Learning shapes as directed closed surfaces
Technical Report

Sandor Szedmak
IIS, University of Innsbruck
sandor.szedmak@uibk.ac.at

Hanchen Xiong
IIS, University of Innsbruck
hanchen.xiong@uibk.ac.at

Justus Piater
IIS, University of Innsbruck
justus.piater@uibk.ac.at

December 19, 2011

1 Introduction

One of the central problems in capturing the environment by a robot is to interpret the objects observed. This interpretation can serve as a starting point to the potential activities. For example: can those objects be grasped, moved and reordered? The interpretation should not stick a label to those objects saying: this is a chair or that is a mug, but it should provide knowledge enough to act in a proper way. A mug or a glass, even a bottle, can be grasped in the same way, thus some common properties are really relevant among those objects, but others, e.g. colors, texture, some details of the shape can be ignored. Some parts, segments of the entire shape of the objects, carry activity-related properties that need to be captured.

The shapes of an object as an entity can not be directly observed by the known machine vision systems. Those systems can yield several different local feature items and the task is to build an abstract shape out of those features. Within that procedure we need to discover how those local items can relate to each other, what is the three dimensional graph connecting them, and recognize those items which can characterize the shape and separate them from those which can relate to something else, e.g. texture.

In learning shapes we assume features which can be characterized by two properties, a 3D position and an orientation, e.g. a surface segment, a patch, etc. These properties can be translated relatively easily into the properties of the potential activities. To collect this kind of features we need proper vision systems which can provide them with sufficient accuracy. Here we assume that these features are available for shape learning.

1.0.1 Shape model

We assume that the shape can be described as a manifold in the three dimensional space. This manifold, we might say surface, is an almost everywhere smooth one allowing to model edges and corners with high curvature, but otherwise it can be partitioned into relatively large connected smooth segments. This assumption expresses the need to eliminate irrelevant small details. Another requirement we should satisfy relates to the potential complexity of the shape, namely it can be a topologically higher order manifold with holes, with a mixture of segments with positive and negative curvature, and convex and concave parts.

To model a surface with complex structure and in the same time forcing a certain high level of smoothness we apply an infinite dimensional parametric representation exploiting the fact that a complex low dimensional manifold can be approximated by a hyperplane in a sufficiently high dimensional space.

The representation space we have chosen is an infinite dimensional Hilbert space of the square integrable functions. Within this space we can apply the probability density functions as features defined on the low dimensional 3D space to be modeled. This mathematical framework allows us to synthesize the probabilistic generative models and the robustness of the maximum margin based discriminative methods. Furthermore, the advantage of the kernel methods in expressing nonlinear relations can be exploited as well. The discrimination happens between the shape and the non-shape points, and the generative, density function based features provide certain local confidence measures on the shape approximation.

The shape modeling is considered as a machine learning procedure where the shape is extracted from local vision features. The learning task is to force a certain type of manifold to closely fit to the parameters, position and orientation, of the visual feature items, and in the same time it has to be as smooth, say simple, as possible which can be achieved via regularization constraining the complexity of the manifold applied.

The outcome of the learning method is an infinite dimensional vector, a combination of probability density functions. This kind of representation admits direct comparison of different objects to express their similarities and dissimilarities. This representation can be reused in other learning method to discover common parts within a given group of object, e.g. by applying Kernel Principal Component Analysis.

The derived shape models can be transformed by any affine transformation, e.g. translation or rotation, via acting on the parameters, expected values and/or covariance matrices, of the density functions used in the expression of the vectors describing the models.

The robot activities, e.g. grasping, can be modeled in a similar framework, thus both the shape models and the actions applied on those shapes

as abstract vectors can be located in a common vector space. In this way potential connections between shapes and actions can be predicted for a new shape and for a new action. To do that the relationships between the novel items and the known ones needs to computed in the common vector space.

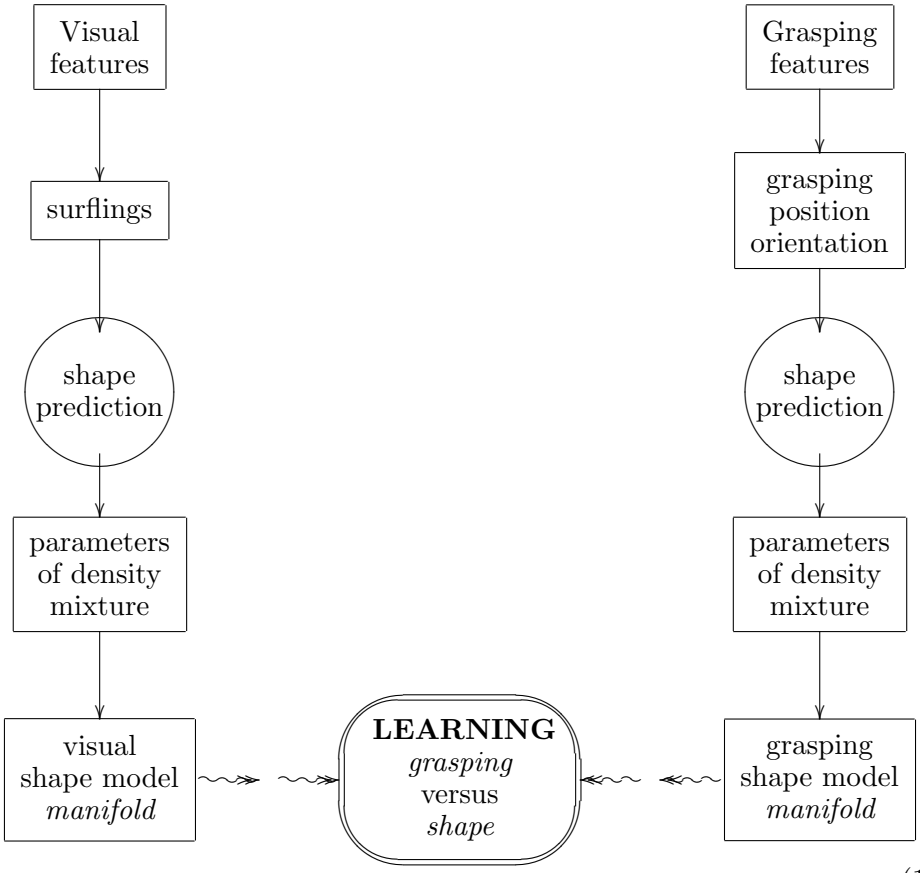
2 Learning task

We are facing the following learning task; given a set of 3D objects characterized by feature representation of different sources

- some visual features, e.g. collection of edges, texlets, surflings, see details about these features in [4],
- grasping properties, e.g. grasp densities,

and based on these data sources we need to learn that how to predict the grasp densities from the visual features. The inverse prediction could provide information to the generalization of the properties of an object to be important in grasping, but in the first case it is not as central as predicting the grasp densities.

To solve this learning task a two-phase model will be introduced; in the first phase so called shape model is computed of a sample of visual features. The parameters derived from the shape model is then used as feature representation of the object to predict the grasp densities. In the first case the shape model can be interpreted as a mixture of density functions, however the optimization framework allows us to further generalize the model, see in Section 5. The shape model can be applied on the grasp densities, since they are given by entities similar to the surflings type visual features, i.e. they are given by position and orientation. The two-phase model is summarized by (1)



(1)

Here we will focus on the surfling type visual features. These features constitute a collection of approximate tangent plane segments of the surface of the 3D objects. The centers and the normal vectors of these segments can be exploited to reproduce the entire surface. In this way the first task in the full learning procedure is to learn these surfaces. To this end we need to create a model of these surfaces such that

- the parameter vectors to be derived of the surface of each object have to live in the same space and in this way they can be compared, and distances, similarity measures can be computed between them,
- the parameter space needs to be sufficiently reach to express the potential complexity of the surfaces,
- since the surflings are only approximation of the real surface, therefore the parameter space should allow to estimate the confidence of the surface model.

To fulfill these requirements the representation space of the surfaces is chosen as a linear vector space containing the probability densities functions.

In this space a surface is expressed as a mixture of densities. The base family of these densities can be chosen as multivariate Gaussians but any other family which allows computationally feasible representation can be considered.

Remark 1. *In this model we are working with linear combination of probability densities that might produce negative probabilities, but this issue is rather technical and will not influence the learning model itself.*

After fitting the surflings based surface model to the objects we can apply the derived surface representations to learn how these features relate to the grasp densities. If in both cases, the surfaces and the grasp densities, are expressed as probability mixture models then the relationships can be revealed not only between the entire models but their parts as well. This kind of analysis can compare the contribution of the elements of the bases spanning the corresponding feature spaces since the elements of these bases, e.g. Gaussian densities, are common among the spaces.

In what follows we assume that there is a preprocessing step of surflings which can separate the surflings of an object from the surflings of the occasional background, hence an object related collection of surflings expresses the properties of an object and only that.

3 Shape model

The shape model of three dimensional objects is built upon the following assumptions:

- The shape $\mathscr{S}$ of an object $\mathscr{O}$ can be expressed by a smooth manifold $\mathcal{M}$ embedded into a Euclidean ambient space $\mathcal{X}$. The dimension of the ambient space is denoted by n.
- The manifold $\mathcal{M}$ is supposed to be closed and all points of the object $\mathscr{S}$ fall inside, with respect to a given orientation of the manifold, or on the manifold.

A smooth manifold can be characterized by an *atlas*, a collection of the pairs $\{U_\alpha, \varphi_\alpha\}$, where $\{U_\alpha\}$, called charts, is a set of open sets covering $\mathcal{M}$, and $\{\varphi_\alpha\}$ is a set of mappings such that for each α $\varphi_\alpha : U_\alpha \to \mathbb{R}^n$, i.e. they map the open sets of the manifold into open sets of a Euclidean space, and these maps and their inverses are differentiable. A surfling at a given point $\mathbf{x}$ of the manifold $\mathcal{M}$ can be interpreted as a segment of the tangent plane at $\mathbf{x}$. A tangent plane is a local linearization of the manifold and it is built upon the charts covering the point $\mathbf{x}$. The definition and properties of the tangent plane can be found for example in [3].

The normal vector to the tangent space can be defined in the ambient space $\mathcal{X}$ as the normal vector of the corresponding tangent plane in $\mathcal{X}$. Another, a more general, way of that which eliminates the need of the ambient space, by considering the normal vector as an element of the linear functionals defined on the tangent plane at a given point of the manifold. The orthogonality then can be expressed by setting the value of these functionals to 0 on all vectors of the corresponding tangent vectors. The elements of the set of these linear functionals are generally called as cotangent vectors.

4 Shape learning

There are given a sample set $\mathcal{S}$ of vector pairs $\{(\mathbf{x}_i, \mathbf{v}_i)\}, i = 1, \ldots, m$, where for each i $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^n$, is a vector assigned to a point of the manifold $\mathcal{M}$ in the ambient space $\mathcal{X}$, and $\mathbf{v}_i \in \mathbb{R}^n$ is a vector, the normal vector of the tangent space of $\mathcal{M}$ at $\mathbf{x}_i$ in the ambient space $\mathcal{X}$. We may refer to the pairs $(\mathbf{x}_i, \mathbf{v}_i)$ as surface elements as well. These surface elements are fundamentally based on a sub-sample of charts covering the manifold $\mathcal{M}$

Let $\phi_s : \mathcal{X} \to \mathcal{H}_s$ be a feature representation of the elements of the manifold $\mathcal{M}$ in a Hilbert space $\mathcal{H}_s$. The inner product of $\mathcal{H}_s$ will be expressed by the kernel function $\kappa : \mathcal{H}_s \times \mathcal{H}_s \to \mathbb{R}$. Here we allow to map all vectors of the ambient space of the manifold into the feature space to avoid some technical difficulties.

In the sequel we use the notations $\langle\, , \rangle_X$ and $\langle\, , \rangle_H$ to denote, and to distinct, the inner products in spaces $\mathcal{X}$ and in $\mathcal{H}_s$ respectively. In some cases the subscript is omitted that refers to the inner product in the feature space $\mathcal{H}_s$.

Suppose the manifold $\mathcal{M}$ can be well approximated by surface of $\mathbb{R}^n$, and this surface can be embedded as a hyperplane into the space $\mathcal{H}_s$, thus the implicit function

$$F(x) = \langle \mathbf{u}, \boldsymbol{\phi_s}(x) \rangle_H - 1 = 0 \tag{2}$$

describes the surface, where the vector $\mathbf{u} \in \mathcal{H}_s$ gives the parametrization, and since it is as element of Hilbert space it can be used as identifier of a shape entity.

If the parameter vector $\mathbf{u}$ is given then the manifold can be recovered by inverting the hyperplane from the feature space back into the space $\mathcal{X}$.

4.1 Including normal vectors

To exploit the information coded into normal vectors $\{\mathbf{v}_i\}$ of the surface elements we need to force that for every i the vector $\mathbf{v}_i$ is orthogonal or at least approximately orthogonal to the tangent plane of $\mathcal{M}$ at $\mathbf{x}_i$. If the surface corresponding to the smooth manifold $\mathcal{M}$ is given in $\mathcal{X}$ by the

implicit function $F(x) = 0$ then the direction of the normal vector at any $\mathbf{x} \in \mathcal{X}$ is given by the gradient of F

$$\nabla_x F = \left(\frac{\partial F}{\partial x_1}, \ldots, \frac{\partial F}{\partial x_n} \right), \tag{3}$$

where $(x_1, \ldots, x_n)$ the scalar components of the vector $\mathbf{x}$. To force the orthogonality between $\nabla_x F(x)|_{x=x_i}$ and $\mathbf{v}_i$ we have several alternatives, here three of them are enumerated those which lead to linear constraints.

- The following constraints imposes exact orthogonality on the function F

$$\nabla_x F|_{\mathbf{x}=\mathbf{x}_i} = \beta_i \mathbf{v}_i, \ i \in \{1, \ldots, m\} \tag{4}$$

  saying the vectors on left and the right hand sides have to be parallel.

- One can eliminate the the coefficients $\{\beta_i\}$ by another form

$$\nabla_x F|_{\mathbf{x}=\mathbf{x}_i} \wedge \mathbf{v}_i = 0 \ i \in \{1, \ldots, m\}, \tag{5}$$

  where $\wedge$ marks the exterior, sometimes called as wedge, product of two vectors. Here we exploited the fact that the exterior product of any two parallel vectors is equal to $\mathbf{0}$.

- The strict orthogonality constraints can be relaxed by suitable approximations

$$\langle \nabla_x F|_{\mathbf{x}=\mathbf{x}_i}, \mathbf{v}_i \rangle_X \geq D, \ i \in \{1, \ldots, m\}, \tag{6}$$

  where D is a positive lower bound of the inner products. Since the inner product can attain its maximum value when the estimated normal vectors of the manifold are parallel to the given set of observed normal vectors, therefore maximizing the lower bound forces the corresponding normal vectors to be closely and uniformly parallel. This type of constraint fundamentally forces the directional derivatives of F in the directions given by $\{\mathbf{v}_i\}$ to be large, and the maximum can be obtained when the vectors are parallel within the inner product.

4.2 Optimization problem

The assumption that the manifold covering an object is closed means that the points of the object is within or on the surface of the volume for which the manifold is the collection of the boundary points. This fact can be expressed as one-class classification problem where the points of the object constituting the class are separated from all other parts of the space containing the object,

therefore the surface can be modeled by the following optimization problem:

$$\begin{array}{lll} \min & \frac{1}{2}\|\mathbf{u}\|_2^2 + C_\xi \mathbf{1}'\boldsymbol{\xi} + C_\eta \mathbf{1}'\boldsymbol{\eta} & \\ \text{w.r.t.} & \mathbf{u} \in \mathcal{H}_s,\ \boldsymbol{\xi} \in \mathbb{R}^m, ,\ \boldsymbol{\eta} \in \mathbb{R}^m, & \\ \text{s.t.} & \langle \mathbf{u}, \boldsymbol{\phi_s}(\mathbf{x}_i)\rangle_H \geq 1 - \xi_i, & \#\# \text{ fitting the points} \\ & \langle \nabla_x F(x)|_{\mathbf{x}=\mathbf{x}_i}, \mathbf{v}_i\rangle_X \geq E - \eta_i, & \#\# \text{ fitting the normal vectors} \\ & \xi_i \geq 0,\ \eta_i \geq 0,\ i \in \{1,\ldots,m\}, & \end{array} \tag{7}$$

where $E > 0$ is margin scaling parameter for trading between the fitting of the surface points and the normal vectors. This formulation is short symbolic summary of the manifold approximation. To the one-class classification problem one can find introduction in [7] and applications for complex structured learning problems [1], [10], [5] and [11]. An approach similar to that which is presented here is published in [9] and [8]. In the [9] a one-class classification approach is mentioned as well. The main differences between that and our approach can be summarized in two points:

- The incorporation of the normal vectors into the surface approximation is carried out by a maximum margin based regression technique, see further details in Section . This approach allows us to include other characteristic properties of the surface, e.g. curvature, via kernelization.
- The representation of the surface elements is built upon infinite dimensional functional features. This representation can express a probabilistic model which can provide confidence estimation as well.

To transform (7) into a computable form the gradients $\nabla_x F(x)|_{\mathbf{x}=\mathbf{x}_i}$ need to be unfolded. To this end we need to compute the derivative

$$\begin{aligned} \nabla_x F(x) &= D_x(F(x))' = D_x(\langle \mathbf{u}, \boldsymbol{\phi_s}(\mathbf{x})\rangle)' \\ &= D_x(\boldsymbol{\phi_s}(\mathbf{x}))'\mathbf{u}. \end{aligned} \tag{8}$$

If we assume that the dimension of the feature space is finite then $D_x(\boldsymbol{\phi_s}(\mathbf{x}))$ is equal to the Jacobian matrix of the partial derivatives of the vector valued function $\boldsymbol{\phi_s}(\mathbf{x})$ with respect to the vector $\mathbf{x}$. An approach to handling the infinite case is described in Section 4.3.

The primal problem, (7), can be written as

$$\begin{array}{lll} \min & \frac{1}{2}\|\mathbf{u}\|_2^2 + C_\xi \mathbf{1}'\boldsymbol{\xi} + C_\eta \mathbf{1}'\boldsymbol{\eta} & \\ \text{w.r.t.} & \mathbf{u} \in \mathcal{H}_s,\ \boldsymbol{\xi} \in \mathbb{R}^m, ,\ \boldsymbol{\eta} \in \mathbb{R}^m, & \\ \text{s.t.} & \langle \mathbf{u}, \boldsymbol{\phi_s}(\mathbf{x}_i)\rangle \geq 1 - \xi_i, & \#\# \text{ fitting the points} \\ & \langle D_x(\boldsymbol{\phi_s}(\mathbf{x}))'|_{x=x_i}\mathbf{u}, \mathbf{v}_i\rangle_X \geq E - \eta_i, & \#\# \text{ fitting the normal vectors} \\ & \xi_i \geq 0,\ \eta_i \geq 0,\ i \in \{1,\ldots,m\}. & \end{array} \tag{9}$$

The solution to this problem can be derived from the Karush-Kuhn-Tucker(KKT) conditions, see details in [2] and references therein. Let the following Lagrangian coefficients be introduced for all i:

$$\begin{array}{rcl} \alpha_i & : & \langle \mathbf{u}, \boldsymbol{\phi_s}(\mathbf{x}_i)\rangle \geq 1-\xi_i, \\ \beta_i & : & \langle D_x(\boldsymbol{\phi_s}(\mathbf{x}))'|_{x=x_i}\mathbf{u}, \mathbf{v}_i\rangle_X \geq E-\eta_i, \\ \gamma_i & : & \xi_i \geq 0, \\ \delta_i & : & \eta_i \geq 0 \end{array} \tag{10}$$

Since the constraints are inequalities all Lagrangians have to be nonnegative. The Lagrangian functional of (9) reads as

$$\begin{array}{ll} L(\mathbf{u},\boldsymbol{\xi},\boldsymbol{\eta},\boldsymbol{\alpha},\boldsymbol{\beta}) & = \frac{1}{2}\langle \mathbf{u},\mathbf{u}\rangle + C_\xi \mathbf{1}'\boldsymbol{\xi} + C_\eta \mathbf{1}'\boldsymbol{\eta} \\ & - \sum_{i=1}^m \alpha_i \langle \mathbf{u}, \boldsymbol{\phi_s}(\mathbf{x}_i)\rangle + \sum_{i=1}^m \alpha_i - \sum_{i=1}^m \alpha_i\xi_i \\ & - \sum_{i=1}^m \beta_i \langle D_x(\boldsymbol{\phi_s}(\mathbf{x}))'|_{x=x_i}\mathbf{u}, \mathbf{v}_i\rangle_X + E\sum_{i=1}^m \beta_i - \sum_{i=1}^m \beta_i\eta_i \\ & - \sum_{i=1}^m \gamma_i\xi_i - \sum_{i=1}^m \delta_i\eta_i \\ \text{s.t.} & \alpha_i \geq 0,\ \beta_i \geq 0,\ \gamma_i \geq 0,\ \delta_i \geq 0\ i=1,\ldots,m. \end{array} \tag{11}$$

The partial derivatives of the Lagrangian with respect to the primal variables and the corresponding KKT conditions are given by

$$\begin{array}{ll} \dfrac{\partial L(\mathbf{u},\boldsymbol{\xi},\boldsymbol{\eta},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\delta})}{\partial \mathbf{u}} & = \mathbf{u} - \sum_{i=1}^m \alpha_i \boldsymbol{\phi_s}(\mathbf{x}_i) - \sum_{i=1}^m \beta_i D_x(\boldsymbol{\phi_s}(\mathbf{x}))'|_{x=x_i}\mathbf{v}_i = 0, \\ \dfrac{\partial L(\mathbf{u},\boldsymbol{\xi},\boldsymbol{\eta},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\delta})}{\partial \boldsymbol{\xi}} & = C_\xi \mathbf{1} - \boldsymbol{\alpha} - \boldsymbol{\gamma} = 0, \\ \dfrac{\partial L(\mathbf{u},\boldsymbol{\xi},\boldsymbol{\eta},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\delta})}{\partial \boldsymbol{\eta}} & = C_\eta \mathbf{1} - \boldsymbol{\beta} - \boldsymbol{\delta} = 0. \end{array} \tag{12}$$

Thus we have

$$\begin{array}{ll} \mathbf{u} & = \sum_{i=1}^m \alpha_i \boldsymbol{\phi_s}(\mathbf{x}_i) + \sum_{i=1}^m \beta_i D_x(\boldsymbol{\phi_s}(\mathbf{x}))'|_{x=x_i}\mathbf{v}_i, \\ \boldsymbol{\alpha} & \leq C_\xi \mathbf{1}, \\ \boldsymbol{\beta} & \leq C_\eta \mathbf{1}, \end{array} \tag{13}$$

where in the last two lines the nonnegativity of the components of $\boldsymbol{\gamma}$ and $\boldsymbol{\delta}$ are exploited. After replacing primal variables in the Lagrangian functional with the expressions containing only the Lagrangians we have the dual problem of (9) where the maximization is turned into minimization.

$$\begin{array}{ll} \min & \frac{1}{2}\begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix}' \overbrace{\begin{bmatrix} \mathbf{K}_{\alpha,\alpha} & \mathbf{K}_{\alpha,\beta} \\ \mathbf{K}_{\beta,\alpha} & \mathbf{K}_{\alpha,\alpha} \end{bmatrix}}^{\text{kernel matrix}} \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} - \begin{bmatrix} \mathbf{1} \\ E\mathbf{1} \end{bmatrix}' \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} \\ \text{w.r.t.} & \boldsymbol{\alpha} \in \mathbb{R}_+,\ \boldsymbol{\beta} \in \mathbb{R}_+, \\ \text{s.t.} & \mathbf{0} \leq \boldsymbol{\alpha} \leq C_\xi \mathbf{1}, \\ & \mathbf{0} \leq \boldsymbol{\beta} \leq C_\eta \mathbf{1}, \end{array} \tag{14}$$

The submatrices of the kernel matrix are obtained by

$$\begin{aligned}
(\mathbf{K}_{\alpha,\alpha})_{ij} &= \langle \boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{x}_i), \boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{x}_j) \rangle \,,\ i,j \in \{1,\dots,m\}, \\
(\mathbf{K}_{\alpha,\beta})_{ij} &= \langle \boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{x}_i), D_x(\boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{x}))'|_{x=x_j}\mathbf{v}_j \rangle \,,\ i,j \in \{1,\dots,m\}, \\
(\mathbf{K}_{\beta,\alpha})_{ij} &= [\mathbf{K}_{\alpha,\beta}]_{ji},\ i,j \in \{1,\dots,m\}, \\
(\mathbf{K}_{\beta,\beta})_{ij} &= \langle D_x(\boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{x}))'|_{x=x_i}\mathbf{v}_i, D_x(\boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{x}))'|_{x=x_j}\mathbf{v}_j \rangle \,,\ i,j \in \{1,\dots,m\}.
\end{aligned} \tag{15}$$

4.3 Evaluation of the kernels

When we are going to represent the vectors of the ambient space $\mathcal{X}$ we need to choose a feature space in which a complex surface of $\mathcal{X}$ can be approximated with high fidelity by a hyperplane. A candidate space could be the so called "functional feature" space where each feature vector is represented by a function. These spaces are generally infinite dimensional, thus very high flexibility can be guaranteed.

To realize an infinite dimensional feature space the following construction is proposed. Let $F : \mathcal{X} \times \mathcal{X} \times \Theta \to \mathbb{R}$ be a real valued function equipped with these properties:

1. F is a nonnegative function,
2. F is square integrable on its full domain,
3. For a fixed $\mathbf{x} \in \mathcal{X}$ and $\theta \in \Theta$

$$\int_{\mathcal{X}} F(\mathbf{t}, \mathbf{x}, \theta) d\mathbf{t} = 1. \tag{16}$$

One can consider function F as a probability density function defined on $\mathcal{X}$ and parametrized on the sets $\mathcal{X}$ and Θ. The parameters taken of $\mathcal{X}$ can be interpreted as localization, e.g. mean, and the parameter θ as scale, e.g. variance. After taking the second and third variables as parameters in F, we can define the following class of functions

$$\mathcal{F} = \{f | f : \mathcal{X} \to \mathbb{R},\ f(t) = F(t, \mathbf{x}, \theta), \mathbf{t} \in \mathcal{X}, \mathbf{x} \in \mathcal{X}, \theta \in \Theta\}. \tag{17}$$

We might denote these functions for a parameter pair $\mathbf{x}$ and θ by $f(.|\mathbf{x}, \theta)$.

Now the feature mapping is given by

$$\boldsymbol{\phi}_{\boldsymbol{s}} : \mathcal{X} \to \mathcal{F}, \tag{18}$$

and defined via the formula

$$\boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{x}) = f(.|\mathbf{x}, \theta), \forall \mathbf{x} \in \mathcal{X}, \tag{19}$$

thus the elements of the original ambient space are used as localization of the corresponding density functions, and the scale parameter, θ, is shared among all these densities.

We need to emphasize that the feature mapping is a function valued function.

The value of the function $\boldsymbol{\phi}$ at $\mathbf{t}$ is denoted by $\boldsymbol{\phi}(\mathbf{t}|\mathbf{x})$, where for sake of simplicity the parameter θ which is fixed for all $\mathbf{t}$ and $\mathbf{x}$ is omitted.

4.4 Representation by Gaussian densities

To compute the elements of the kernel matrix in (15) we need to assign concrete representations to the points in the ambient space. Let the feature representation be chosen from the family of the multivariate Gaussian probability density functions

$$\boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{t}|\mathbf{x}) = f(t|\mathbf{x},\theta) = \frac{1}{(2\pi)^{n/2}\det(\theta)^{1/2}} e^{-\frac{1}{2}(\mathbf{t}-\mathbf{x})'\theta^{-1}(\mathbf{t}-\mathbf{x})}, \tag{20}$$

where $\mathbf{x}$ serves as mean vector and θ as covariance matrix. To force the parsimony of our model the covariance matrix is supposed to be diagonal, and all diagonal elements are equal to σ^2, therefore we have

$$\begin{aligned} \boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{t}|\mathbf{x}) = f(t|\mathbf{x},\theta) &= \frac{1}{(2\pi)^{n/2}\sigma^n} e^{-\frac{1}{2\sigma^2}\langle \mathbf{t}-\mathbf{x},\mathbf{t}-\mathbf{x}\rangle_X} \\ &= \frac{1}{(2\pi)^{n/2}\sigma^n} e^{-\frac{1}{2\sigma^2}\|\mathbf{t}-\mathbf{x}\|2} \\ &= \frac{1}{(2\pi)^{n/2}\sigma^n} e^{-\frac{\sum_{r=1}^{n}(t_r-x_r)^2}{2\sigma^2}}. \end{aligned} \tag{21}$$

The differential operator for general multivariate Gaussian case reads as

$$\begin{aligned} &D_x\boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{t}|\mathbf{x}) = D_xF(\mathbf{t},\mathbf{x},\theta) = \frac{\partial F(\mathbf{t},\mathbf{x},\theta)}{\partial \mathbf{x}} \\ &= \frac{\partial\left(\frac{1}{(2\pi)^{n/2}\det(\theta)^{1/2}} e^{-\frac{1}{2}(\mathbf{t}-\mathbf{x})'\theta^{-1}(\mathbf{t}-\mathbf{x})}\right)}{\partial \mathbf{x}} \\ &= \frac{1}{(2\pi)^{n/2}\det(\theta)^{1/2}} e^{-\frac{1}{2}(\mathbf{t}-\mathbf{x})'\theta^{-1}(\mathbf{t}-\mathbf{x})} \otimes \theta^{-1}(\mathbf{t}-\mathbf{x}) \\ &= \boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{t}|\mathbf{x}) \otimes \theta^{-1}(\mathbf{t}-\mathbf{x}), \end{aligned} \tag{22}$$

and in case of the reduced diagonal case we have

$$D_x\boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{t}|\mathbf{x}) = \frac{1}{\sigma^2}\boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{t}|\mathbf{x}) \otimes (\mathbf{t}-\mathbf{x}), \tag{23}$$

where we need to be aware on the fact that $\boldsymbol{\phi}_{\boldsymbol{s}}(\mathbf{t}|\mathbf{x})$ is a function of $\mathbf{t}$ as well.

Before the inner products are computed some notations and reformulations are being introduced. For sake of simplicity the following abbreviation is introduced

$$C_G = \frac{1}{(2\pi)^{n/2}\sigma^n}. \tag{24}$$

We are going to exploit the well known identity connecting the tensor and inner products, namely

$$\langle \otimes_{r=1}^{n} \mathbf{u}_r, \otimes_{r=1}^{n} \mathbf{v}_r \rangle = \prod_{r=1}^{n} \langle \mathbf{u}_r, \mathbf{v}_r \rangle , \tag{25}$$

further details can be found in Appendix A.

The x_{ir} denotes the rth component of the vector $\mathbf{x}_i$ and similar notation is used for the vectors $\mathbf{v}_i$ as well.

The next simple assertion can eliminate plenty of technical details of the kernel derivation.

Lemma 2. *Assuming that the feature representation given in (21) then the point wise product of any two feature vectors can be expressed as a product of two functions*

$$\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i)\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j) = g(\|\mathbf{x}_i, -\mathbf{x}_j\|_2, \sigma)h(\mathbf{t}, (\mathbf{x}_i + \mathbf{x}_j)/2, \sigma/2^{1/2}), \tag{26}$$

such that the function g depends only on the distance $\|\mathbf{x}_i, -\mathbf{x}_j\|_2$ and scale σ but not on $\mathbf{t}$, and the function h is a multivariate Gaussian density function defined on the domain $(\mathbf{t} \in)\mathcal{X}$ with mean $\frac{\mathbf{x}_i, +\mathbf{x}_j}{2}$ and with a diagonal covariance matrix with the same diagonal elements being equal to $\sigma^2/2$.

Proof. The proof is based on a straightforward reformulation of the point wise product, namely

$$\begin{aligned}
&\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i)\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j) = C_G^2 e^{-\frac{\|\mathbf{t}-\mathbf{x}_i\|^2}{2\sigma^2}} e^{-\frac{\|\mathbf{t}-\mathbf{x}_j\|^2}{2\sigma^2}} \\
&= C_G^2 e^{-\frac{2\|\mathbf{t}-\frac{\mathbf{x}_i+\mathbf{x}_j}{2})\|^2 + \frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2}}{2\sigma^2}} \\
&= C_G^2 e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{4\sigma^2}} e^{-\frac{\|\mathbf{t}-\frac{\mathbf{x}_i+\mathbf{x}_j}{2})\|^2}{\sigma^2}} \\
&= C_G^2 \frac{(2\pi)^{n/2}\sigma^n}{2^{n/2}} e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{4\sigma^2}} \frac{2^{n/2}}{(2\pi)^{n/2}\sigma^n} e^{-\frac{\|\mathbf{t}-(\frac{\mathbf{x}_i+\mathbf{x}_j}{2})\|^2}{2(2^{-1/2}\sigma)^2}} \\
&= \underbrace{\frac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n} e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{4\sigma^2}}}_{g(\|\mathbf{x}_i,-\mathbf{x}_j\|_2,\sigma)} \underbrace{\frac{2^{n/2}}{(2\pi)^{n/2}\sigma^n} e^{-\frac{\|\mathbf{t}-(\frac{\mathbf{x}_i+\mathbf{x}_j}{2})\|^2}{2(2^{-1/2}\sigma)^2}}}_{h(\mathbf{t},(\mathbf{x}_i,+\mathbf{x}_j)/2,\sigma/2^{1/2})},
\end{aligned} \tag{27}$$

where the last line shows the decomposition claimed. □

From this statement we can conclude that

Corollary 3. *The inner product between any two feature vectors can be computed by*

$$\begin{aligned}
&\langle \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle = \textstyle\int_{\mathcal{X}} \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i)\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)d\mathbf{t} \\
&= \tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n} e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{4\sigma^2}} \underbrace{\int_{\mathcal{X}} \frac{2^{n/2}}{(2\pi)^{n/2}\sigma^n} e^{-\frac{\|\mathbf{t}-(\frac{\mathbf{x}_i+\mathbf{x}_j}{2})\|^2}{2(2^{-1/2}\sigma)^2}} d\mathbf{t}}_{=1} \\
&= \tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n} e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2(2^{1/2}\sigma)^2}},
\end{aligned} \tag{28}$$

which is a Gaussian kernel function with scale, or width, parameter $2(2^{1/2}\sigma)^2$ *multiplied with the scalar* $\frac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n}$

It is worth mentioning that the inner product in Corollary 3 can be interpreted as a multivariate Gaussian density function if one of the parameters, $\mathbf{x}_i$ and $\mathbf{x}_j$, is taken as a variable and the other as mean.

4.4.1 Computation of kernel elements

In the derivation of the dual problem we end up with four types of sub-kernels, see in (15). Two of them are just transpose of each other, thus we need to deal with three types only.

- Based on Corollary 3 we have in the first case

$$(\mathbf{K}_{\alpha,\alpha})_{ij} = \langle \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle = \tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n} e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2(2^{1/2}\sigma)^2}}. \tag{29}$$

- The cross kernels between the two types of constraints relating to the positions and the surface normals can be computed by

$$\begin{aligned}
&(\mathbf{K}_{\alpha,\beta})_{ij} = \langle \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), D_x(\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}))'|_{x=x_j}\mathbf{v}_j\rangle \\
&= \langle \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \frac{1}{\sigma^2}\big[\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\otimes(\mathbf{t}-\mathbf{x}_j)'\big]\mathbf{v}_j\rangle \\
&= \frac{1}{\sigma^2}\langle \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle(\mathbf{t}-\mathbf{x}_j),\mathbf{v}_j\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle \\
&= \frac{1}{\sigma^2}\langle \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), (\langle\mathbf{t},\mathbf{v}_j\rangle_X - \langle\mathbf{x}_j,\mathbf{v}_j\rangle_X)\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle \\
&= \frac{1}{\sigma^2}\big(\langle \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle\mathbf{t},\mathbf{v}_j\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle - \langle \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle\mathbf{x}_j,\mathbf{v}_j\rangle_X\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle\big).
\end{aligned} \tag{30}$$

In computing this expression by parts we can exploit Corollary 3 again

$$\begin{aligned}
&\langle \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle\mathbf{x}_j,\mathbf{v}_j\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle = \langle\mathbf{x}_j,\mathbf{v}_j\rangle_X \textstyle\int_{\mathcal{X}} \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i)\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)d\mathbf{t} \\
&= \langle\mathbf{x}_j,\mathbf{v}_j\rangle_X \tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n} e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2(2^{1/2}\sigma)^2}},
\end{aligned} \tag{31}$$

and

$$\begin{aligned}&\langle \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{t}, \mathbf{v}_j\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle = \textstyle\int_{\mathcal{X}} \langle \mathbf{t}, \mathbf{v}_j\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i)\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)d\mathbf{t}\\ &= \langle \mathbf{v}_j, \textstyle\int_{\mathcal{X}} \mathbf{t}\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i)\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)d\mathbf{t}\rangle_X .\end{aligned} \tag{32}$$

After applying the decomposition of Lemma 2 note that the expression in the integral can be interpreted as an expected value computation

$$\begin{aligned}&\langle \mathbf{v}_j, \textstyle\int_{\mathcal{X}} \mathbf{t}\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i)\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)d\mathbf{t}\rangle_X\\ &= g(\|\mathbf{x}_i, -\mathbf{x}_j\|_2, \sigma)\, \langle \mathbf{v}_j, \textstyle\int_{\mathcal{X}} \mathbf{t}h(\mathbf{t}, (\mathbf{x}_i, +\mathbf{x}_j)/2, \sigma/2^{1/2})d\mathbf{t}\rangle_X\\ &= g(\|\mathbf{x}_i, -\mathbf{x}_j\|_2, \sigma)\, \langle \mathbf{v}_j, (\mathbf{x}_i, +\mathbf{x}_j)/2\rangle_X\\ &= \tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n} e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2(2^{1/2}\sigma)^2}} \langle \mathbf{v}_j, \tfrac{\mathbf{x}_i+\mathbf{x}_j}{2}\rangle_X .\end{aligned} \tag{33}$$

Then putting together the sub-expressions an element of the cross kernel is given by

$$(\mathbf{K}_{\alpha,\beta})_{ij} = \tfrac{1}{\sigma^2} \tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n} e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2(2^{1/2}\sigma)^2}} \langle \mathbf{v}_j, \tfrac{\mathbf{x}_i-\mathbf{x}_j}{2}\rangle_X . \tag{34}$$

- We have also the transpose of the previously computed sub-kernel.

$$(\mathbf{K}_{\beta,\alpha})_{ij} = [\mathbf{K}_{\alpha,\beta}]_{ji} = \tfrac{1}{\sigma^2} \tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n} e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2(2^{1/2}\sigma)^2}} \langle \mathbf{v}_i, \tfrac{\mathbf{x}_j-\mathbf{x}_i}{2}\rangle_X . \tag{35}$$

- The computation of kernel items relating to the normal vectors follows a schema resembling to those mentioned above.

$$\begin{aligned}&(\mathbf{K}_{\beta,\beta})_{ij} = \langle D_x(\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}))'|_{x=x_i}\mathbf{v}_i, D_x(\boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}))'|_{x=x_j}\mathbf{v}_j\rangle\\ &= \frac{1}{\sigma^4} \langle \langle (\mathbf{t}-\mathbf{x}_i), \mathbf{v}_i\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle (\mathbf{t}-\mathbf{x}_j), \mathbf{v}_j\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle\\ &= \frac{1}{\sigma^4} \langle \langle (\mathbf{t}-\mathbf{x}_i), \mathbf{v}_i\rangle_X \langle (\mathbf{t}-\mathbf{x}_j), \mathbf{v}_j\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle\\ &= \frac{1}{\sigma^4} \Big[\langle \langle \mathbf{t}, \mathbf{v}_i\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{t}, \mathbf{v}_j\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle\\ &\quad + \langle \langle \mathbf{x}_i, \mathbf{v}_i\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{x}_j, \mathbf{v}_j\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle\\ &\quad - \langle \langle \mathbf{t}, \mathbf{v}_i\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{x}_j, \mathbf{v}_j\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle\\ &\quad - \langle \langle \mathbf{x}_i, \mathbf{v}_i\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{t}, \mathbf{v}_j\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle \Big].\end{aligned} \tag{36}$$

Except the first term we can apply almost the same unfolding steps on the sub-expressions that have been used above thus we have

$$\begin{aligned}&\langle \langle \mathbf{x}_i, \mathbf{v}_i\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{x}_j, \mathbf{v}_j\rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle\\ &\quad = \langle \mathbf{x}_i, \mathbf{v}_i\rangle_X \langle \mathbf{x}_j, \mathbf{v}_j\rangle_X \langle \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j)\rangle\\ &\quad = \langle \mathbf{x}_i, \mathbf{v}_i\rangle_X \langle \mathbf{x}_j, \mathbf{v}_j\rangle_X \tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n} e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2(2^{1/2}\sigma)^2}},\end{aligned} \tag{37}$$

$$\begin{aligned}
&\langle \langle \mathbf{t}, \mathbf{v}_i \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{x}_j, \mathbf{v}_j \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j) \rangle \\
&\quad = \langle \mathbf{x}_j, \mathbf{v}_j \rangle_X \langle \langle \mathbf{t}, \mathbf{v}_i \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j) \rangle \\
&\quad = \langle \mathbf{x}_j, \mathbf{v}_j \rangle_X \langle \tfrac{\mathbf{x}_i + \mathbf{x}_j}{2}, \mathbf{v}_i \rangle_X \tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2(2^{1/2}\sigma)^2}}
\end{aligned} \tag{38}$$

and

$$\begin{aligned}
&\langle \langle \mathbf{x}_i, \mathbf{v}_i \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{t}, \mathbf{v}_j \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j) \rangle \\
&\quad = \langle \mathbf{x}_i, \mathbf{v}_i \rangle_X \langle \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{t}, \mathbf{v}_j \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j) \rangle \\
&\quad = \langle \mathbf{x}_i, \mathbf{v}_i \rangle_X \langle \tfrac{\mathbf{x}_i + \mathbf{x}_j}{2}, \mathbf{v}_j \rangle_X \tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2(2^{1/2}\sigma)^2}}.
\end{aligned} \tag{39}$$

The first term requires a little bit more care, where we have

$$\begin{aligned}
&\langle \langle \mathbf{t}, \mathbf{v}_i \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{t}, \mathbf{v}_j \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j) \rangle \\
&\quad = \textstyle\int_{\mathcal{X}} \langle \mathbf{t}, \mathbf{v}_i \rangle_X \langle \mathbf{t}, \mathbf{v}_j \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i) \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j) d\mathbf{t}.
\end{aligned} \tag{40}$$

If we apply the decomposition of Lemma 2 again and the identity relating to the inner product of tensor products, see (25), we receive the following chain of equalities

$$\begin{aligned}
&\langle \langle \mathbf{t}, \mathbf{v}_i \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{t}, \mathbf{v}_j \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j) \rangle \\
&= \langle (\mathbf{v}_i \otimes \mathbf{v}_j), \textstyle\int_{\mathcal{X}} (\mathbf{t} \otimes \mathbf{t}) \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i) \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j) d\mathbf{t} \rangle_{\text{Frob}} \\
&= \langle (\mathbf{v}_i \otimes \mathbf{v}_j), \textstyle\int_{\mathcal{X}} (\mathbf{t} \otimes \mathbf{t}) g(\|\mathbf{x}_i, -\mathbf{x}_j\|_2, \sigma) h(\mathbf{t}, (\mathbf{x}_i, +\mathbf{x}_j)/2, \sigma/2^{1/2}) d\mathbf{t} \rangle_{\text{Frob}} \\
&= g(\|\mathbf{x}_i, -\mathbf{x}_j\|_2, \sigma) \langle (\mathbf{v}_i \otimes \mathbf{v}_j), \textstyle\int_{\mathcal{X}} (\mathbf{t} \otimes \mathbf{t}) h(\mathbf{t}, (\mathbf{x}_i, +\mathbf{x}_j)/2, \sigma/2^{1/2}) d\mathbf{t} \rangle_{\text{Frob}}.
\end{aligned} \tag{41}$$

Note the integral expression is equal to the second, non-centralized, moment of the multivariate Gaussian variable with density function h. Based on the identity

$$\text{cov}(\mathbf{t}) = E(\mathbf{t} \otimes \mathbf{t}) - E(\mathbf{t}) \otimes E(\mathbf{t}) \tag{42}$$

which displays that how the covariance can be expressed by the first the second moments of vector valued random variables, thus we can write

$$\begin{aligned}
&\langle \langle \mathbf{t}, \mathbf{v}_i \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{t}, \mathbf{v}_j \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j) \rangle \\
&= g(\|\mathbf{x}_i, -\mathbf{x}_j\|_2, \sigma) \langle (\mathbf{v}_i \otimes \mathbf{v}_j), \text{cov}(\mathbf{t}) + E(\mathbf{t}) \otimes E(\mathbf{t}) \rangle_{\text{Frob}} \\
&= g(\|\mathbf{x}_i, -\mathbf{x}_j\|_2, \sigma) \langle (\mathbf{v}_i \otimes \mathbf{v}_j), \tfrac{\sigma^2}{2} \mathbf{I}_n + \tfrac{\mathbf{x}_i, +\mathbf{x}_j}{2} \otimes \tfrac{\mathbf{x}_i, +\mathbf{x}_j}{2} \rangle_{\text{Frob}},
\end{aligned} \tag{43}$$

where $\mathbf{I}_n$ denotes the n-dimensional identity matrix. Now we can reverse Identity 25

$$\begin{aligned}
&\langle \langle \mathbf{t}, \mathbf{v}_i \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_i), \langle \mathbf{t}, \mathbf{v}_j \rangle_X \boldsymbol{\phi_s}(\mathbf{t}|\mathbf{x}_j) \rangle \\
&= g(\|\mathbf{x}_i, -\mathbf{x}_j\|_2, \sigma) (\tfrac{\sigma^2}{2} \langle \mathbf{v}_i, \mathbf{v}_j \rangle_X + \langle \mathbf{v}_i, \tfrac{\mathbf{x}_i, +\mathbf{x}_j}{2} \rangle_X \langle \mathbf{v}_j, \tfrac{\mathbf{x}_i, +\mathbf{x}_j}{2} \rangle \\
&= (\tfrac{\sigma^2}{2} \langle \mathbf{v}_i, \mathbf{v}_j \rangle_X + \langle \mathbf{v}_i, \tfrac{\mathbf{x}_i, +\mathbf{x}_j}{2} \rangle_X \langle \mathbf{v}_j, \tfrac{\mathbf{x}_i, +\mathbf{x}_j}{2} \rangle_X) \tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2(2^{1/2}\sigma)^2}}.
\end{aligned} \tag{44}$$

After combining the sub-expressions we arrive at

$$
\begin{aligned}
(\mathbf{K}_{\beta,\beta})_{ij} &= \tfrac{1}{\sigma^4}\tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n}e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{4\sigma^2}} \\
&\Big(\langle \mathbf{x}_i,\mathbf{v}_i\rangle_X \langle \mathbf{x}_j,\mathbf{v}_j\rangle_X - \langle \mathbf{x}_j,\mathbf{v}_j\rangle_X \langle \tfrac{\mathbf{x}_i+\mathbf{x}_j}{2},\mathbf{v}_i\rangle_X - \langle \mathbf{x}_i,\mathbf{v}_i\rangle_X \langle \tfrac{\mathbf{x}_i+\mathbf{x}_j}{2},\mathbf{v}_j\rangle_X \\
&\quad + \tfrac{\sigma^2}{2}\langle \mathbf{v}_i,\mathbf{v}_j\rangle_X + \langle \mathbf{v}_i,\tfrac{\mathbf{x}_i,+\mathbf{x}_j}{2}\rangle_X \langle \mathbf{v}_j,\tfrac{\mathbf{x}_i,+\mathbf{x}_j}{2}\rangle_X \Big) \\
&= \tfrac{1}{\sigma^4}\tfrac{1}{(2\pi)^{n/2}(2^{1/2}\sigma)^n}e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2(2^{1/2}\sigma)^2}} (\tfrac{\sigma^2}{2}\langle \mathbf{v}_i,\mathbf{v}_j\rangle_X + \langle \tfrac{\mathbf{x}_j-\mathbf{x}_i}{2},\mathbf{v}_j\rangle_X \langle \tfrac{\mathbf{x}_i-\mathbf{x}_j}{2},\mathbf{v}_i\rangle_X).
\end{aligned} \tag{45}
$$

5 General description of the learning model

The learning task that we are going to solve is the following. There is a set, called sample, of pairs of output and input objects $\{(y_i, x_i) : y_i \in \mathcal{Y},\ x_i \in \mathcal{X},\ i = 1, \ldots, m, \}$ independently and identically chosen out of an unknown multivariate distribution $\mathcal{P}(Y, X)$. Here we would like to emphasize the input and output objects can be arbitrary, e.g. they may be graphs, matrices, functions, probability distributions etc. To these objects there are given two functions $\phi : \mathcal{X} \to \mathcal{H}_\phi$ and $\psi : \mathcal{Y} \to \mathcal{H}_\psi$ mapping the input and output objects respectively into linear vector spaces, called in the sequel, feature space in case of the inputs and label space when the outputs are considered.

The objective is to find a linear function acting on the feature space

$$f(\boldsymbol{\phi}(x)) = \mathbf{W}\boldsymbol{\phi}(x) + \mathbf{b} \tag{46}$$

and produces a prediction of every input object in the label space and in this way could implicitly give back a corresponding output object. Formally we have

$$y = \psi^{-1}(\psi(y)) = \psi^{-1}(f(\boldsymbol{\phi}(x))). \tag{47}$$

The learning procedure can be summarized by the following table:

Embedding	$\phi : \overbrace{\text{input space}}^{\mathcal{X}} \to \overbrace{\text{feature space}}^{\mathcal{H}_\phi},$ $\psi : \overbrace{\text{output space}}^{\mathcal{Y}} \to \overbrace{\text{label space}}^{\mathcal{H}_\psi},$
Similarity transformation	$\widetilde{\mathbf{W}} = (\mathbf{W}, \mathbf{b}) \Rightarrow \boldsymbol{\psi}(y) \sim \widetilde{\mathbf{W}}\boldsymbol{\phi}(x),$
Inversion	$\boldsymbol{\psi}^{-1} : \overbrace{\text{label space}}^{\mathcal{H}_\psi} \to \overbrace{\text{output space}}^{\mathcal{Y}}.$

In the framework of the Support Vector Machine the outputs represent two classes and the labels are chosen out of the set $y_i \in \{-1, +1\}$. The

aim is to find a separating hyperplane, via its normal vector, such that the distance between the elements of the two classes, called margin, is the possible largest measured in the direction of this normal vector. This base schema can be extended allowing some sample items to fall closer to the separating hyperplane than the margin. This is demonstrated on Figure 1

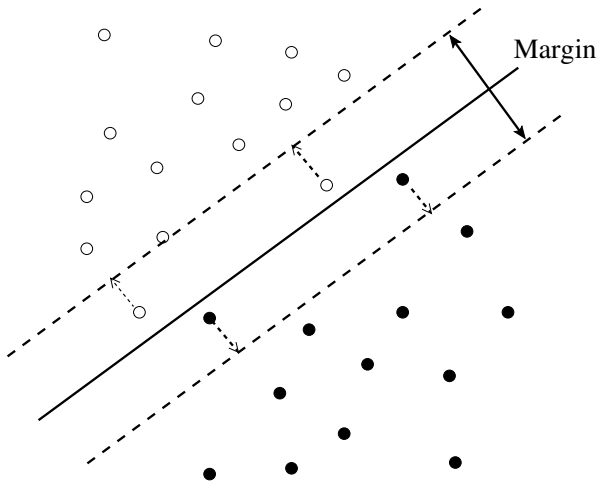


Figure 1: The schema of the Support Vector Machine. There are two classes that we are going to separate by using a hyperplane maximizing the distance between the classes and minimizing the potential errors

This learning scenario can be formulated as an optimization problem similar to this:

$$\begin{aligned}
&\min \quad \tfrac{1}{2}\boxed{\mathbf{w}'\mathbf{w}} + C\mathbf{1}'\boldsymbol{\xi} \\
&\text{w.r.t.} \quad \boxed{\mathbf{w} : \mathcal{H}_\phi \to \mathbb{R}}, \text{ normal vector} \\
&\qquad\quad \boxed{b \in \mathbb{R}}, \text{ bias} \\
&\qquad\quad \boldsymbol{\xi} \in \mathbb{R}^m, \text{ error vector} \\
&\text{s.t.} \quad \boxed{y_i(\mathbf{w}'\boldsymbol{\phi}(\mathbf{x}_i) + b)} \geq 1 - \xi_i \\
&\qquad\quad \boldsymbol{\xi} \geq \mathbf{0}, \ i = 1, \ldots, m.
\end{aligned}$$

5.1 Reinterpretation of the normal vector w

The normal vector **w** formally behaves as a linear transformation acting on the feature vectors which makes rise the idea to extend the capability of the original schema. This reinterpretation can be characterized briefly in the following way

SVM	ExtendedView
• $\mathbf{w}$ is the normal vector of the separating hyperplane.	• $\mathbf{W}$ is a linear operator projecting the feature space into the label space.
• $y_i \in \{-1, +1\}$ binary outputs.	• $y_i \in \mathcal{Y}$ arbitrary outputs
• The labels are equal to the binary objects.	• $\boldsymbol{\psi}(y_i) \in \mathcal{H}_\psi$ are the labels, the embedded outputs in a linear vector space

If we apply a one-dimensional normalized label space invoking binary labels $\{-1, +1\}$ in the general framework one can restore the original scenario of the SVM, and the normal vector is a projection into the one dimensional label space.

The extended form of the SVM tries to find an affine transformation which maps the configuration of the input items to gain the highest similarity between the image of the inputs and the outputs.

In summarizing the learning task we end up in the following optimization problem presented parallel with the original primal form of the SVM to emphasize the similarities and dissimilarities between the original and the extended form.

Primal problems for maximum margin learning

	Binary class learning	Vector label learning
	Support Vector Machine(SVM)	Maximum Margin Regression(MMR)
min	$\frac{1}{2}\underbrace{\boxed{\mathbf{w}'\mathbf{w}}}_{\|\mathbf{w}\|_2^2} + C\mathbf{1}'\boldsymbol{\xi}$	$\frac{1}{2}\underbrace{\boxed{\mathbf{tr}(\mathbf{W}'\mathbf{W})}}_{\|\mathbf{W}\|_F^2} + C\mathbf{1}'\boldsymbol{\xi}$
w.r.t.	$\boxed{\mathbf{w} : \mathcal{H}_\phi \to \mathbb{R},}$ normal vec. $\boxed{b \in \mathbb{R},}$ bias, $\boldsymbol{\xi} \in \mathbb{R}^m,$ error vector,	$\boxed{\mathbf{W} : \mathcal{H}_\phi \to \mathcal{H}_\psi,}$ linear operator, $\boxed{\mathbf{b} \in \mathcal{H}_\psi,}$ translation(bias), $\boldsymbol{\xi} \in \mathbb{R}^m,$ error vector,
s.t.	$\boxed{y_i(\mathbf{w}'\boldsymbol{\phi}(\mathbf{x}_i) + b)} \geq 1 - \xi_i,$ $\boldsymbol{\xi} \geq \mathbf{0},\ i = 1, \ldots, m,$	$\boxed{\langle \boldsymbol{\psi}(\mathbf{y}_i), \mathbf{W}\boldsymbol{\phi}(\mathbf{x}_i) + \mathbf{b} \rangle_{\mathcal{H}_\psi}} \geq 1 - \xi_i,$ $\boldsymbol{\xi} \geq \mathbf{0},\ i = 1, \ldots, m.$

In the extended formulation we exploit the fact the Frobenius norm and the Frobenius inner product correspond to the linear vector space of matrices with the dimension being equal to the number of elements of the matrices,

hence it gives an isomorphism between the space spanned by the normal vector of the hyperplane occurring in the SVM and the space spanned by the linear transformations.

One can recognize that if no bias term included in the MMR problem then we have a completely symmetric relationship between the label and the feature space via the representations of the input and the output items, namely

$$\langle \boldsymbol{\psi}(\mathbf{y}_i), \mathbf{W}\boldsymbol{\phi}(\mathbf{x}_i)\rangle_{\mathcal{H}_\psi} = \langle \mathbf{W}^*\boldsymbol{\psi}(\mathbf{y}_i), \boldsymbol{\phi}(\mathbf{x}_i)\rangle_{\mathcal{H}_\phi} = \langle \boldsymbol{\phi}(\mathbf{x}_i), \mathbf{W}^*\boldsymbol{\psi}(\mathbf{y}_i)\rangle_{\mathcal{H}_\phi}\,.$$

Thus, in predicting the input items as the image of the corresponding linear function defined on the outputs the adjoint of $\mathbf{W}$, $\mathbf{W}^*$, need to be used. This adjoint is equal to the transpose of the matrix representation of $\mathbf{W}$ when both the label space and the feature space are finite.

5.2 Dual problem

The dual problem of the MMR presented in the right column of (5.1) is given by

$$\begin{array}{ll}
\text{min} & \sum_{i,j=1}^m \alpha_i\alpha_j \overbrace{\langle \boldsymbol{\phi}(\mathbf{x}_i), \boldsymbol{\phi}(\mathbf{x}_j)\rangle}^{\kappa^\phi_{ij}} \overbrace{\langle \boldsymbol{\psi}(\mathbf{y}_i), \boldsymbol{\psi}(\mathbf{y}_j))\rangle}^{\kappa^\psi_{ij}} - \sum_{i=1}^m \alpha_i, \\
\text{w.r.t.} & \alpha_i \in \mathbb{R}, \\
\text{s.t.} & \sum_{i=1}^m (\boldsymbol{\psi}(\mathbf{y}_i))_t \alpha_i = 0,\ t = 1, \ldots, \dim(\mathcal{H}_\psi), \\
& 0 \le \alpha_i \le C,\ i = 1, \ldots, m. \\
\hline
\kappa^\phi_{ij} & \text{kernel items corresponding to the feature vectors,} \\
\kappa^\psi_{ij} & \text{kernel items corresponding to the label vectors}
\end{array}$$

The objective function contains no direct reference to the implicit representation either the label or the feature vectors, only the corresponding kernel elements appear. The symmetry of the objective function is clearly recognizable showing that the underlying problem without bias is completely reversible.

The constraints

$$\sum_{i=1}^m (\boldsymbol{\psi}(\mathbf{y}_i))_t \alpha_i = 0,\ t = 1, \ldots, \dim(\mathcal{H}_\psi) \tag{48}$$

appear in the dual only if the bias term is included into the primal model.

The explicit occurrences of the label vectors can be transformed into implicit ones by exploiting that the feasibility domain covered by the constraints:

$$\sum_{i=1}^m (\boldsymbol{\psi}(\mathbf{y}_i))_t \alpha_i = 0,\ t = 1, \ldots, \dim(\mathcal{H}_\psi),$$

coincides with a domain

$$\sum_{i=1}^{m} \kappa_{ij}^{\psi} \alpha_i = 0,\ j = 1, \ldots, m$$

referring only to inner products of the label vectors.

5.3 Simple solution for the unbiased case

The unbiased case of has the form

$$\begin{array}{ll} \min & \frac{1}{2}\boldsymbol{\alpha}'\mathbf{K}\boldsymbol{\alpha}' + \mathbf{q}'\boldsymbol{\alpha} \\ \text{w.r.t.} & \boldsymbol{\alpha}, \\ \text{s.t.} & \mathbf{0} \leq \boldsymbol{\alpha} \leq C, \end{array} \tag{49}$$

where $\mathbf{K} = \mathbf{K}_{\psi(y)} \bullet \mathbf{K}_{\phi(x)}$ is the point wise product of the output and input kernel, and $\mathbf{q} = \mathbf{1}$ a vector with every component equals to 1

The next simple, coordinate descent, approach seems to be over simplified but when the sample size is really large, > 10000 then the inherent simplicity becomes superior when the matrix Q is dense. We would like to emphasize another approach, e.g. interior point methods, could perform better in smaller problems, but the difference not much significant.

Step 1 Let $\boldsymbol{\alpha}^{\mathbf{0}} = \mathbf{0}$ a feasible initial solution, ϵ_α an error tolerance, and $k = 0$ a counter.

Step 2 $k = k + 1$, $\boldsymbol{\alpha}^k = \boldsymbol{\alpha}^{k-1}$, and set the component index of $\boldsymbol{\alpha}^k$, i to 0.

Step 3 Solve the unconditional problem:

$$\min_\tau (\boldsymbol{\alpha}^k + \mathbf{e}_i\tau)'\mathbf{K}(\boldsymbol{\alpha}^k + \mathbf{e}_i\tau) + \mathbf{q}'(\boldsymbol{\alpha}^k + \mathbf{e}_i\tau), \tag{50}$$

where $\mathbf{e}_i$ is a vector with 0 components except the component i which is equal to 1. Problem (50) has a closed form optimal solution τ_* which reads as

$$\tau_* = \frac{-q_i - \mathbf{e}_i'\mathbf{K}\boldsymbol{\alpha}^k}{Q_{ii}} = \frac{-q_i - \mathbf{K}_i\boldsymbol{\alpha}^k}{Q_{ii}}, \tag{51}$$

where $\mathbf{K}_i$ denotes the ith row of Q.

Step 4 Set the ith component of $\boldsymbol{\alpha}^k$ to $\alpha_i^k = \alpha_i^k + \tau$.

Step 5 If $\alpha_i^k > C$ then $\alpha_i^k = C$, and if $\alpha_i^k < 0$ then $\alpha_i^k = 0$; which operations is the projection of an infeasible solution back into(onto) the domain of the box constraint.

Step 6 $i = i + 1$, go to Step 3!

Step 7 If $\|\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1}\|_2^2 \leq \epsilon_\alpha$ then Stop, otherwise go to Step 2!

The reasonable advantage of this coordinate descent method is that: it requires only a row of the matrix $\mathbf{K}$ in an iteration step, furthermore the division by Q_{ii} is a numerically well controllable operation, since Q_{ii} has a constant value during the procedure and if the kernels are normalized it has value 1 eliminating the need of any division in the computation process.

5.4 Prediction

After solving the dual problem with the help of the optimum dual variables we can write up the optimal linear operator

$$\mathbf{W} \quad = \sum_{i=1}^{m} \alpha_i \boldsymbol{\psi}(\mathbf{y}_i)\boldsymbol{\phi}(\mathbf{x}_i)'.$$

Comparing this expression with the corresponding formula which gives the optimal solution to the SVM, i.e.

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{\phi}(\mathbf{x}_i),$$

we can see that the new part includes the vectors representing the output items which in the SVM were only scalar values but we could say in the new interpretation they are one-dimensional vectors. With the expression of the linear operator $\mathbf{W}$ at hand the prediction to a new input item $\mathbf{x}$ can be written up by

$$\begin{aligned} \boldsymbol{\psi}(\mathbf{y}) &= \mathbf{W}\boldsymbol{\phi}(\mathbf{x}) \\ &= \sum_{i=1}^{m} \alpha_i \boldsymbol{\psi}(\mathbf{y}_i) \boxed{\underbrace{\langle \boldsymbol{\phi}(\mathbf{x}_i), \boldsymbol{\phi}(\mathbf{x}) \rangle}_{\kappa^{\phi}(\mathbf{x}_i, \mathbf{x})}}. \end{aligned}$$

It involves only the input kernel κ^{ϕ} and provides the implicit representation of the prediction $\boldsymbol{\psi}(\mathbf{y})$ to the corresponding output $\mathbf{y}$.

If only the implicit image of the output is given we need to invert the function ψ to gain the $\mathbf{y}$. This inversion problem is sometimes called as pre-image problem as well. Unfortunately there is no general procedure to do that efficiently in case of complex and non-invertible mapping. We mention here a schema that can be applied when the set of all possible outputs is finite with a reasonable small cardinality. The meaning of the "reasonable small" cardinality depends on the given problem, e.g. how expensive to compute the inner product between the output items in the label space where they are represented.

At the conditions mentioned above we can follow this scenario

$$
\begin{aligned}
\mathbf{y} &\in \widetilde{\mathcal{Y}} \quad \Leftarrow \text{Set of the possible outputs,} \\
\mathbf{y}^* &= \arg\max\nolimits_{\mathbf{y}\in\widetilde{\mathcal{Y}}} \boldsymbol{\psi}(\mathbf{y})'\mathbf{W}\boldsymbol{\phi}(\mathbf{x}), \\
&= \arg\max\nolimits_{\mathbf{y}\in\widetilde{\mathcal{Y}}} \sum\nolimits_{i=1}^{m} \alpha_i \overbrace{\langle \boldsymbol{\psi}(\mathbf{y}), \boldsymbol{\psi}(\mathbf{y}_i)\rangle}^{\kappa^{\psi}(\mathbf{y},\mathbf{y}_i)} \overbrace{\langle \boldsymbol{\phi}(\mathbf{x}_i)'\boldsymbol{\phi}(\mathbf{x})\rangle}^{\kappa^{\phi}(\mathbf{x}_i,\mathbf{x})}, \\
\mathbf{y} &\in \widetilde{\mathcal{Y}} = \{\mathbf{y}_1, \ldots, \mathbf{y}_K\},\ K \ll \infty.
\end{aligned}
$$

The main advantage of this approach is that it requires only the inner products in the label space, in turn, it is independent from the representation of the output items and can be applied in any complex structural learning problem, e.g. on graphs. A suitable candidate for $\widetilde{\mathcal{Y}}$ could be the training set.

5.5 One-class SVM interpretation

Let us reformulate the inner-product occurring in the constraints whilst the bias term being dropped

$$
\begin{aligned}
\langle \boldsymbol{\psi}(\mathbf{y}_i), \mathbf{W}\boldsymbol{\phi}(\mathbf{x}_i)\rangle_{\mathcal{H}_\psi} &= \mathbf{tr}\big(\boldsymbol{\psi}(\mathbf{y}_i)'\mathbf{W}\boldsymbol{\phi}(\mathbf{x}_i)\big) \\
&= \mathbf{tr}\big(\mathbf{W}\boldsymbol{\phi}(\mathbf{x}_i)\boldsymbol{\psi}(\mathbf{y}_i)'\big) = \langle \mathbf{W}, \big[\boldsymbol{\psi}(\mathbf{y}_i) \otimes \boldsymbol{\phi}(\mathbf{x}_i)\big]\rangle_{\mathcal{H}_\psi \otimes \mathcal{H}_\phi}
\end{aligned}
$$

thus, we have a one-class SVM problem living in the tensor product space of the feature and the label spaces, where $\otimes$ denotes the tensor product.

One can extend the range of applications by using not only tensor product but more general relationship between the output and input items, i.e.,

$$
\langle \mathbf{W}, \boldsymbol{\Psi}(\mathbf{y}_i, \mathbf{x}_i)\rangle_{\mathcal{H}_W},\ \boldsymbol{\Psi} : \mathcal{H}_\psi \times \mathcal{H}_\phi \to \mathcal{H}_W.
$$

If $\mathbf{dim}(\mathcal{H}_W) > \mathbf{dim}(\mathcal{H}_\psi) + \mathbf{dim}(\mathcal{H}_\phi)$ then the support of the distribution of one-class sample items is restricted on a manifold in $\mathcal{H}_W$. Further details of the extensions beyond the tensor product can be found in [12].

6 Preliminary results on shape estimation

On Figures 2 and 3 some preliminary results are presented. The first figure shows the sample of points of a torus, an object with hole, and the predicted surface learned of those sample points. The number of smaple point is equal to 200. To these point the corresponding normal vectors are computed and used in the prediction. The points are randomly and uniformly subsampled from the parametric represention of the torus.

The second figure demonstrates a complex object which consists of parts with significantly different geometries. The sample points of the shape is

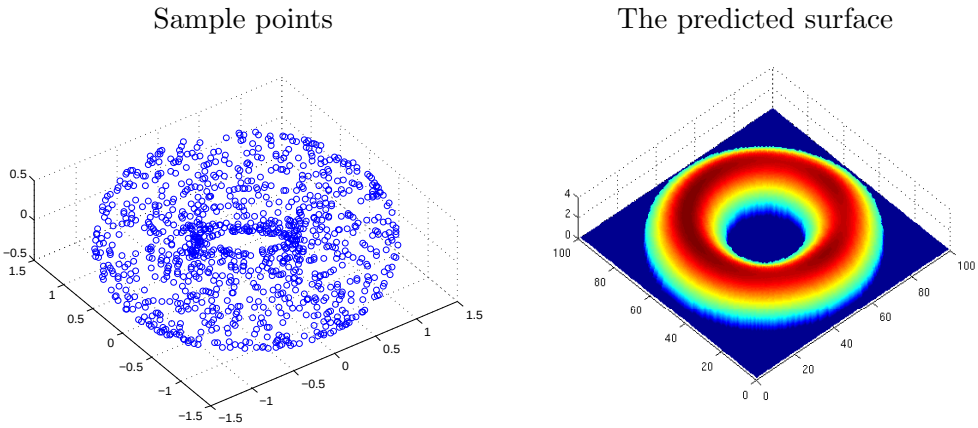


Figure 2: Learning the shape of a torus, an object with hole, of randomly, uniformly sampled surface points

provided by the Microsoft Kinect device. To those point the normal vectors are estimated by the Point Cloud Library, an open source package, see details [6]. Within Figure 3 on the first image the points and the Support Vectors are shown, the second image additionally presents the confidence region around the point cloud.

Acknowledgment

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 270273, Xperience.

A Use of operators in the derivation of the kernels

When the kernels are derived we intensively exploiting the following rules connecting vectors of different vector spaces.

From two vectors of two distinct Hilbert spaces, $u_\alpha \in \mathcal{H}_\alpha$ and $u_\beta \in \mathcal{H}_\beta$ we can create an operator

$$[u_\alpha \otimes u_\beta] : \mathcal{H}_\beta \to \mathcal{H}_\alpha, \tag{52}$$

which action on a vector v_β of $\mathcal{H}_\beta$ is defined by

$$[u_\alpha \otimes u_\beta] v_\beta \stackrel{def}{=} \langle u_\beta, v_\beta \rangle \, u_\alpha. \tag{53}$$

The conjugate of this operator is defined and denoted by

$$[u_\alpha \otimes u_\beta]^* \stackrel{def}{=} [u_\beta \otimes u_\alpha], \tag{54}$$

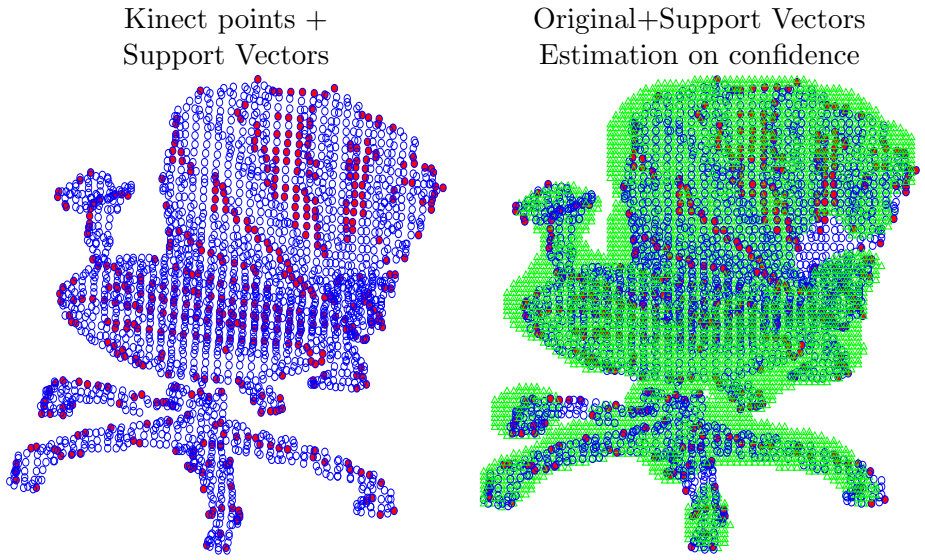


Figure 3: Learning the surface of an armchair from a Kinect provided point set. On the left the sample points(blue) and the Support Vectors(red) are presented, on the right to those points on the left the confidence region is added(green)

which maps $\mathcal{H}_\alpha$ into $\mathcal{H}_\beta$.

The product of two operators

$$[u_\alpha \otimes u_\beta][v_\beta \otimes v_\gamma] : \mathcal{H}_\gamma \to \mathcal{H}_\alpha, \tag{55}$$

where

$$\begin{array}{l} [u_\alpha \otimes u_\beta] : \mathcal{H}_\beta \to \mathcal{H}_\alpha \\ [v_\beta \otimes v_\gamma] : \mathcal{H}_\gamma \to \mathcal{H}_\beta \end{array} \tag{56}$$

is defined as

$$[u_\alpha \otimes u_\beta][v_\beta \otimes v_\gamma] w_\gamma \stackrel{def}{=} \langle u_\beta, v_\beta \rangle \langle v_\gamma, w_\gamma \rangle u_\alpha \tag{57}$$

References

[1] K. Astikainen, L. Holm, E. Pitkanen, J. Rousu, and S. Szedmak. Reaction kernels, structured output prediction approaches for novel enzyme function. In *Conference on Bioinformatics 2010, Valencia*. 2010. Best Paper Award.

[2] D.P. Bertsekas. *Nonlinear Programming*. Athena Scienctific, second edition edition, 1999.

[3] J.M. Lee. *Introduction to Smooth Manifolds*, volume 218 of *Graduate Texts in Mathematics*. Springer, 2003.

[4] M. Popović, G. Kootstra, J. A. Jørgensen, D. Kragic, and N. Krüger. Grasping unknown objects using an early cognitive vision system for general scene understanding. In *2011 IEEE*. 2011.

[5] J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor. Effcient algorithms for maxmargin structured classification. In *Predicting Structured Data*, pages 105–129. 2007.

[6] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[7] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson. Estimating the support of a high dimensional distribution. *Neural Computation*, 13(7):1443 –1472, 2001.

[8] F. Steinke, M. Hein, J. Peters, and B. Schölkopf. Manifold-valued thinplate splines with applications in computer graphics. *Computer Graphics Forum*, 27(2):437–448, 2008.

[9] F. Steinke, B. Schölkopf, and V. Blanz. Support vector machines for 3d shape processing. *Computer Graphics Forum*, 24(3), EUROGRAPHICS 2005):285–294, 2005.

[10] S. Szedmak and Z. Hussain. A universal machine learning optimization framework for arbitrary outputs. 2009. http://eprints.pascal-network.org.

[11] S. Szedmak, Y. Ni, and S. R. Gunn. Maximum margin learning with incomplete data: Learning networks instead of tables. *Journal of Machine Learning Research, Proceedings*, 11, Workshop on Applications of Pattern Analysis:96–102, 2010. jmlr.csail.mit.edu/proceedings/papers/v11/szedmak10a/szedmak10a.pdf.

[12] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 6(Sep):1453–1484, 2005.