsymbol{s}, \boldsymbol{s}')}{||\boldsymbol{s}|| \cdot ||\boldsymbol{s}'||}\right) \tag{8}$$

Substituting equations (7), (8) and (6) in equation (5) leads to the approximation of any action-value

$$\hat{Q}(\boldsymbol{s}, \boldsymbol{a}) = \frac{\sum\limits_{(\boldsymbol{s}', \boldsymbol{a}') \in \mathcal{Q}(\boldsymbol{s})} d(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}', \boldsymbol{a}') Q(\boldsymbol{s}', \boldsymbol{a}')}{\sum\limits_{(\boldsymbol{s}', \boldsymbol{a}') \in \mathcal{Q}(\boldsymbol{s})} d(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}', \boldsymbol{a}')} = \frac{\sum\limits_{(\boldsymbol{s}', \boldsymbol{a}') \in \mathcal{Q}(\boldsymbol{s})} K_\varphi(\varphi(\bar{\boldsymbol{s}}, \bar{\boldsymbol{s}}')) K_\mu(\mu(\boldsymbol{s}, \boldsymbol{s}')) Q(\boldsymbol{s}', \boldsymbol{a}')}{\sum\limits_{(\boldsymbol{s}', \boldsymbol{a}') \in \mathcal{Q}(\boldsymbol{s})} K_\varphi(\varphi(\bar{\boldsymbol{s}}, \bar{\boldsymbol{s}}')) K_\mu(\mu(\boldsymbol{s}, \boldsymbol{s}'))} \tag{9}$$

with the two kernel functions $K_\varphi(x) = \exp\left(-x^2/D_\varphi^2\right)$ and $K_\mu(x) = \exp\left(-x^2/D_\mu^2\right)$. The parameters D_φ and D_μ describe how local (for small D) or global (for large D) the approximation is working. "Local" means that faraway states have only a slight influence on the approximated action-value resulting in a very detailed approximation. This is very useful if there are a lot of state-action pairs. On the contrary, a "global" approximation includes data over a wide area of distances and is suitable if only a very limited set of collected action-values are available.

object	rec.rate	o1	o2	o3	o4	o5
o1: band	68.0%	100	40	7	0	0
o2: lamp	87.9%	21	153	0	0	0
o3: quiver	94.7%	8	0	143	0	0
o4: gun	64.7%	0	0	0	110	60
o5: trumpet	91.4%	0	0	0	16	169

object	rec.rate	o1	o2	o3	o4	o5
o1: band	92.0%	46	2	2	0	0
o2: lamp	96.0%	2	48	0	0	0
o3: quiver	98.0%	0	1	49	0	0
o4: gun	98.0%	0	0	0	49	1
o5: trumpet	100.0%	0	0	0	0	50

Table 1. Classification results (in percent) and confusion matrix (absolute numbers) for the Eigenspace approach: Left, without viewpoint selection (randomly chosen views). Right, with viewpoint selection

3.3 Viewpoint Selection by Function Optimization

Viewpoint selection, i.e. the computation of the policy π (see equation (3)), is now an optimization problem

$$\pi(\boldsymbol{s}) = \underset{\boldsymbol{a}}{\operatorname{argmax}}\, \pi(\boldsymbol{s}, \boldsymbol{a}) = \underset{\boldsymbol{a}}{\operatorname{argmax}}\, \widehat{Q}(\boldsymbol{s}, \boldsymbol{a}). \tag{10}$$

Up to now, we have not looked for a closed form solution of the maximum of this or any other kind of parameterized function. Instead, we applied numerical optimization algorithms to the optimization problem in equation (10), like the adaptive random search algorithm, followed by a simplex step (cf. [13]).

4 Experimental Evaluation

For the experiments presented in this section we have decided for an appearance based classifier using the Eigenspace approach [7]. As already mentioned, the proposed viewpoint selection is independent of the used classifier. The only classifier dependent part is the reward function as used in (1). We use the following function

$$r_t = \min_{\lambda, \lambda \neq \kappa} \left(\left(\min_{\kappa} d(O_t | B_\kappa) \right) - d(O_t | B_\lambda) \right) \tag{11}$$

with $d(O_t|B_\kappa)$ being the distance of the picture O_t to the object class B_κ measured in the Eigenspace (for an explanation of the Eigenspace approach and how classification is usually done see for example [7]). In other words, we define a viewpoint to be useful if the difference between the best and second best object hypotheses is large. It is worth noting that of course other definitions of the reward are possible. Nevertheless such a discussion is not the focus of this paper.

Our data set consists of five toy manikins (shown in Figure 1). Two groups of manikins have been selected in a way that they are strongly ambiguous within the group: for the first group they only differ by the band, the lamp and the quiver. The objects in the second group can only be distinguished by the gun and the trumpet, which the manikins hold in their hands. The reader should note that there does not exist one unique viewpoint, which allows to distinguish all five objects.

During the training of the Eigenspace classifier for each object class 1200 images have been taken covering the sphere around the object in steps of nine degree for the azimuthal angle and three degree for the colatitude angle. Two different lighting conditions have been used. After the configuration of the classifier we got an overall recognition rate of 81.6%. The single results are shown in Table 1, on the left side, together with the confusion matrix on the right. As expected the objects within the two groups (o1/o2/o3 and o4/o5) are sometimes mixed up. This is caused by the ambiguities that cannot be resolved in any case having only one view. These results are compared in the following with the viewpoint selection approach. The function $Q(\boldsymbol{s}, \boldsymbol{a})$ (compare equation (2) and equation (9)) has been estimated by performing for each object 150 random movements of the camera around the object. The value γ has been set to zero, i.e. only the current reward is taken into account in the computation of $Q(\boldsymbol{s}, \boldsymbol{a})$. Being in state $\boldsymbol{s}_t$, i.e. having a class and pose estimate for the object, a random camera movement $\boldsymbol{a}_t$ is chosen. The resulting view is used to classify the object. As a result, the reward is returned, which is stored in $Q(\boldsymbol{s}_t, \boldsymbol{a}_t)$. It is worth mentioning that this is a unsupervised training step. This means also that the system is not told whether or not a classification result is correct.

During the test of our viewpoint selection approach the camera has been positioned randomly on the sphere. An image is taken and based on the classification result the decision for the next view is made based on equation (3). The next view is taken and used to classify the object. Thus, only one new viewpoint is used in this case. One reason is that these two images allow in almost all cases for a reliable classification with respect to the reward defined in (11).

The classification rates for the five objects using viewpoint selection are shown in Table 1, right. We got an overall classification rate of 96.8% compared to a rate of 81.6% with a strategy which randomly chooses next views. The classification rate of 81.6% is calculated from the rates of the training set where only random views were produced. As the tests of our viewpoint selection approach start from randomly chosen positions on the sphere, the two classification rates are well comparably. As expected, the number of confusions between objects within one group is noticeable reduced.

In Figure 4, two estimated functions $\hat{Q}(\boldsymbol{s}, \boldsymbol{a})$ for object o1 (band) are shown. One can see that there are several significant views. The best and the worst view for object o1 based on the estimated function $\hat{Q}(\boldsymbol{s}, \boldsymbol{a})$ are shown in Figure 4. As one can see by means of the plots, choosing low values for D_φ and D_μ results in a more detailed $\hat{Q}(\boldsymbol{s}, \boldsymbol{a})$ but comes along with many local maxima.

The computation of one $\hat{Q}(\boldsymbol{s}, \boldsymbol{a})$ takes about $8 \cdot 10^{-3}$ seconds on a SGI O^2 (R10000 150 MHz). The optimization algorithm needs an average of 300 function evaluations of $\hat{Q}(\boldsymbol{s}, \boldsymbol{a})$ which results in a total time needed for one viewpoint selection of 2.4 seconds.

5 Summary and Future Work

In this paper we have presented a general framework of viewpoint selection that is independent of the chosen classifier that can be trained automatically without user interactions, and that results in a continuous space for the possible viewpoints. We claim that these three properties have not been provided by any other approach up to now. The experimental results using an Eigenspace approach for classification show that even with just one, optimally chosen additional view, recognition can be improved from 81.6% to 96.8%.

Currently, some valid objections are possible: first, we neither do sensor data fusion for the two views nor fusion of the classification results. This is the reason, why we can use the parameter $\gamma = 0$. Of course, if we do sensor data fusion, i.e. we combine the information of two images and more general methods of reinforcement learning, like Q–learning [12], can be applied. This is one important goal of our future work. Second, we have only used five classes. The reason was to show the principles of our approach, and how it works in practice. Currently, we have started experiments in an office scene, where more objects and a more difficult environment is found. Then, also the whole framework of reinforcement learning becomes more important, where episodes and final states must be taken into account. This was not necessary for the five classes in our experiments, since almost always after the second view a correct classification was possible. Nevertheless, the classification rate could be improved by 18.6%. Third, we have only tested one classifier in our experiments. Of course, to show the classifier independency, we have

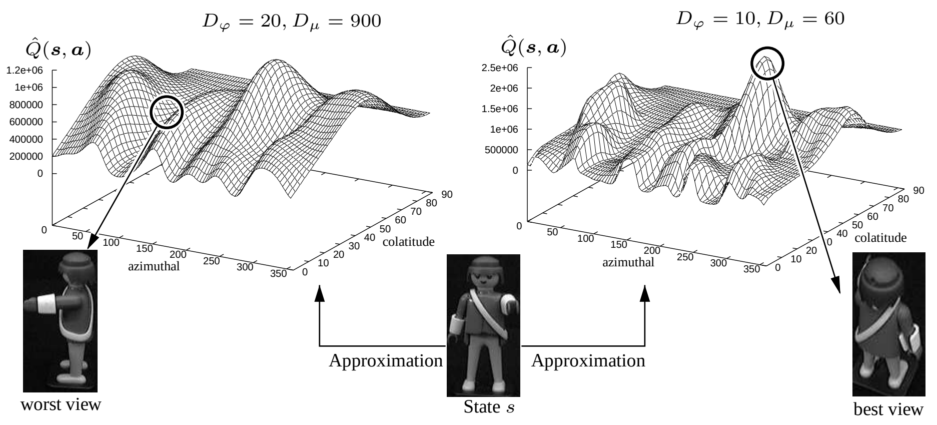

Figure 4. Results of viewpoint selection for object o1 (band), i.e. $\boldsymbol{s} = (o1, \alpha, \beta)^T$. The two plots show the estimated function $\hat{Q}(\boldsymbol{s},\boldsymbol{a})$ for the state represented by the picture in the center. Left picture: The action which leads to the worst view (no distinction possible to object o2). Right picture: The best action and the resulting view. The action which leads to this view is found by the optimization algorithm.

to show results for other classifier, too. Actually, we have preliminary results for the statistical classifier, which has been described in [9]. Finally, we have not evaluated pose estimation for the results neither with nor without viewpoint selection. This is our near future work, to show that not only classification but also localization can be improved.

References

[1] D.P. Bertsekas and J.N. Tstsiklis. *Neuro–Dynamic Programming*. Athena Scientific, 1996.

[2] H. Borotschnig, L. Paletta, M. Prantl, and A. Pinz. A comparison of probabilistic, possibilistic and evidence theoretic fusion schemes for active object recognition. *Computing*, 62:293–319, 1999.

[3] F. Deinzer, J. Denzler, and H. Niemann. Viewpoint selection - a classifier independent learning approach. In *IEEE SSIAI*, pages 209–213. Los Alamitos, 2000.

[4] B. Krebs, M. Burkhardt, and B. Korn. Handling Uncertainty in 3D Object Recognition using Bayesian Networks. In *ECCV98*, pages 782–795, 1998.

[5] B. Krebs, B. Korn, and M. Burkhardt. A 3D Object Recognition System with Decision Reasoning under Uncertainty. In E. Paulus and F.M. Wahl, editors, *Mustererkennung 1997*, pages 183–190, Braunschweig, 1997.

[6] C.B. Madsen and H.I. Christensen. A Viewpoint Planning Strategy for Determining True Angles on Polyhedral Objects by Camera Alignment. *PAMI*, 19(2), 1997.

[7] H. Murase and S. Nayar. Visual Learning and Recognition of 3–D Objects from Appearance. *International Journal of Computer Vision*, 14:5–24, 1995.

[8] P. Lehel and E.E. Hemayed and A.A. Farag. Sensor planning for a trinocular active vision system. In *CVPR*, pages II:306–312, 1999.

[9] J. Pösl and H. Niemann. Wavelet features for statistical object localization without segmentation. In *ICIP*, volume 3, pages 170–173, 1997.

[10] J. Carlo Santamaria, Richard S. Sutton, and Ashwin Ram. Experiments with Reinforcement Learning in Problems with Continuous State and Action Spaces. Technical report, University of Massachusetts, Amherst, Computer Science, 1996.

[11] B. Schiele and J.L. Crowley. Transinformation for Active Object Recognition. In *ICCV*, pages 249–254, Bombay, India, 1998.

[12] R.S. Sutton and A.G. Barto. *Reinforcement Learning*. A Bradford Book, Cambridge, London, 1998.

[13] A. Törn and A. Žilinskas. *Global Optimization*, volume 350 of *Lecture Notes in Computer Science*. Springer, Heidelberg, 1987.