3D OBJECT CATEGORIZATION WITH PROBABILISTIC CONTOUR MODELS

Gaussian Mixture Models for 3D Shape Representation

Kerstin Pötsch and Axel Pinz
Inst. of El. Meas. and Measurement Sig. Proc., Graz University of Technology, Kopernikusgasse 24/IV, Graz, Austria

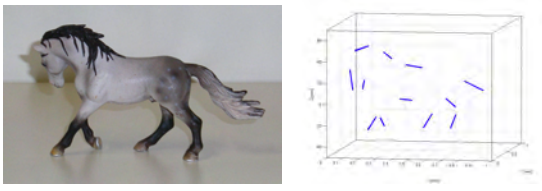
Keywords: 3D Object categorization, 3D Contour model, Gaussian mixture models.

Abstract: We present a probabilistic framework for learning 3D contour-based category models represented by Gaussian Mixture Models. This idea is motivated by the fact that even small sets of contour fragments can carry enough information for a categorization by a human. Our approach represents an extension of 2D shape based approaches towards 3D to obtain a pose-invariant 3D category model. We reconstruct 3D contour fragments and generate what we call '3D contour clouds' for specific objects. The contours are modeled by probability densities, which are described by Gaussian Mixture Models. Thus, we obtain a probabilistic 3D contour description for each object. We introduce a similarity measure between two probability densities which is based on the probability of intra-class deformations. We show that a probabilistic model allows for flexible modeling of shape by local and global features. Our experimental results show that even with small inter-class difference it is possible to learn one 3D Category Model against another category and thus demonstrate the feasibility of 3D contour-based categorization.

1 MOTIVATION AND IDEA

Shape features, especially contour features are challenging in computer vision for several reasons. The research topics vary from 2D contour generation, 2D object recognition and categorization to 3D contour reconstruction from multiview-stereo image sequences or 3D contour generation from 3D models. This paper deals with object categorization based on 3D contour models, which we call '3D contour clouds'. We represent these '3D contour clouds' by Mixture of Gaussian Models and we learn partitions of probability density functions to achieve a 3D category shape model.

In 2D, shape features play an important role for object categorization. Contour features, e.g. silhouette features as well as inner contours, have successfully been used in several recent categorization systems (Leordeanu et al., 2007), (Opelt et al., 2006), (Shotton et al., 2008). These approaches are based on fragments of contours (Opelt et al., 2006; Shotton et al., 2008) or by simplifying them to a sparse point representation (Leordeanu et al., 2007). In such systems, 2D contour features are used to model the shape of various object categories. Either, these category models are learned in a 2D Implicit Shape Model (ISM) (Leibe et al., 2004) manner, where an object center is used as reference point to build indirect relations between pairs of contour fragments and this 'centroid' (Opelt et al., 2006), (Shotton et al., 2008), or they are learned using direct pairwise relations between such contour fragments (Leordeanu et al., 2007). The main disadvantage of these 2D shape models is their view dependency because they are quite sensitive to view/pose changes and consequently, one model per significant aspect of a category has to be learned independently to achieve robustness to view and pose changes.



(a) (b)

Figure 1: (a) A horse, shown as a collection of 3D line segments (b).

2D object categorization building on 2D contour models (Opelt et al., 2006; Shotton et al., 2008) relies on the fact that contour fragments are sufficient to

Pötsch K. and Pinz A..
3D OBJECT CATEGORIZATION WITH PROBABILISTIC CONTOUR MODELS - Gaussian Mixture Models for 3D Shape Representation.
DOI: 10.5220/0003317402590270
In *Proceedings of the International Conference on Computer Vision Theory and Applications* (VISAPP-2011), pages 259-270
ISBN: 978-989-8425-47-8

represent object shape (Lowe and Binford, 1983). By looking at Figure 1(b), humans can easily identify the object based on this simple collection of straight 3D lines. We want to extend the idea of a 2D shape model for categorization to 3D to obtain a pose-invariant 3D category model. Therefore, we make use of the following principle:

Let us assume that we have a 3D model of an object such as the horse in Figure 1(a). Such a horse consists of several parts, e.g. head, legs, back, tail etc. We can assume, that all these parts are in a spatial relation to each other which can be described by probabilities. Furthermore, if these parts are represented by 3D contour fragments, we can assume that these 3D contour fragments are in a spatial relation and orientation to each other. So, each object can be described by a small set of fragments having a specified orientation and spatial relation to each other.

Our approach can then be summarized as follows: We use probabilistic density functions for 3D shape modeling. 3D contour fragments - 3D manifolds in 1D - are represented as Gaussian Mixture Models (GMMs). These 3D contour fragments are reconstructed from stereo image sequences, and build '3D contour clouds' for specific objects of a category. We learn partitions of probability density functions using a random feature selection algorithm. The distance function for such partitions is based on pairwise spatial relations and a similarity measure for mixture components. In contrast to 2D based approaches we build one single 3D model per category instead of one 2D model per significant view.
The contributions of this paper are

- 3D shape modeling by using GMMs for 3D contour fragments, and
- Learning of a pose-invariant 3D Contour Category Model consisting of partitions of probability densities between two categories with small interclass difference.

2 STATE-OF-THE-ART

Based on the contributions of this paper, this section is divided into three parts: 3D contour generation, 3D shape modeling and matching, and Gaussian Mixture Models for shape representation and matching.

The research on 3D contours (1D surface embedded in 3D) so far concentrates on 3D curve reconstruction (Ebrahimnezhad and Ghassemian, 2007; Park and Han, 2002; Fabbri and Kimia, 2010) or 3D contour extraction from 3D surface models (DeCarlo et al., 2003; DeCarlo and Rusinkiewicz, 2007; Ohtake et al., 2004; Pauly et al., 2003). (Ebrahimnezhad and Ghassemian, 2007) present a 3D contour reconstruction method based on the usage of a double stereo rig. They describe a method for 3D reconstruction of object curves from different views and motion estimation based on these 3D curves. (Park and Han, 2002) propose a method for Euclidean contour reconstruction including self-calibration. Unfortunately, this contour matching algorithm is not applicable to our image sequences. Recently, (Fabbri and Kimia, 2010) presented an approach for multi-view stereo reconstruction and calibration of curves. They concentrate on the reconstruction of contour fragments assuming that motion analysis can be obtained by other methods and their algorithm is mainly based on so called view-stationary curves, e.g. shadows, sharp ridges, reflectance curves. Therefore, it is well applicable for aerial images, as they show in their results. 3D contour extraction methods are manifold and differ mainly in the type of the extracted contour/line fragments, e.g. occluding contours, ridges (Ohtake et al., 2004), suggestive contours (DeCarlo et al., 2003), suggestive highlights and principal highlights (DeCarlo and Rusinkiewicz, 2007). For a more detailed literature overview we refer to the mentioned publications. As described in Appendix A, we use a method where 3D contour fragments are reconstructed from stereo image sequences. Our goal is to reconstruct a qualitative '3D contour cloud' representation rather than precise 3D contours for the shape of a category, which is not possible just based on stereo sequences.

There are many 3D shape matching approaches, where a 3D shape is directly matched to a 3D model database, such as the Princeton Shape Benchmark (Shilane et al., 2004) and the ISDB (Gal et al., 2007a). Point clouds, generated by laser range scanners, by stereo vision, or by Structure-from-Motion techniques, are probably the most obvious and simplest way to represent 3D shape. Often, these point clouds are converted to triangle meshes or polygonal models. The research which has been done to analyze, match and classify such models is extensive. The shape representations vary from shape distributions (Gal et al., 2007b; Mahmoudi and Sapiro, 2009; Ohbuchi et al., 2005; Osada et al., 2002) to symmetry descriptors (Kazhdan et al., 2004) or Skeletal Graphs (Sundar et al., 2003). In (Iyer et al., 2005; Tangelder and Veltkamp, 2007) several shape representation methods are discussed. To our knowledge, the method suggested in this paper is the first to demonstrate learning on the basis of 3D contour fragments.

Gaussian Mixture Models often have been used for shape modeling and shape matching, especially for point set registration (Cootes, 1999; Wang et al.,

2008; Peter and Rangarajan, 2009; Jian and Vemuri, 2005). Most of such methods take into account the whole Gaussian Mixture Models shape matching by defining or approximating information theoretic divergences on these mixtures. In contrast, we learn partitions of Mixture of Gaussians by considering pairwise relations between probability density functions and a similarity measure between densities based on the principal eigenvector.

3 3D MIXTURE OF GAUSSIAN CONTOUR MODEL

Modeling 3D contour fragments by probability density functions has several advantages. In particular, noise, outliers and deformations can be handled in a simple, natural way. Our algorithm is applicable to all kind of 3D contour fragments which describe the shape of a category. As mentioned in Section 2, there are many possibilities to generate 3D contour fragments. For the experiments of this paper, we reconstruct 3D contour fragments which build '3D contour clouds' from stereo image sequences of several objects. The '3D contour cloud' generation is not the main contribution of this paper, and therefore, it is covered in Appendix A. An example '3D contour cloud' of a horse is shown in Figure 2.

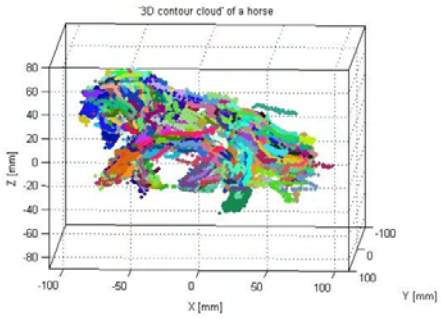


Figure 2: '3D contour cloud' of a horse. Each 3D contour fragment is represented by one color.

The representation with probability density functions is very flexible and robust. We can maintain the 3D geometry of 3D contour fragments. The time-complexity of generation and matching, and the quality of 3D geometry are closely interrelated. On the one hand, a higher number of mixtures better describes the shape of a contour fragment (see Figure 3). On the other hand, the time complexity increases during the learning stage.

The representation with Gaussian Mixture Models can handle several problems that may occur:

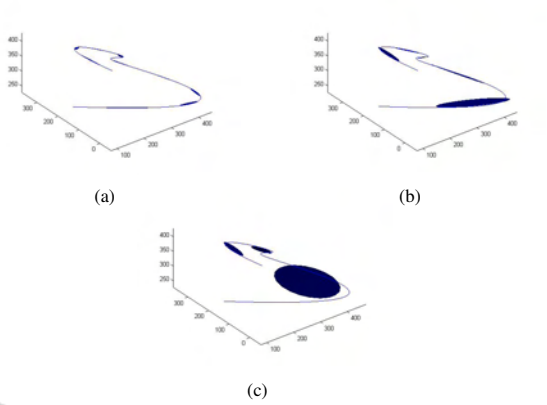


Figure 3: Gaussian Mixture Models (represented by blue ellipsoids) for a 3D contour with different numbers of mixture components K, (a) $K = 10$ (b) $K = 5$, (c) $K = 3$.

- Noise/Outliers: When working with real data like reconstructed 3D contours from images instead of synthetic 3D models, noise (like slightly displaced reconstructions) always plays an important role. If a few contour points are noisy, the reconstructed 3D contour fragment may no longer be represented as a long, connected contour. By the representation with a set of probability density functions, such a contour can be split into smaller parts and noisy parts can be reduced during the learning stage.
- Linking: When working with contour fragments, linking of edges to long connected contours always plays a role. Different linking in different models may have a strong influence on a matching algorithm. With the representation as Mixture of Gaussians, mixture components can be flexibly grouped during learning, resulting in partitions that are adapted to particular linkings (see Figure 4).
- Deformation: By modeling with probability functions, intra-class variability in form of shape deformations can be handled.

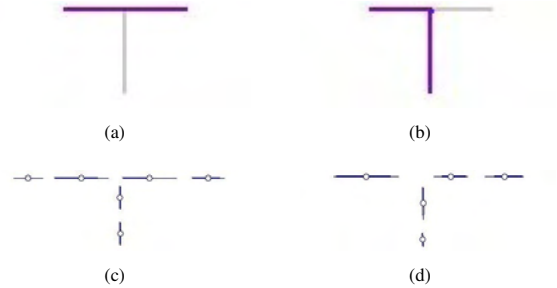


Figure 4: 2D example for different linking of the T-shape. (a) & (b) Show differently linked edges that form longer contour fragments, (c) & (d) represented as GMMs.

Given a '3D contour cloud' CC consisting of a set of 3D contour fragments F_i

$$CC = \{F_i; i = 1 : N\}, \tag{1}$$

where N is the number of 3D contour fragments in a cloud and $F_i = p_1, ..., p_n$ is a set of 3D points, we can fit a mixture of multivariate Gaussian distributions to each 3D contour fragment F_i using the standard Expectation-Maximization (EM) algorithm (see e.g. (Xu and Jordan, 1996)), so that each 3D contour fragment F_i is given by

$$\Theta_K = \sum_{k=1}^{K} \alpha_k N(\mu_k, \Sigma_k) \tag{2}$$

where Θ_K is the Gaussian Mixture Model for a fragment F_i and K is the number of mixture components with mean μ_k, variance Σ_k, and weight α_k. In our representation, K is chosen according to the length of the 3D contour fragment. In many mentioned approaches it is assumed that the Gaussians are spherical and the covariance matrix is set to the identity matrix. In our case, the covariance matrix gives essential information about the orientation of a 3D contour fragment, but we assume that each mixture component has the same weight, so that $\alpha_k = 1$. In our algorithm, we do not decide on the basis of the weight, if a mixture component is relevant for a category model or not because even a mixture component with a small weight can be important for a category model.

4 3D GAUSSIAN CONTOUR CATEGORY MODEL

We build one 3D Gaussian Contour Category Model from a set of Gaussian Contour Models of specific objects of a category. In contrast to other work on shape matching based on Gaussian Mixture Models we do not take into account the Gaussian Mixture Models in a whole and do not try to find the divergence to another GMM. We randomly select partitions of mixture components which are discriminative for a category against another one. The distance measure is given by a similarity measure between components and relative pairwise relations between components of one partition. In the following we describe the test statistic and the practical implementation in form of a random feature selection algorithm.

4.1 Test Statistic

Our approach uses a hypothesis test[1], to identify, if a given specific object O belongs to an object category C

$$\begin{array}{ll} H_0: & O \in C \\ H_1: & O \notin C \end{array} \qquad \text{reject } H_0 \text{ if } \quad SM(f_O \| g_C) > \gamma \tag{3}$$

where g_C is a learned Mixture of Gaussians category model and f_O is the Mixture of Gaussian of the specific object. The test statistic $TS = SM(f_O \| g_C)$ is a similarity measure between two Mixtures of Gaussians. On the basis of the threshold γ, the null hypothesis is rejected or not.

Most of the existing shape matching methods that use Gaussian Mixture Models are based on the Kullback-Leibler (KL) divergence between Mixtures of Gaussians or, similarily, the Jensen-Shannon divergence (e.g. (Wang et al., 2008)) as a similarity measure SM. There exists no closed-form for the KL divergence between two Mixtures of Gaussians, but there exists a closed-form Kullback-Leibler divergence between two Gaussians $N(\mu_1, \Sigma_1)$ and $N(\mu_2, \Sigma_2)$. It is given by

$$KL = \frac{1}{2}(\log \frac{|\Sigma_2|}{|\Sigma_1|} + tr(\Sigma_2^{-1}\Sigma_1) + (\mu_1 - \mu_2)^T \Sigma_2^{-1}(\mu_1 - \mu_2)) \tag{4}$$

Given two Mixtures of Gaussians f and g, where

$$f = \sum_{i=1}^{n} f_i = \sum_{i=1}^{n} \alpha_i N(\mu_i, \Sigma_i) \tag{5}$$

and

$$g = \sum_{j=1}^{m} g_j = \sum_{j=1}^{m} \beta_j N(\mu_j, \Sigma_j), \tag{6}$$

the following approximation of the KL-divergence between them has been suggested in (Goldberger et al., 2003):

$$KL(f\|g) \approx \sum_{i=1}^{n} \alpha_i \min_j (KL(f_i\|g_j) + \log \frac{\alpha_i}{\beta_j}) \tag{7}$$

With (7) we can rewrite the hypothesis test (3) as

reject H_0 if

$$KL(f\|g) \approx \sum_{i=1}^{n} \alpha_i \min_j (KL(f_i\|g_j) + \log \frac{\alpha_i}{\beta_j}) > \gamma. \tag{8}$$

[1]The statistical hypothesis, which has to be tested is the null hypothesis H_0. The alternative hypothesis is denoted by H_1. The test statistic TS is a statistic on whose value the null hypothesis will be rejected or not. The treshold of rejecting a null hypothesis is given by γ (notation is based on (Ross, 2005)).

With the simplification that all mixture components have the same weight $\alpha_i = \beta_j = 1$ and with $\gamma_0 = \gamma/n$, we further obtain

reject H_0 if

$$\sum_{i=1}^{n} \min_j (KL(f_i \| g_j) - \gamma_0) > 0. \tag{9}$$

In this approximation, the term $\min_j(KL(f_i\|g_j)-\gamma_0)$ might be very high when a part of an object is missing as there will be no g_j which is near to f_i. However, for our application we want to permit that certain parts of an object can be missing. Therefore, we suggest to use only discrete values for the similarity measure between two Gaussians:

reject H_0 if

$$\sum_{i=1}^{n} \mathrm{sgn}(\min_j (KL(f_i \| g_j) - \gamma_0)) > -n + 2l \tag{10}$$

where l is the number of Gaussians that are permitted to be missing in the sample. Please note that due to the discretization, the term $\mathrm{sgn}(\min_j(KL(f_i\|g_j)-\gamma_0))$ which is equivalent to $\min_j(\mathrm{sgn}(KL(f_i\|g_j)-\gamma_0))$ can be considered a hypothesis test: It is -1 (keep H_0), if there is at least one Gaussian in the set g that is equal (with respect to a certain significance level) to f_i and +1 (reject H_0) otherwise. The 'global' hypothesis is rejected if the number of Gaussians of f that do not have a match in g is larger than a predefined number l. In our similaritycase the Kullback-Leibler divergence $KL(f_i\|g_j)$ is not a particularly useful similarity measure. The probability density function, especially the covariance matrix Σ_i of points, gives no evidence about the shape variability of 3D contours in an object category. It is rather a representation of the reconstruction quality; noise/outliers may have an important influence on the covariance matrix. In the Kullback-Leibler divergence, a shift of the mean μ_i has more effect on the divergence measure than changes of the covariance matrix.

We propose a different similarity measure between two Gaussians which is better suitable for our objective, i.e. the learning of a 3D category shape model. As mentioned above, the covariance matrix represents the orientation of a contour by its principal component and it can handle noise and reconstruction errors by the other two dimensions. Consequently, GMMs are better suitable for shape representations than just an approximation by straight lines.

In the remaining section we use the following notation: (R,T) denotes the global transformation between objects, (R_i,T_i) denotes the local transformation of a single mixture component (μ_i,Σ_i). (μ_j^O,Σ_j^O) and (μ_i^C,Σ_i^C) are mixture components of a specific object O and a model C. Further, we will define v_{ab} as the difference between a pair of mixture components $((\mu_a^O,\Sigma_a^O),(\mu_b^O,\Sigma_b^O))$ of the object O and v_{xy} as the difference between a pair of mixture components $((\mu_x^C,\Sigma_x^C),(\mu_y^C,\Sigma_y^C))$ of the model C.

Specific objects of a category may differ by a global rigid transformation (R,T) between their Mixtures of Gaussians and a local shape transformation (R_i,T_i) between mixture components. Then the global and local transformation can be described by

$$(\mu_j^O,\Sigma_j^O) = (R*\mu_i^C + T_i + T, R*R_i*\Sigma_i^C). \tag{11}$$

To be insensitive to a global transformation, we consider pairs of mixture components. We can compute first a global rotation R between a pair of the object and the pair of the model by estimating the rotation between their principal eigenvectors. Afterwards, to be insensitive to the global displacement, we consider pairwise relations between mixture components, which is given by

$$v_{ab}^O = \mu_a^O - \mu_b^O. \tag{12}$$

Because of

$$\begin{aligned} v_{xy}^C &= (\mu_x^C + T + T_x) - (\mu_y^C + T + T_y) \\ &= (\mu_x^C + T_x) - (\mu_y^C + T_y), \end{aligned} \tag{13}$$

the global rigid translation T can be eliminated. For performance reasons we avoid to compute the global rotation for each pair. Instead, we did as preprocessing step a coarse alignment of the rotation of the whole 3D models.

We can define the following hypothesis to test the similarity of Gaussians:

$$\begin{aligned} H_0 &: \quad (\mu_j^O,\Sigma_j^O) = (\mu_i^C + T_i, R_i*\Sigma_i^C) \\ H_1 &: \quad (\mu_j^O,\Sigma_j^O) \neq (\mu_i^C + T_i, R_i*\Sigma_i^C) \end{aligned} \tag{14}$$

$$\text{reject} \quad H_0 \text{ if } TS_1 > \gamma_1 \text{ or } TS_2 > \gamma_2$$

where γ_1 and γ_2 are the thresholds and TS_1 and TS_2 are test statistics. For the test statistics we always consider pairs of mixture components. TS_1 is the test statistic for the pairwise difference of pairs of mixture components

$$TS_1 = \|v_{ab}^O - v_{xy}^C\|_2. \tag{15}$$

Let e_a^O (e_b^O) be the principal eigenvectors of Σ_a^O (Σ_b^O) and $e_x^C(e_y^C)$ the principal eigenvectors of $\Sigma_x^C(\Sigma_y^C)$, then TS_2 is given by the scalar product between the eigenvectors

$$TS_2 = e_i^O * e_j^C \text{ for } i = a,b \text{ and } j = x,y. \tag{16}$$

With this, we can introduce a discrete similarity measure

$$SM(f_i,g_j) = \begin{cases} 1 & \text{if } TS_1 > \gamma_1 \text{ or } TS_2 > \gamma_2 \\ -1 & \text{otherwise.} \end{cases} \tag{17}$$

Consequently, $SM(f_i, g_j) = 1$ is two mixture components have the same orientation ($TS_2 < \gamma_2$) and the same relative position ($TS_1 < \gamma_1$). Otherwise, H_0 (see 14) will be rejected. and modify (10) to

$$\text{reject } H_0 \text{ if} \quad \sum_{i=1}^{n} \min_j(SM(f_i, g_j))) > -n + 2l \tag{18}$$

The presented approach is suitable when we have more or less rigid objects, that are deformed only by small translations and rotations of the contours. However, an object may actually be composed of several parts. For example, the head of a horse will be a rather rigid object but can have significant displacement with respect to other parts of the animal. Therefore, we do not consider the GMM of a 3D model in a whole, instead we randomly select subsets of mixture components which we call partition. Such a partition lets us handle each part of an object independently or in context to other parts. The partitioning model Θ_M of a subset of probability functions of size M is given by:

$$\Theta_M = \sum_{m=1}^{M} \alpha_m N(\mu_m, \Sigma_m). \tag{19}$$

A 3D category models consists of a set of partitions which are learned using the method in Section 4.2. Whether a partition belongs to a category model or not is decided on the basis of equation (18) such that we define a hypothesis test for each partition Θ_M.

4.2 Learning by Random Feature Selection

In the previous section it was shown how we can test a sample object on a model of a category. In this section we describe how we extract a model of a category from a number of sample objects. In our practical implementation we use a random feature selection algorithm (see Figure 5) for the partitioning of probability density functions. Random feature selection is a fast method for reducing the number of features to a discriminative subset of features. In the random feature selection, we randomly select partitions of probability density functions and verify if they are discriminant on training data by testing the hypothesis above for pairs of the partitions. Before starting the random feature selection, the 3D models where aligned on the basis of their bounding box in that way that all animals show in the same direction. For performance issues it may be useful to do a preprocessing by considering the position of probability densities on the object. The random selection algorithm stops when a number of iterations or a number of selected partitions is reached.

By the partitioning of probability density functions we can make additional constraints about their distribution on the object:

- Locality constraint: The selected partitions of probability density functions should be distributed in a local environment on the object. By this constraint we may represent local features on the object, e.g. the leg or the head of an animal.
- Uniformity constraint: The selected partitions of probability densities should be distributed on the object, so that no two density functions should be in a local neighborhood. This constraint yields partitions that are distributed on the object such that each density may represent one part of the object.
- No constraint: The partitions are selected randomly without restrictions on the spatial distribution on the object.

In the random feature selection algorithm we first test if a selected partition of densities is discriminative on the positive training data, afterwards, if it is discriminative against the negative data. Finally, we obtain a subset of partitions of probability density functions.

The classification output $h_j(O)$ of a partition Θ_J on a 3D model O is given by

$$h_j(O) = \begin{cases} 1 & \text{if } \sum_{i=1}^{J} \min_j(SM(f_i, g_j))) > -J + 2l \\ & \forall g_j \in \Theta_J \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

The output of the whole detector $H(O)$ then is

$$H(O) = \frac{1}{Q} \sum_{q=1}^{Q} (h_q(O)). \tag{21}$$

5 EXPERIMENTS AND VALIDATION

We evaluate our approach by using a k-fold cross validation based on the idea of a leave-one-out cross validation but leaving out in each iteration one positive and one negative 3D model (see Section 5.1). The training and test data build '3D contour clouds' of the dataset, which is described in Section 5.2. We show that our approach can not only be used for the generation of a 3D category model but also for outlier reduction. The main part of this section consists of

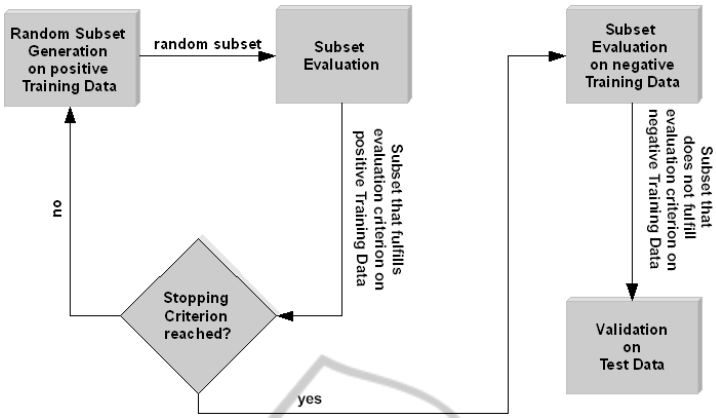


Figure 5: Random feature selection algorithm with validation.

experiments and results that show the possibility of learning one 3D model per category.

5.1 Cross Validation

We use k-fold cross validation to evaluate our experiments. Given a dataset D, this dataset is split into k subsets of approximately equal size. In our case $D = \{D^p_1, ..., D^p_{S_1}, D^n_1, ..., D^n_{S_2}\}$ of positive and negative 3D models and $k = S_1 + S_2$ subsets. Consequently, one subset is one 3D model. We train the classifier $S_1 * S_2$ times. In each iteration $t \in 1, ..., S_1 * S_2$, we leave out one positive and one negative 3D model. So we train on $D \backslash \{D^p_i, D^n_j\}$ $\forall i = 1, ..., S_1$ and $\forall j = 1, ..., S_2$ and test on D^p_i and D^n_j. In the cross validation we then build the average over the results for the positive and the negative test data.

5.2 Dataset

For our experiments we use our video databases containing stereo image sequences of small toy objects of the categories 'horse' and 'cow' (see Figure 6). The videos are taken by a calibrated stereo rig. The objects are either manipulated naturally by hand in front of a stereo camera system (see Figure 7) or they are rotated using a turntable. The objects are manipulated such that they are seen from all sides, showing all aspects. 3D contour fragments are generated using a multiview-stereo reconstruction scheme (see Appendix A).

Figure 8 shows one example stereo frame pair from a horse image sequence. Figure 2 shows an example of a '3D contour cloud' reconstruction of this horse as described in Appendix A. Figure 9(a) is the representation of the Gaussian Mixture Models of this '3D contour cloud', Figure 9(c) shows selected partitions. Our dataset consists of nine horse '3D contour clouds' and seven cow '3D contour clouds'. This leads to $9 * 7 = 63$ 'leave-one-out' experiments.

Figure 6: Database of objects containing nine horses and seven cows.

Figure 7: Stereo rig to capture the database. We use two μEye 1220C cameras and Cosmicar/Pentax lenses with a focal length of 12.5 mm. The baseline is approximately 6cm (human eye distance), the vergence angle 5.5^o. The frame rate is 15 Hz. The size of the images is 480x752 px. For the calibration of the stereo rig we use the Camera Calibration Toolbox for Matlab (Bouguet).

5.3 Experiments and Results

On the one hand, we show that the random feature selection algorithm applied to positive training data can be used to reduce outliers in 3D models. On the other

Figure 8: One example stereo frame pair of a horse. The hand is masked using the method by (Unger et al., 2009).

hand, - and this is the main part of this section - we show several experiments and results to achieve a 3D model for the category 'horse' against the category 'cow'.

5.3.1 Outlier Reduction

We mentioned in Section 3 that outliers in form of wrongly reconstructed contour points always play an important role in 3D reconstruction.

Noise is represented in the probability density function as mentioned in Section 4. But there exist also outliers that result from the reconstruction process and where a probability density function no longer belongs to the shape of the object. By running the random feature selection only on the positive training data, we can see, that noisy parts can be reduced significantly. Given nine horses, we randomly select 1000 partitions of five mixture components, where in each round the horse is randomly selected out of a set of eight horses. Figure 9(a) and Figure 9(b) show the Gaussian Mixture Models of two horses. For visualization, the probability densities are drawn by 3D lines given by their mean and the principal eigenvector. We can see that outliers exist, which results from the '3D contour cloud' generation. In Figure 9(c) and Figure 9(d), we can see selected partitions of size 5 from the two horses which were discriminant for all other horses. We can see, that most of the discriminative probability densities are located on the head, the back, and the tail of the horses. Fewer densities are located on the legs, because of different arrangements of the legs in the 3D horse models. Outliers have been significantly reduced.

5.3.2 Categorization

The aim of the following experiments is to learn a category model 'horse' against 'cows'. The challenge in this learning experiments is the small inter-class difference between horses and cows. Our object categorization system is validated using the cross-validation scheme described in Section 5.1. We did several experiments with different kinds of constraints (see Section 4.2) on the random selection of partitions. We can summarize the experiments as follows:

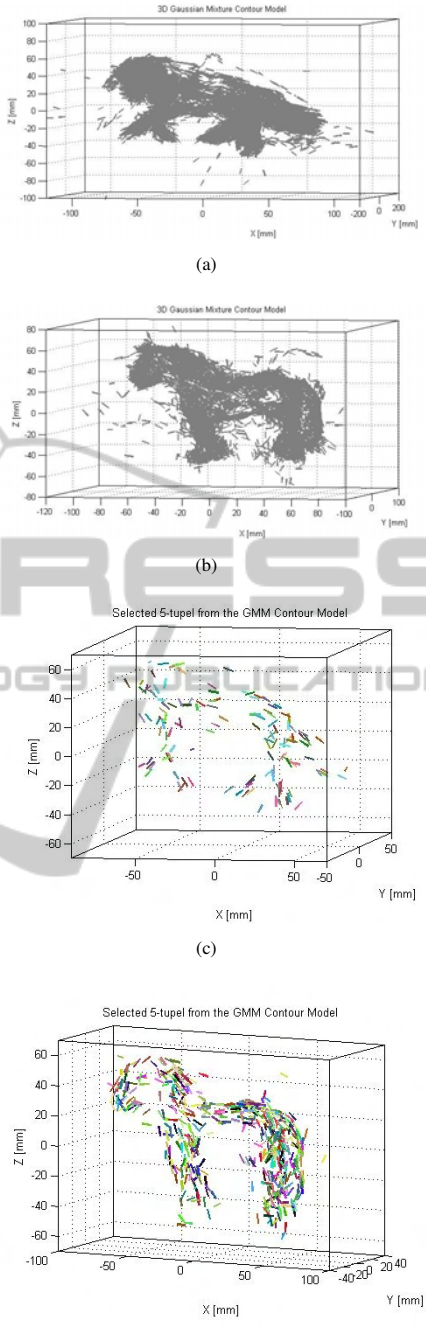


Figure 9: (a) & (c) GMM of a '3D contour cloud' of a horse and 28 partitions which have been selected. (b) & (d) GMM of a '3D contour cloud' of a horse and 128 partitions which have been selected. We see, that the number of outliers is significantly reduced. A partition of size 5 is represented by the same color.

- Experiment 1: We randomly select partitions of probability density functions from the 3D Gaussian Mixture Contour Models of randomly se-

lected horses of size 3 or 5, where we use the uniformity constraint (see Section 4.2). We perform a cross validation experiment where the results of this experiment are summarized in Table 1.

- Experiment 2: We randomly select partitions of probability densities from the 3D Gaussian Mixture Contour Models of randomly selected horses of size 5 or 7, where we use no constraint. The results of this experiment are summarized in Table 2.
- Experiment 3: We randomly select partitions of probability density functions from the 3D Gaussian Mixture Contour Models of randomly selected horses of size 5, where we use the locality constraint. The results of this experiment are summarized in Table 3.

For these experiments we typically choose $\gamma_2 = 0.98$ and $l = 0$. For Experiment 1 and Experiment 2 $\gamma_1 = 0.2$, for Experiment 3 $\gamma_1 = 0.1$. Too small γ_1 and γ_2 do not handle intra-class variability, too large γ_1 and γ_2 do not handle inter-class difference. The result tables (Table 1, Table 2, Table 3) contain seven entries for each experiment. The average result of the cross-validation for the positive training set 'horse' and the negative training set 'cow' is shown in the first (μ_{horse}) and the third (μ_{cow}) row. Assuming a normal distribution we can also compute the standard deviations σ_{horse} and σ_{cow}, as well as a classification threshold *c_thresh* on whose basis we can decide if a 3D test model is a 'horse' or a 'cow'. c_error_{horse} and c_error_{cow} are the classification errors. Figure 10 shows a graphical representation of the results of Experiment 1 - Partition (size 3). We can see that the two categories are well separable. As the results show, Experiment 1 and Experiment 2 perform better than Experiment 3 with the local features. In Experiment 1 and Experiment 2, the classification errors are $\leq 21\%$. The locality constraint seems to be less useful than a uniform distribution. This result is not very surprising. Local features often have similar shape which is not category specific, e.g. legs of horses and cows. Only combinations with other local features (e.g. leg and head of an animal) could give more discriminance. Moreover, we saw, that we have to learn longer for Experiment 3 for the same number of partitions than for Experiment 1 or Experiment 2. This is also true for learning smaller partitions e.g. Experiment 1 - partition (size 3).

Figure 11 shows an example of a learned 3D category model 'horse' from one training step of Experiment 1 - partition (size 5). For this model we randomly choose 1000 partitions, where 135 are found to be discriminative for horses and not for cows. The category model in Figure 11 has 135 partitions from eight horses. All partitions are drawn in one model without special aligning. We can see that discriminant probability density functions are located mainly on the head, the back and the tail, fewer are on the legs which is due to different arrangements of legs on different training models. We can see a similar behavior for Experiment 2. Figure 12 shows a learned 3D category model 'horse' for Experiment 2 - partition (size 7). The distribution of the probability density functions is similar to that of Experiment 1, on head, back, and tail. However, most of the densities are located on the head of the horse.

Table 1: Experiment 1: partition selection based on the uniformity constraint .

	Partition (size 3)	**Partition (size 5)**
μ_{horse}	0.7380	0.6781
σ_{horse}	0.1965	0.2328
μ_{cow}	0.3141	0.2481
σ_{cow}	0.2432	0.2191
c_thresh	0.5248	0.4632
c_error_{horse}	0.1401	0.1788
c_error_{cow}	0.1922	0.1635

Table 2: Experiment 2: partition selection based on no constraint .

	Partition (size 5)	**Partition (size 7)**
μ_{horse}	0.7066	0.6648
σ_{horse}	0.2225	0.2271
μ_{cow}	0.2836	0.2325
σ_{cow}	0.2555	0.2302
c_thresh	0.4912	0.4485
c_error_{horse}	0.1660	0.1711
c_error_{cow}	0.2090	0.1736

6 CONCLUSIONS AND OUTLOOK

We have presented a new approach for learning a 3D Gaussian Contour Category Model using a probabilistic framework based on Gaussian Mixture Models. Instead of modeling the whole shape by a GMM and computing a divergence between shapes, we represent 3D contour fragments by GMMs and apply a partitioning on them. We show that it is possible to build one single, pose-invariant 3D model per category instead of building one model per significant view as it is the case in 2D shape based models. Our results demonstrate that we can learn one category against another one, even when the inter-class difference is small. The experiments show that global partitions

Table 3: Experiment 3: partition selection based on the locality constraint .

	Partition (size 5)
μ_{horse}	0.5723
σ_{horse}	0.3304
μ_{cow}	0.2619
σ_{cow}	0.2907
c_thresh	0.4462
c_error_{horse}	0.3520
c_error_{cow}	0.2643

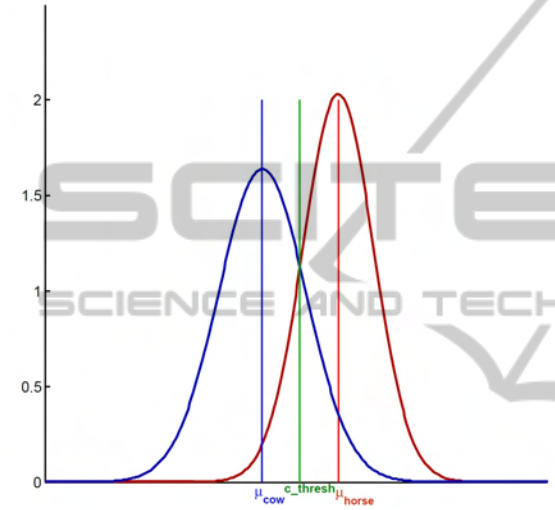


Figure 10: Graphical representation of the results of Experiment 1: Partition (size 3). The two categories are well separable.

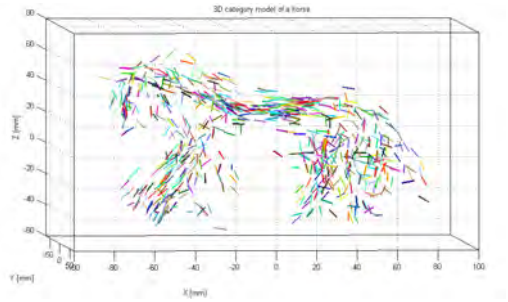


Figure 11: 3D category 'horse' model with 154 partitions from eight horses of size 5 of Experiment 1. The probability densities are drawn by 3D lines given by their mean and the principal eigenvector.

(with a classification error $\leq 21\%$) are better suitable for a category model than local features.

In our future work we plan to extend this approach to more object classes and to concentrate more on modeling shape properties, e.g. relations between transformations of probability density function and pairwise relations. Here, also a hierarchical approach

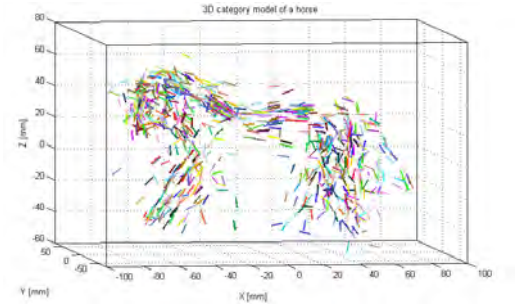


Figure 12: 3D category 'horse' model with 142 partitions from eight horses of size 7 of Experiment 2. The probability densities are drawn by 3D lines given by their means and their principal eigenvectors.

would make sense for building partitions. Moreover, we will improve learning by using a learning algorithm after the random subset selection. A further objective is to see if such a probabilistic 3D model can be used for pose estimation in 2D images (e.g. (Liebelt and Schmid, 2010)) or even for categorization in 2D images. This would overcome known problems of standard 2D categorization like sensitivity to pose/view changes.

ACKNOWLEDGEMENTS

This work was supported by the Austrian Science Fund (FWF) under the doctoral program Confluence of Vision and Graphics W1209.

REFERENCES

Belongie, S., Malik, J., and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE Trans. PAMI*, 24(4):509–522.

Bouguet, J.-Y. Camera calibration toolbox for matlab.

Cootes, T. (1999). A mixture model for representing shape variation. *Image and Vision Computing*, 17(8):567–573.

DeCarlo, D., Finkelstein, A., Rusinkiewicz, S., and Santella, A. (2003). Suggestive contours for conveying shape. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 22(3):848–855.

DeCarlo, D. and Rusinkiewicz, S. (2007). Highlight lines for conveying shape. In *International Symposium on NPAR*.

Ebrahimnezhad, H. and Ghassemian, H. (2007). Robust motion and 3d structure from space curves. *ISSPA 2007*, pages 1–4.

Fabbri, R. and Kimia, B. B. (2010). 3D curve sketch: Flexible curve-based stereo reconstruction and calibration. In *Proceedings of the IEEE CVPR 2010*.

Gal, R., Shamir, A., and Cohen-Or, D. (2007a). Pose-oblivious shape signature. *IEEE TVCG*, 13(2):261–271.

Gal, R., Shamir, A., and Cohen-Or, D. (2007b). Pose-oblivious shape signature. *IEEE TVCG*, 13(2):261–271.

Goldberger, J., Gordon, S., and Greenspan, H. (2003). An efficient image similarity measure based on approximations of KL-divergence between two gaussian mixtures. In *Proc. on ICCV 2003*, pages 487 –493 vol.1.

Iyer, N., Jayanti, S., Lou, K., Kalyanaraman, Y., and Ramani, K. (2005). Three-dimensional shape searching: state-of-the-art review and future trends. *Computer-Aided Design*, 37(5):509–530.

Jian, B. and Vemuri, B. (2005). A robust algorithm for point set registration using mixture of Gaussians. In *Proc. on ICCV 2005*, volume 2, pages 1246 – 1251 Vol. 2.

Kazhdan, M., Funkhouser, T., and Rusinkiewicz, S. (2004). Symmetry descriptors and 3D shape matching. In *Symposium on Geometry Processing*.

Leibe, B., Leonardis, A., and Schiele, B. (2004). Combined object categorization and segmentation with an implicit shape model. In *Workshop on Statistical Learning in Computer Vision, ECCV*.

Leordeanu, M., Hebert, M., and Sukthankar, R. (2007). Beyond local appearance: Category recognition from pairwise interactions of simple features. In *Proc. of CVPR*.

Liebelt, J. and Schmid, C. (2010). Multi-View Object Class Detection with a 3D Geometric Model. In *IEEE Conference on CVPR*.

Lowe, D. G. and Binford, T. O. (1983). Perceptual Organization as a Basis for Visual Recognition. In *AAAI*, pages 255–260.

Mahmoudi, M. and Sapiro, G. (2009). Three-dimensional point cloud recognition via distributions of geometric distances. *Graph. Models*, 71(1):22–31.

Ohbuchi, R., Minamitani, T., and Takei, T. (2005). Shape-similarity search of 3d models by using enhanced shape functions. In *IJCAT*, volume 2005, pages 70–85.

Ohtake, Y., Belyaev, A., and Seidel, H.-P. (2004). Ridge-valley lines on meshes via implicit surface fitting. *ACM Trans. Graph.*, 23(3):609–612.

Opelt, A., Pinz, A., and Zisserman, A. (2006). A boundary-fragment-model for object detection. In *Proc. of ECCV*.

Osada, R., Funkhouser, T., Chazelle, B., and Dobkin, D. (2002). Shape distributions. *ACM Trans. Graph.*, 21(4):807–832.

Park, J. S. and Han, J. H. (2002). Euclidean reconstruction from contour matches. *PR*, 35:2109–2124.

Pauly, M., Keiser, R., and Gross, M. H. (2003). Multi-scale feature extraction on point-sampled surfaces. *Comput. Graph. Forum*, 22(3):281–290.

Peter, A. M. and Rangarajan, A. (2009). Information Geometry for Landmark Shape Analysis: Unifying Shape Representation and Deformation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(2):337–350.

Ross, S. M. (2005). *Introductory Statistics*. Academic Press, Inc., Orlando, FL, USA.

Schweighofer, G., Segvic, S., and Pinz, A. (2008). Online/realtime structure and motion for general camera models. *IEEE WACV*, pages 1–6.

Shilane, P., Min, P., Kazhdan, M., and Funkhouser, T. (2004). The princeton shape benchmark. In *Shape Modeling International*.

Shotton, J., Blake, A., and Cipolla, R. (2008). Multi-Scale Categorical Object Recognition Using Contour Fragments. *IEEE Transactions on PAMI*, 30(7):1270–1281.

Sundar, H., Silver, D., Gagvani, N., and Dickinson, S. (2003). Skeleton based shape matching and retrieval. *SMI '03: Proceedings of the Shape Modeling International 2003*, page 130.

Tangelder, J. and Veltkamp, R. (2007). A survey of content based 3d shape retrieval methods. *Multimedia Tools and Applications*.

Unger, M., Mauthner, T., Pock, T., and Bischof, H. (2009). Tracking as segmentation of spatial-temporal volumes by anisotropic weighted tv. In *7th International Conference, EMMCVPR*, volume 5681 of *LNCS*, Bonn, Germany. Springer.

Wang, F., Vemuri, B. C., Rangarajan, A., and Eisenschenk, S. J. (2008). Simultaneous Nonrigid Registration of Multiple Point Sets and Atlas Construction. *IEEE Trans. PAMI*, 30(11):2011–2022.

Xu, L. and Jordan, M. I. (1996). On convergence properties of the em algorithm for gaussian mixtures. *Neural Comput.*, 8(1):129–151.

APPENDIX A

In this Appendix, we briefly describe how we build 3D contour fragments from stereo image sequences - so called '3D contour clouds'. However, we point out, that the method described in Section 4 will work on all kinds of contours independent of the reconstruction method, as long as the 3D contour fragments represent the shape of a category.

In our approach, we combine stereo correspondence on contour fragments and a robust Structure and Motion analysis. The '3D contour cloud' stereo reconstruction process consists of several steps:

1. **Image Acquisition - Dataset Generation.** We capture several stereo videos of different hand-held objects which are manipulated in front of a stereo rig (see an example in Section 5.2).
2. **Preprocessing.** To reconstruct just contours of the hand-held objects and not contours of the

hand, we first have to mask the hand in the stereo videos of hand-held objects. Here, we use a segmentation algorithm based on variational methods (Unger et al., 2009), which gives us a precise hand segmentation. Features that belong to the hand are subsequently ignored.

3. **Contour Fragment Extraction.** We apply the Canny edge detection algorithm (subpixel accuracy) to extract contour fragments in the left and the right frame of a stereo frame pair. Then, a linking algorithm based on smoothing constraints (Leordeanu et al., 2007) is used to obtain long, connected 2D contour fragments.
4. **Stereo Correspondence.** For the reconstruction of 3D contour fragments we need to find corresponding 2D contour fragments and point correspondences on them. We identify stereo correspondences in two steps. First, we match contour fragments in the right and the left stereo frame pair. Next, we compute point correspondences on these matches. In this step we combine 2D Shape Context (Belongie et al., 2002) and epipolar information, which is available by the stereo calibration.
5. **Stereo Reconstruction.** Based on the contour point correspondences we reconstruct the 3D contour fragments using the 'Object Space Error for General Camera Models' (Schweighofer et al., 2008).
6. **S+M Analysis.** The robust S+M analysis uses point features and is based on the work by (Schweighofer et al., 2008).