# Support Vectors Learning for Vector Field Reconstruction

Marcos Lage, Rener Castro, Fabiano Petronetto, Alex Bordignon, Geovan Tavares, Thomas Lewiner, Hélio Lopes
*Matmídia Laboratory – Department of Mathematics, PUC–Rio – Rio de Janeiro, Brazil*
*www.matmidia.mat.puc-rio.br/{mlage,fbipetro,alexlaier,tavares,tomlew,lopes}*

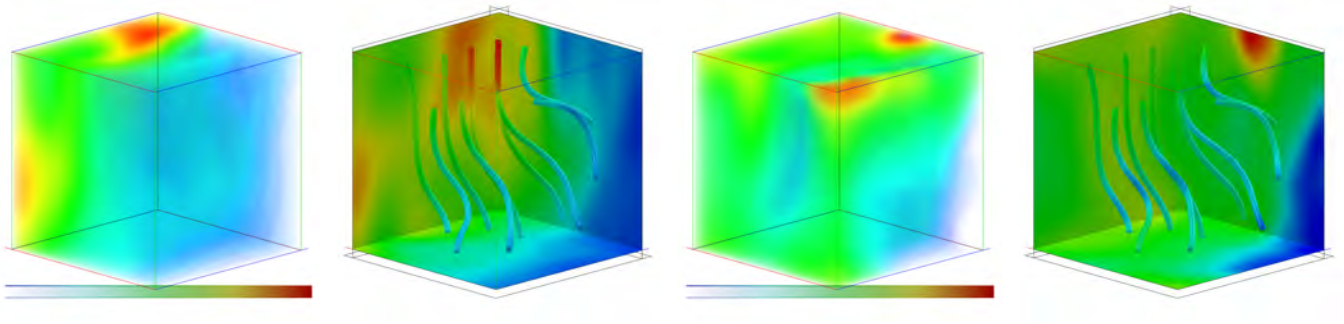


Figure 1. Reconstruction of a real 3D velocity field captured by a PIV device with sparse, irregular sampling: magnitude (left) and phase (right).

***Abstract*—Sampled vector fields generally appear as measurements of real phenomena. They can be obtained by the use of a Particle Image Velocimetry acquisition device, or as the result of a physical simulation, such as a fluid flow simulation, among many examples. This paper proposes to formulate the unstructured vector field reconstruction and approximation through Machine-Learning. The machine learns from the samples a global vector field estimation function that could be evaluated at arbitrary points from the whole domain. Using an adaptation of the Support Vector Regression method for multi-scale analysis, the proposed method provides a global, analytical expression for the reconstructed vector field through an efficient non-linear optimization. Experiments on artificial and real data show a statistically robust behavior of the proposed technique.**

***Keywords*-Discrete Vector Field; Support Vector Machine;**

## I. Introduction

In the last few years, lot of attention has been paid to the problem of object reconstruction from sparse samples [1]. However, there are still very few reconstruction methods for vector fields, which is the fundamental object in classical Physics and Engineering (velocity fields, force fields, etc.). Moreover, most of existing methods restrict the samples to be structured on a regular grid.

Sampled vector fields generally appear as measurements of real phenomena, for example using a Particle Image Velocimetry acquisition device. They also appear as the result of physical simulations, such as fluid dynamics simulations. In those context, the vector field reconstruction problem consists in inferring a *differentiable* vector field on the whole region of experimentation from only a finite, set of samples.

*Learning Vector Field:* Reconstructing a sampled field is a helpful step to analyze it identifying the existence of vortices and singularities, to improve simulations incorporating global information in local computations and to interpretation of the measured or simulated field returning numerical and visual information (see Figure 1).

Contrarily to the curve or surface reconstruction application problems, vector fields usually characterize local behaviors and thus require only consistency based on local pattern. We therefore argue that the vector field reconstruction problem should be solved in the *Machine-Learning* context. Kernel-based methods are considered the state of the art in machine-learning. Amid them, the *Support Vector Machines* (SVM) proposed by Vapnik *et al.* [2] is one of the most robust in terms of statistical learning, since they provide a deterministic and analytical result from an efficient non-linear optimization. Starting from a cost function that is insensitive to small errors, it reduces the learning process to a linearly constrained quadratic programming problem, guaranteeing a unique and globally optimal solution. Moreover, the solution is a combination of a reduced set of the input, the support vectors, which turns the SVM evaluation particularly efficient.

*Related work:* Unlike surface reconstruction from unorganized points, vector field reconstruction of unstructured data does not appear frequently in the literature. Schaback and Wendland [3] introduced the radial-based interpolants and other approximation methods for multivariate functions. Mussa-Ivaldi [4] presents a method for 2D vector field reconstruction using least squares schemes. In this strategy, the reconstruction is done in two steps. The first step

reconstructs the rotational-free part of the velocity field, and the second one reconstructs from the residual vector field the divergence-free part. Another work for 2D vector field reconstruction from sparse samples was done by Lage *et al.* [5]. In their method the vector field is reconstructed by adjusting locally a polynomial for each coordinate and then the global approximation is obtained by the use of a partition of unity. In fact, their work is a generalization of the Multi-level Partition of Unity for surface reconstruction [6].

Two important reconstruction methods for surface reconstruction based on Support Vector Machines have been proposed in the last years. Schölkopf *et al.* [7] introduces a surface reconstruction scheme using the so–called single class Support Vector Machine method. Steinke *et al.* [8] present a multi-scale method for surface reconstruction based on a Support Vector Regression. The Support Vector Regression has also been recently used for optical flow reconstruction [9] in the work of [10]. This paper proposes methods that extend these learning techniques for scalar fields to vector fields reconstruction.

*Contributions:* This paper presents a method for vector field reconstruction from sparse points/vectors pairs, introducing a learning formulation to such problems. It uses the Support Vector Machine for function estimation technique, called the *Support Vector Regression* (SVR), as the basic tool for the learning part (see Section II). The learning machine is trained on the samples and evaluated over the whole domain, introducing an adaptation of SVR for multi-scale analysis (see Section III). This method provides a single analytical expression for the reconstructed vector field on the whole domain. Experiments on artificial and real data in two and three dimensions show a statistically robust behavior of the proposed technique (see Section IV).

## II. Support Vector Regression

The *Support Vector Regression* (SVR) is a universal learning machine for solving multidimensional scalar value prediction and estimation problems. It has received a lot of attention in the machine-learning community since it is very well grounded on a statistical learning theory, called the *VC-Theory* [11]. Its consistency conditions, its convergence, its generalization abilities and its implementation efficiency have been studied by several authors from the last four decades (see [11] and [12]). This section describes the $\varepsilon$-SVR. A complete introduction can be found in [13].

### A. $\varepsilon$-SVR learning problem

Consider the training set $\mathcal{S} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)\}$, where $\mathbf{x}_i \in \mathbb{R}^n$ are the explicative data and $y_i \in \mathbb{R}$ are the target values. The SVR method first maps the data $\mathbf{x} \in \mathbb{R}^n$ into some chosen Hilbert space $\mathcal{F}$, called the *feature space*, via a nonlinear function $\phi : \mathbb{R}^n \rightarrow \mathcal{F}$. In this feature space, the prediction function $f$ is formulated by the affine equation:

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b \tag{1}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product in $\mathcal{F}$, with $\mathbf{x} \in \mathbb{R}^n$, and $\mathbf{w} \in \mathcal{F}$ and $b \in \mathbb{R}$ are the variables to be determined.

### B. $\varepsilon$-SVR optimization problem

In the $\varepsilon$-SVR learning method, the values of $\mathbf{w}$ and $b$ are determined by the following optimization problem [11]:

$$\begin{array}{ll} \text{Minimize}_{\mathbf{w},b} & \frac{1}{2}\|\mathbf{w}\|^2 + P \cdot \sum_{i=1}^{l}(\xi_i + \hat{\xi}_i) \\ \text{subject to:} & \left\{ \begin{array}{rcl} y_i - (\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle - b) & \leq & \varepsilon + \xi_i \\ (\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle + b) - y_i & \leq & \varepsilon + \hat{\xi}_i \\ \xi_i, \hat{\xi}_i & \geq & 0 \end{array} \right. \end{array}$$

where $P > 0$ determines the trade-off between the flatness of $f$ (small $||\mathbf{w}||$) and the amount up to which deviations of the estimation is larger than $\varepsilon$ are tolerated. The variables $\xi_i$ and $\hat{\xi}_i$ represent the deviation at sample $i$ when $f(\mathbf{x}_i)$ is above or below $y_i$, respectively.

One can rewrite this optimization problem in its dual form, using Lagrange multipliers $\alpha_i, \hat{\alpha}_i$:

$$\begin{array}{ll} \text{Maximize}_{\alpha_i, \hat{\alpha}_i} & \sum_{i=1}^{l}(\hat{\alpha}_i - \alpha_i)y_i - \varepsilon \sum_{i=1}^{l}(\hat{\alpha}_i + \alpha_i) - \\ & \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(\hat{\alpha}_i - \alpha_i)(\hat{\alpha}_j - \alpha_j)\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \\ \text{subject to:} & \sum_{i=1}^{l}(\hat{\alpha}_i - \alpha_i) = 0, \quad 0 \leq \alpha_i, \hat{\alpha}_i \leq P \\ & \mathbf{w} - \sum_{i=1}^{l}(\hat{\alpha}_i^* - \alpha_i^*)\phi(\mathbf{x}_i) = 0 \\ & \alpha_i^*(\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle + b - y_i - \varepsilon - \xi_i) = 0 \\ & \hat{\alpha}_i^*(y_i - \langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle - b - \varepsilon - \hat{\xi}_i) = 0 \\ & \hat{\alpha}_i^* \cdot \alpha_i^* = 0, \quad \hat{\xi}_i \cdot \xi_i = 0 \\ & (\hat{\alpha}_i^* - P)\hat{\xi}_i = 0, \ (\alpha_i^* - P)\xi_i = 0 \end{array}$$

This dual problem is a convex quadratic programming problem, thus it has an unique global solution. Such optimal solution will be denoted by $\mathbf{w}^\star, b^\star, \hat{\alpha}^\star, \alpha^\star$.

*Support Vectors:* The second restriction of this problem means that at the optimal solution $\mathbf{w}^\star$ for the primal problem is a linear combination of the explicative points mapped to the feature space: $\mathbf{w}^\star = \sum_{i=1}^{l}(\hat{\alpha}_i^\star - \alpha_i^\star)\phi(\mathbf{x}_i)$, equation (1) can be rewritten as:

$$f(\mathbf{x}) = \sum_{i=1}^{l}(\hat{\alpha}_i^\star - \alpha_i^\star)\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}) \rangle + b^\star. \tag{2}$$

where $b^*$ is chosen so that $f(\mathbf{x}_i) - y_i = -\varepsilon$ for any $i$ such that $\alpha_i^\star \in (0, P)$.

The other set of restrictions says that when $\alpha_i^\star$ and $\hat{\alpha}_i^\star$ are both equal to zero the scalar function prediction for the explicative point $\mathbf{x}_i$ distances from the target value $y_i$ less than $\varepsilon$. The explicative points $\mathbf{x}_i$ whose one of the associated $\alpha_i^\star$ or $\hat{\alpha}_i^\star$ does not vanish are called the *support vectors*.

### C. Kernel functions

Kernel functions have been recognized as an important tool in several numerical analysis applications, including approximation, interpolation, meshless method for solving differential equations and also in Machine Learning [3].

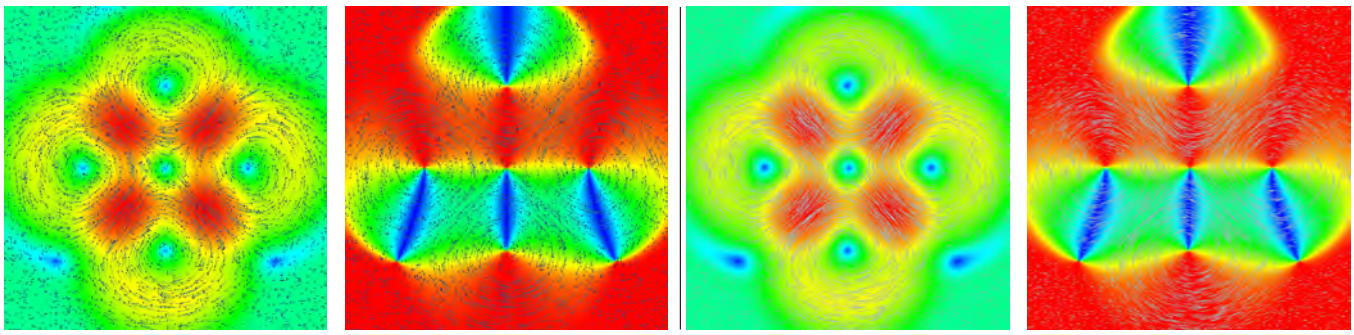


Figure 2. 3600 vectors of an synthetic vector field sampled randomly in the unit square (magnitude and cosine of the phase on the left), reconstructed by learning in polar coordinates (magnitude and cosine of the phase on the right). The color scale from blue to red.

In the $\varepsilon$-SVR problem, the non-linear function $\phi$, which maps the explicative point to the feature space, appears in two equations: one in the objective function of the $\varepsilon$-SVR dual optimization problem as $\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$, and the other in the prediction function $f$ as $\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}) \rangle$ (Equation (2)). Notice that in both cases it is sufficient to know how to compute the inner-product $\langle \phi(\mathbf{y}), \phi(\mathbf{z}) \rangle$ of two points mapped to feature space by $\phi$.

This operation is directly modeled by the use of *Kernel functions*. A *kernel function* $K : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ is defined by:

$$K(\mathbf{y}, \mathbf{z}) = \langle \phi(\mathbf{y}), \phi(\mathbf{z}) \rangle.$$

In fact, kernel functions represent implicitly the mapping $\phi$ to the feature space $\mathcal{F}$. For example, consider that $\mathbf{y}$ and $\mathbf{z}$ are in $\mathbb{R}^2$. Also consider the non-linear mapping $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ as $\phi(\mathbf{y}) = (y_1^2, y_2^2, \sqrt{2} y_1 y_2)$. Then,

$$\langle \phi(\mathbf{y}), \phi(\mathbf{z}) \rangle = y_1^2 z_1^2 + y_2^2 z_2^2 + 2 y_1 y_2 z_1 z_2 = (\langle \mathbf{y}, \mathbf{z} \rangle)^2.$$

In this case computing the inner-product using $\phi$, requires the evaluation of the non–linear mapping at each 2D point and, after that, the computation of their inner-product in $\mathbb{R}^3$. A more efficient way to evaluate it uses the kernel function $K(\mathbf{y}, \mathbf{z}) = (\langle \mathbf{y}, \mathbf{z} \rangle)^2$, which computes firstly the inner-product in $\mathbb{R}^2$ and then takes the square of it.

In the general case, it is more efficient and more suitable to choose kernels rather than non-linear mappings $\phi$. However, not all functions $K$ represent an inner-product in the feature space. The Mercer's theorem characterizes these functions [11]. Some examples of kernel functions that satisfy the Mercer's conditions are:

- Polynomial kernel: [2]: $K(\mathbf{y}, \mathbf{z}) = (1 + \langle \mathbf{y}, \mathbf{z} \rangle)^d$,
- Gaussian kernel: [2]: $K(\mathbf{y}, \mathbf{z}) = exp\left(-\frac{\|\mathbf{y}-\mathbf{z}\|^2}{2\sigma^2}\right)$,
- Wavelet kernel [14]: $K(\mathbf{y}, \mathbf{z}) = \prod_{i=1}^{n} h\left(\frac{y_i - z_i}{\sigma}\right)$, where $h(u) = \cos(1.75u) e^{-\frac{u^2}{2}}$,

## III. Reconstruction by Learning Method

### A. Sampled vector fields

A *vector field* $\mathbf{F}$ defined on a subset $\Omega \subset \mathbb{R}^n$ is a map that assigns to each point $\mathbf{x} \in \Omega$ a vector $\mathbf{v} \in \mathbb{R}^n$ (Figure 2 (left)). In the Cartesian coordinate system, the vector field is represented by an ordered $n$-tuple of scalar functions $\mathbf{F}(\mathbf{x}) = (F_1(\mathbf{x}), F_2(\mathbf{x}), \ldots, F_n(\mathbf{x}))$ The function $F_i$ is called the $i$-th coordinate function of $\mathbf{F}$. A vector field is differentiable when all of its $n$ coordinate functions are.

This paper aims at providing a differentiable vector field $\hat{\mathbf{F}} : \Omega \rightarrow \mathbb{R}^n$ that approximates an ideal vector field $\mathbf{F}$ on the region $\Omega$ by the use of a learning-machine method based on $\varepsilon$-SVR. As an input of the reconstruction problem, it is considered a set of $l$ pairs of $n$ dimensional points $\mathcal{S} = \{(\mathbf{x}_1, \mathbf{v}_1), \ldots, (\mathbf{x}_l, \mathbf{v}_l)\}$ sampled from $\mathbf{F}$, such that $\mathbf{x}_i \in \Omega$ and $\mathbf{v}_i \approx \mathbf{F}(\mathbf{x}_i) \in \mathbb{R}^n$. It is assumed that $\mathbf{x}_i$'s are independent and identically distributed samples, and that both $\mathbf{x}_i$ and $\mathbf{v}_i$ are on the same basis of the Cartesian coordinates.

### B. Learning 2D vector fields

There are two classical ways to represent a vector $\mathbf{v} \in \mathbb{R}^2$. One is the Cartesian coordinates system $\mathbf{v} = (v_1, v_2)$ and the other is the polar coordinates system $(r, \theta)$, for $r \in [0, \infty)$ and $\theta \in [0, 2\pi)$ (Figure 2). The equality $(v_1, v_2) = r(\cos\theta, \sin\theta)$ is used to convert one system into the other. Thus, we propose two methods for learning 2D vector field $\mathbf{F} : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}^2$, one for each coordinate system.

*Learning in Cartesian coordinates:* The first 2D vector field learning method determines, from the samples in $\mathcal{S}$, the reconstructed field $\hat{\mathbf{F}}$ by learning each coordinate function of $\mathbf{F}$ separately. This means that two $\varepsilon$-SVR machine learning problems are solved, one to find $\hat{F}_1$ that approximates $F_1$ and other to find $\hat{F}_2$ that approximates $F_2$. The approximation for the vector field function $\mathbf{F}$ is thus obtained by:

$$\hat{\mathbf{F}}_{2Dc}(\mathbf{x}) = (\hat{F}_1(\mathbf{x}), \hat{F}_2(\mathbf{x})),$$

where $\hat{F}_j(\mathbf{x}) = \sum_{i=1}^{l} (\hat{\alpha}_{i,j}^{\star} - \alpha_{i,j}^{\star}) K(\mathbf{x}_i, \mathbf{x}) + b_j^{\star}$.

*Learning in polar coordinates:* The second 2D method determines from the sampling set $\mathcal{S}$ the reconstructed field $\hat{\mathbf{F}}$ by learning three functions:

$$R(\mathbf{x}) = \|\mathbf{F}(\mathbf{x})\|, C(\mathbf{x}) = \frac{F_1(\mathbf{x})}{\|\mathbf{F}(\mathbf{x})\|}, \text{ and } S(\mathbf{x}) = \frac{F_2(\mathbf{x})}{\|\mathbf{F}(\mathbf{x})\|}.$$

The first represents the norm of the vector $\mathbf{F}(\mathbf{x})$, the second represents the cosine of the phase and the last the sine of the phase. The approximation for these three functions, respectively named $\hat{R}, \hat{C}$, and $\hat{S}$, are also expressed using equation (2). Figure 2 shows the result of this approach on a synthetic field.

It is better to learn both the cosine of $\theta$ and the sine of $\theta$ instead of only the argument angle $\theta$ itself since it avoids the discontinuity of the argument $\theta$ close to 0 or $2\pi$. Such discontinuities do not fit well for SVR, since the prediction function $f$ is continuous for continuous kernels (see equation (1)). To avoid this problem, the following adjustment is proposed: the predicted point $(\hat{C}(\mathbf{x}), \hat{S}(\mathbf{x}))$ is projected on the unit circle orthogonally. Notice that the projected point has the same argument as the original, correcting only the phase as desired. According to this, the adjusted point is:

$$(\underline{\hat{C}}(\mathbf{x}), \underline{\hat{S}}(\mathbf{x})) = \frac{(\hat{C}(\mathbf{x}), \hat{S}(\mathbf{x}))}{\sqrt{\hat{C}^2(\mathbf{x}) + \hat{S}^2(\mathbf{x})}},$$

$$\hat{\mathbf{F}}_{2Dp}(\mathbf{x}) = (\hat{R}(\mathbf{x})\underline{\hat{C}}(\mathbf{x}), \hat{R}(\mathbf{x})\underline{\hat{S}}(\mathbf{x})).$$

Figure 6(right) shows that the approximation $\hat{\mathbf{F}}_{2Dp}$ for the vector field function $\mathbf{F}$ using this second strategy improves the pointwise relative error.

### C. The role of the support-vectors.

According to the definition given in section II, in the $\varepsilon$-SVR learning method, the explicative points $\mathbf{x}_i$ for which the associated $\alpha_i^\star$ or $\hat{\alpha}_i^\star$ does not vanish are called the *support vectors*. Consequently, the explicative points that are not support vectors have estimation errors less than $\varepsilon$. Since the support vectors are the only explicative points used to compute the $\varepsilon$-SVR estimated function (see equation 2), it is important to notice that these points have a strong geometric meaning. Since the reconstructed functions are computed only from the support vectors, they capture the main elements of the reconstructed vector field. It is important to observe that the number of support vectors heavily depends on the parameters of the SVR, in particular on $\varepsilon$ and $P$.

For example, in Figure 3 the support vectors for the sampled vector field of Figure 2 using the Cartesian coordinate method (left) and the polar coordinate method (right). In this example, the Cartesian coordinates system reconstruction uses 160 support vectors for the x-coordinate and the same number for the y-coordinate, while in the polar coordinates system reconstruction there are 155 support vectors for the norm predicted function, and respectively 182 and 142 support vectors for the cosine and for the sine predicted functions (Figure 2).

The left image Figure 3 shows that, the support vectors of each direction in the Cartesian method identified features of the field in the corresponding directions. However, the right image shows that the support vector in the polar method captured much more features. This occurs because the polar method have to learn the norm, the sine and the cosine of the phase, which characterize more clearly the singularities of the vector field. As a conclusion, this advantage of the polar coordinate method compensates the fact that it has to solve three learning problems to obtain the reconstruction.

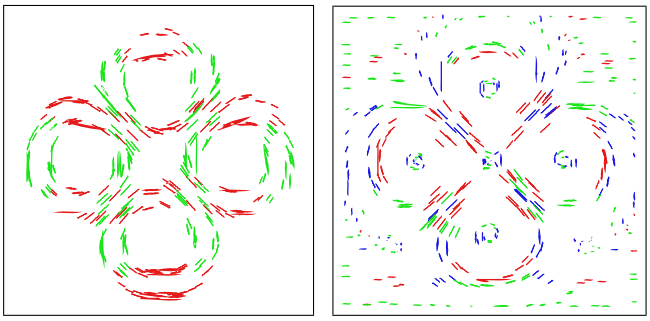


Figure 3. The support vectors in the Cartesian (left) and polar coordinates (right), with the following color code: (left) red for $x$, green for $y$, (right) red for the norm, green and blue for the phase cosine and sine.

### D. Learning 3D vector fields

The learning methods proposed for 2D vector field reconstruction are easily generalized to 3D as follows.

*Learning in Cartesian coordinates:* The reconstruction method for a 3D vector field $\mathbf{F} : \Omega \in \mathbf{R}^3 \rightarrow \mathbf{R}^3$ using Cartesian coordinates learns from the sampling set $\mathcal{S}$ each coordinate function individually. obtaining:

$$\hat{\mathbf{F}}_{3Dc}(\mathbf{x}) = (\hat{F}_1(\mathbf{x}), \hat{F}_2(\mathbf{x}), \hat{F}_3(\mathbf{x})),$$

where $\hat{F}_j(\mathbf{x}) = \sum_{i=1}^{l}(\hat{\alpha}_{i,j}^\star - \alpha_{i,j}^\star)K(\mathbf{x}_i, \mathbf{x}) + b_j^\star$.

*Learning in spherical coordinates:* In spherical coordinates, a vector $\mathbf{v} = (v_1, v_2, v_3) \in \mathbb{R}^3$ is represented by the triple $(r, \theta, \gamma)$, for $r \in [0, \infty)$, $\theta \in [0, 2\pi)$ and $\gamma \in [0, \pi)$. The equality $(v_1, v_2, v_3) = r(\cos\theta \sin\gamma, \sin\theta \sin\gamma, \cos\gamma)$ is used to convert from the Cartesian to the spherical coordinate system and vice-versa.

Similarly to the polar coordinates in 2D, the approximation method for a 3D vector field function $\mathbf{F} : \Omega \in \mathbf{R}^3 \rightarrow \mathbf{R}^3$ learns from the sampling set $\mathcal{S}$ the functions:

$$R = \|\mathbf{F}\|, \quad C = \frac{F_3}{\|\mathbf{F}\|}, \quad CS = \frac{F_1}{\|\mathbf{F}\|}, \quad \text{and} \quad SS = \frac{F_2}{\|\mathbf{F}\|}.$$

$R(\mathbf{x})$ represents the norm of the vector $\mathbf{F}(\mathbf{x})$, $C(\mathbf{x})$ represents the cosine of $\gamma$, $CS(\mathbf{x})$ represents the cosine of $\theta$ times the sine of $\gamma$, and, finally, $SS(\mathbf{x})$ represents the sine of $\theta$ times the sine of $\gamma$. The approximation for these four functions, respectively named $\hat{R}, \hat{C}, \hat{CS}$ and $\hat{SS}$, are also expressed using equation (2).

In order to adjust the prediction of $C$, $CS$, and $SS$ to satisfy the identity $CS^2(\mathbf{x}) + SS^2(\mathbf{x}) + C^2(\mathbf{x}) = 1$ the following adjustment is used:

$$(\underline{\hat{CS}}(\mathbf{x}), \underline{\hat{SS}}(\mathbf{x}), \underline{\hat{C}}(\mathbf{x})) = \frac{(\hat{CS}(\mathbf{x}), \hat{SS}(\mathbf{x}), \hat{C}(\mathbf{x}))}{\sqrt{\hat{C}^2(\mathbf{x}) + \hat{CS}^2(\mathbf{x}) + \hat{SS}^2(\mathbf{x})}}.$$

As a result, the approximation for the 3D vector field function $\mathbf{F}$ using spherical coordinates is obtained by:

$$\hat{\mathbf{F}}_{3Ds}(\mathbf{x}) = (\hat{R}(\mathbf{x})\underline{\hat{C}\hat{S}}(\mathbf{x}), \hat{R}(\mathbf{x})\underline{\hat{S}\hat{S}}(\mathbf{x}), \hat{R}(\mathbf{x})\underline{\hat{C}}(\mathbf{x})).$$

Figure 1 shows the reconstruction of a real 3D velocity field captured by PIV acquisition device (piv.vsj.or.jp/piv/image3d/image351.html). The volumetric and projected visualization of the reconstruction map are displayed. The color scale from blue to red means the magnitude of the reconstruction field in the two first images, and the cosine of phase in the two last images. Some stream lines in the reconstructed velocity field are shown at the second and the fourth images.

### *E. Parameters for the $\varepsilon$-SVR regression*

For all methods discussed above the user has to choose the following parameters:

- the tolerance value $\varepsilon$,
- the penalizing constant $P$,
- the Kernel function $K$ and its corresponding parameters. For example, the Gaussian kernel is parameterized by $\sigma$.

Large $\varepsilon$ generally results in a very smooth approximation and a small number of support vectors. Small $\varepsilon$ results on better approximations since it will use almost all points (Figure 4). Tiny $\varepsilon$ (from $1.0$ to $0.25$ in the figure) may induce overfitting the data, which is not suitable in noisy cases. Small $P$ generates smooth reconstructions that loosely approximate the training data. For large $P$, the approximation fits the vector field very close to the training set, but this may harm the prediction elsewhere.

Usually, Gaussian kernel is a very nice choice for vector fields when no particular structure is known a priori. As suggested by Steinke *et al.* [8] a good initial choice of its parameter $\sigma$ is the half of the diameter of the bounding box of the points. In order to have a control of these parameters, the coordinates of the vectors $\mathbf{x}_i$ and the coordinates of the target values $y_i$ are all standardized, i.e. their value is subtracted by the mean and the result is divided by the standard deviation. After the prediction the values are transformed back using the inverse process. Using this strategy, an initial suggestion is to set $P = 1$ and $\varepsilon = 0.01$.

Another option to deduce good parameters is to separate the samples into a training set, used to compute the vector field $\hat{\mathbf{F}}$ and a validation set used to tune the SVR. This cross-validation can be easily implemented by pre-selecting values for each SVR parameter and perform a regression for each combination of these values. The parameters that generate the best prediction on the validation set are retained.

### *F. Multi-scale $\varepsilon$-SVR regression*

For surface reconstruction Steinke *et al.* [8] proposed the combination of kernels with different sizes, for example to interpolate across holes. Their scheme provides an approximation with enough variability to capture the details while guaranteeing good results in large distances. It uses a coarse-to-fine approximation: In the first scale, it captures basically the sign of the scalar function and on the subsequent scale levels it approximates the residual errors. Since they are using a kernel that is a radial basis function, at each level the scale is divided by two. The points having the desired error tolerance are not considered on the next level learning procedure. The final function is given by the sum of the functions obtained at each level. A novel adaptation of this multi-scale method is proposed here to improve the approximation results of the vector field reconstruction.

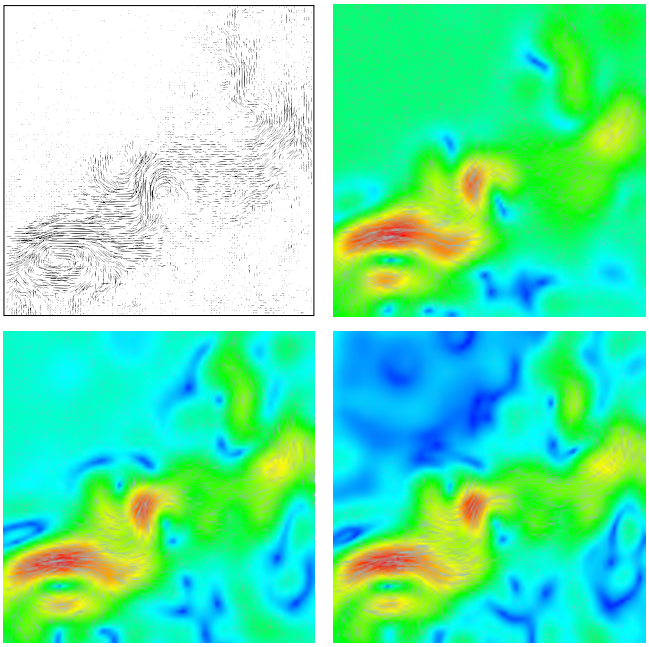


Figure 4. The effect of the $\varepsilon$ parameter: original data (top left) and decreasing $\varepsilon$ (in reading order) the reconstruction fits closer to the input data, but harms the smoothness of the reconstructed field.

Given a data set $\mathcal{S} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_l, y_l)\}$, the initial multi-scale targets are $err_i^0 = \mathbf{y}_i$ for $i = 1, \ldots, l$, thus the initial multi-scale data set is $\mathcal{S}^0 = \mathcal{S}$ and the initial estimation function is $f^0 \rightarrow \text{SVR}\{\mathcal{S}^0, \varepsilon, \sigma\}$. At scale $k$, $\mathcal{S}^k$, $\varepsilon^k$, $\sigma^k$ and $P^k$ represent, respectively, the Gaussian kernel parameter, the loss function parameter, the error penalizing constant and the training data set. The procedure $\text{SVR}\{\mathcal{S}^k, \varepsilon^k, \sigma^k, P^k\}$ returns the estimated function $f_k$ using the $\varepsilon$-SVR technique.

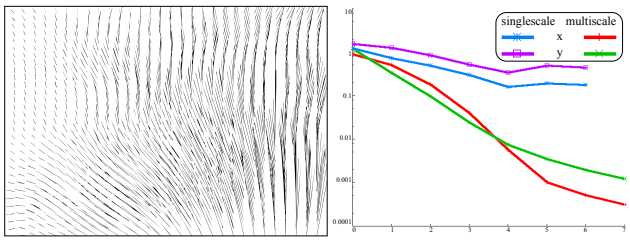
The proposed multi-scale method can be summarized by the following procedure:

$$\begin{aligned} & err_i^k = err_i^{k-1} - f^{k-1}(err_i^{k-1}) \\ & \mathcal{S}^k = \{(\mathbf{x}_1, err_1^k), (\mathbf{x}_2, err_2^k), \ldots, (\mathbf{x}_l, err_l^k)\} \\ & \varepsilon^k = \tfrac{\varepsilon}{2^k}\ , \ \sigma^k = \tfrac{\sigma}{2^k} \text{ and } P^k = P \\ & f^k \rightarrow \text{SVR}\{\mathcal{S}^k, \tfrac{\varepsilon}{2^k}, \tfrac{\sigma}{2^k}, P^k\} \end{aligned} \tag{3}$$

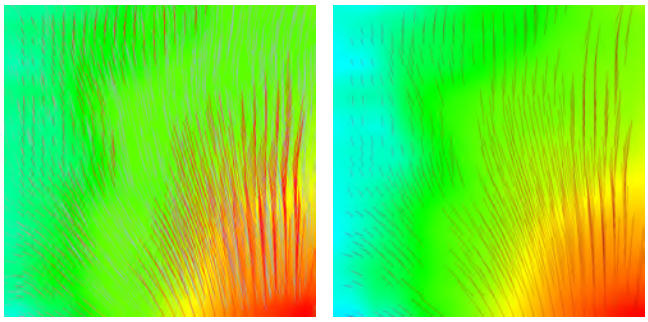
Taking into account the estimations at $N$ different scales, the final multi-scale estimation function is:

$$f^*(\mathbf{x}) = \sum_{k=1}^{N} f^k(\mathbf{x}) = \sum_{k=1}^{N} \sum_{\mathbf{v} \in SV_k} (\hat{\alpha}_{\mathbf{v}}^{k,*} - \alpha_{\mathbf{v}}^{k,*}) K(\mathbf{v}, \mathbf{x}) \quad (4)$$

Figure 5 shows an example of the improvement provided by the multi-scale scheme on a PIV data.




(a) Original filed (left) and mean absolute error (right) for $x$ and $y$ with 1 and 4 scales vs number of support vectors (logarithmic scale axies).



(b) Single scale. (c) 4 scales.

Figure 5. Single-scale regression versus multi-scale regression: the ability to reconstruct several scales at the same time improves the reconstruction on the original field. The multi-scale parameters are: $\sigma^0 = 4$, $P^0 = 1$, $\varepsilon^0 = 0.1$.

The scheme proposed above differs from Steinke *et al.* [8] in the following points: First, their method adapts SVR for surface reconstruction. Second, their method discards the points $\mathbf{x}_i$ having the desired error tolerance, in the method proposed in this work the residual vector is set to zero if the distance from $\mathbf{y}_i$ to $\hat{\mathbf{F}}(\mathbf{x}_i)$ is less than $\varepsilon$. Third, the proposed method divides by two at each level not only the $\sigma$, but also the $\varepsilon$ parameter. By doing so, this proposed scheme corresponds to the physical paradigm that higher frequencies generally have lower amplitudes. We apply this multi-scale approach to each learning method for 2D and 3D vector field reconstruction.

### G. Implementation

All the proposed vector field reconstruction methods use a Sequential Minimal Optimization (SMO) [15] to the $\varepsilon$-SVR quadratic optimization problem, the implementation is based on the open source libSVM [16], [17]. All the SVR inputs, including the errors in the multi-scale regression, are normalized to a normal distribution before being processed.

## IV. Results

The reconstruction methods proposed above were tested on different kind of vector fields: synthetic fields, velocity fields acquired from PIV devices, and velocity fields of fluid flow simulation.

We measure the quality of the reconstruction by the distribution of a punctual error at a point $\mathbf{x}$ with a known vector field $\mathbf{v} = F(\mathbf{x})$. We compute the punctual magnitude error by:

$$\frac{\|\mathbf{v} - \hat{\mathbf{F}}(\mathbf{x})\|}{max\{\|\mathbf{v}\|, \|\hat{\mathbf{F}}(\mathbf{x})\|\}}.$$

The phase error is measured by the cosine of the angle between the estimated vector and the correct vector from the analytical function. To maintain the coherence of the quality measure, no point $\mathbf{x}$ used for the error computation is used in the learning process.

### A. Synthetic Analytic Fields

Figure 2 illustrate the reconstruction of a 2D synthetic field from unstructured samples, using the Cartesian and the polar coordinates learning method respectively. Since it is ansynthetic field, a global measure of the error can be done using the average of the pointwise error at the vertices of a regular grid. Figure 6(left) shows this average error for various samplings of the same vector field. The quality of the reconstruction is not sensitive to a particular sampling. The improvement in normalization of the polar coordinates is illustrated in Figure 6(right).

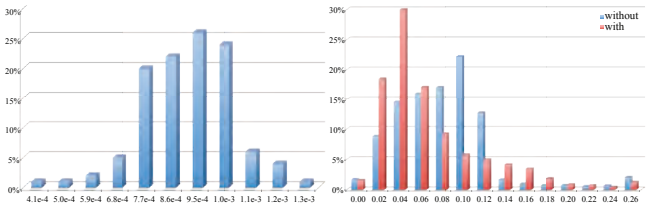



Figure 6. Error histograms in the synthetic field (Figure 2): (left) Average error distribution over 112 random uniform samplings of 3600 points: the estimation of $\hat{F}_1$ have similar behaviors independently of the sampling, provided it has the same average density. (right) The normalization of the polar coordinates improves the histogram of the pointwise relative error.

Figure 7 shows the reconstruction of the 3D synthetic field $F(x, y, z) = (-z, 0.2y^2, x)$ with 5000 random samples in the $[-2.8, 2.8]^3$ domain. In reading order, the first image shows some stream lines of the reconstruction field with colors representing, from blue to red, the cosine of the field's phase. The second one displays the support vectors of the function $\hat{CS}$ using the spherical coordinates learning scheme. Observe that the support vectors here show the spiral behavior of the field. The next two images of this figure illustrate respectively the pointwise errors of the field's magnitude and the cosine of phase absolute error measured on a 3D regular grid. The color scale from blue to red corresponds to, respectively, small and big errors. In the

magnitude error, the error range is $\theta(\text{error}) = [10^{-4}, 10^{0}]$, while the relative error order of the cosine phase error is $\theta(\text{error}) = [10^{-8}, 10^{-1}]$. Observe that the phase error is concentrated where the magnitude vanishes.

### B. Fields acquired from PIV techniques

Figure 8 shows our experiments on the velocity fields of a gas flow acquired from a PIV device. The original data is given on a regular grid and we selected randomly $80\%$ of the original data to use in the learning stage, and we use the remaining $20\%$ for the test/validation phase. Using the

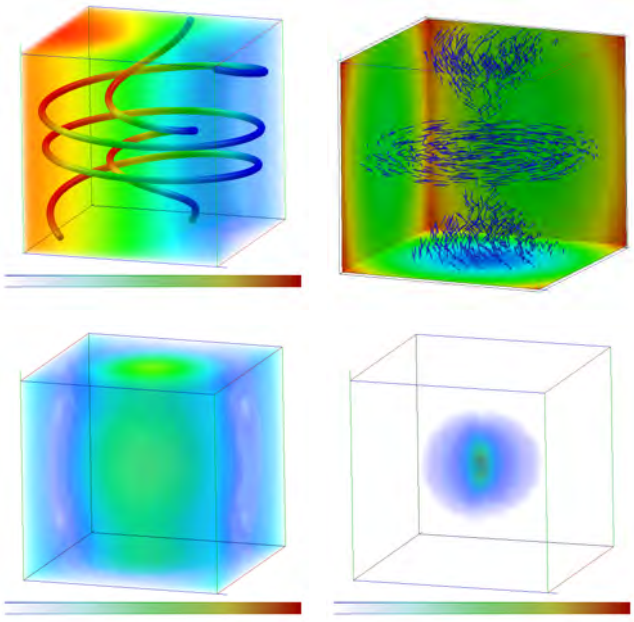


Figure 7. Reconstruction of a 3D vector field $F(x, y, z) = (-z, 0.2y^2, x)$ with 5000 random samples in the $[-2.8, 2.8]^3$ domain using a single scale $\varepsilon = 0.1$, $P = 1$ and $\sigma = 2.63522$.

Cartesian coordinate method with Gaussian kernel in single scale, we obtain the reconstructed field whose phase cosine is represented in Figure 8(left). The error (Figure 8(right)) is measured by the reconstruction error on the training set ($20\%$ of the original data). In the above context, the multi-scale approach improves on single-scale reconstruction as can be seen in the example of Figure 5.

### C. Fluid Flow Simulation Examples

Figure 9 provides an example of a velocity field reconstruction obtained from 4096 samples of an Eulerian grid-based fluid-simulation [18]. From left to right, the first image shows a grid sampled velocity field of a smoke flow, the second illustrates this field reconstructed by the 2D learning method in polar coordinates. Finally, the last image displays the support vectors results for the three $\varepsilon$-SVR problems. The vectors in red, green and blue represent, respectively, the support vector from the norm, from the cosine and from the

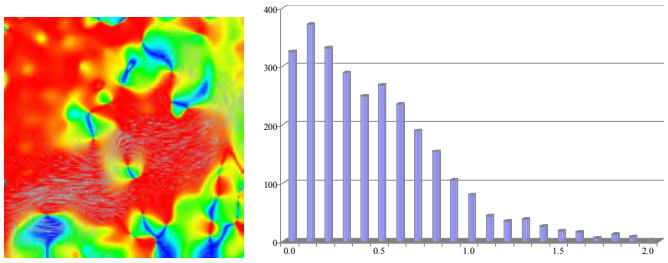



Figure 8. Exponential decreasing of the relative error computing using the test data set, which corresponds to $20\%$ separated from the original PIV data that contains 16000 samples. In the learning process, only $80\%$ of the original data was used to train the SVR machine using the Cartesian coordinates method.

sine of the polar coordinates learning scheme. Notice that, in this image, the support vectors visually identify relevant features of the vector field.

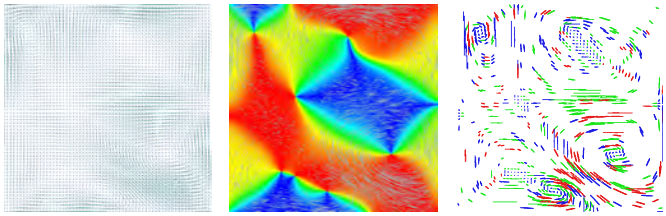


Figure 9. Reconstruction of a smoke-flow using the polar coordinates learning method: (left) The 4096 sampling points. (middle) The reconstructed cosine of phase, the colors range from blue (cosine equal to -1) to red (cosine equal to 1). (right) The vectors in red, green and blue represents respectively the support vectors from the norm, cosine and sine SVR learning process.

Figure 10 shows the reconstruction of velocity fields obtained from a 3D SPH simulation. The input data correspond to 3840 fluid particles in the free-surface flow simulation of the dam-break problem after the impact of the fluid front against the vertical wall at the end of the dry deck. Figure 10(bottom) show several examples of integral curves computed using an Euler method on the evaluation of the reconstruction function. The color scale from blue to red means the magnitude of the reconstructed field, moreover Figure 10(top) show the volumetric and projected visualization of the same map.

*Limitations:* The main limitation of our current implementation of the proposed method is still the execution time on huge data. For example, in a field with $10,000$ samples, the learning process last around $4$ minutes and the evaluation on a $100 \times 100$ grid last around $150$ seconds, which is slower than MPU approaches [5]. Since the reconstruction is global, it requires a global optimization which harms its efficiency. This will be improved in a future work by factoring results for repetitive regression, or by using local solutions for good initial guess of the SMO quadratic solver.

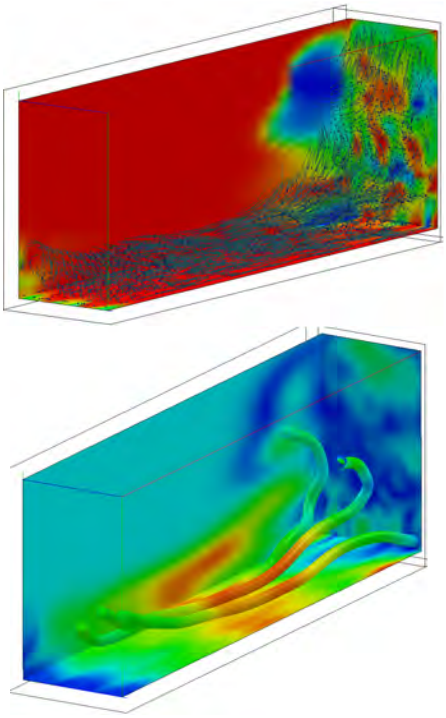


Figure 10. 3D Velocity field reconstruction using the 3D Cartesian coordinate scheme from a 3840 fluid particles in the free-surface flow simulation of the dam-break problem after the impact of the fluid front against the vertical wall at the end of the dry deck.

## V. Conclusions

This paper proposes to solve the vector field reconstruction problem in a Machine Learning context. Using the well known Support Vector Regression scheme, the proposed 2D and 3D schemes achieve faithful reconstruction on synthetic and real data. Moreover, the reconstruction is statistically stable with respect to a specific sampling. A multi-scale variation of the method improves its robustness on real data.

Other contribution of this paper is the use of support vectors as a useful tool for a visual analysis of the vector field before an eventual relatively long-lasting evaluation of topological feature detection algorithms. Since the support vectors generally appear close to the field features.

With the proposed approach, the reconstructed field is global and differentiable. This is suitable for vector field analysis involving derivatives, which can be directly calculated from the derivatives of the kernel. The authors plan to develop a new method for vector field differentiable topological analysis from samples based on the formulation presented in this work.

## Acknowledgements

The authors thank CNPq, CAPES and FAPERJ for their support during the preparation of this work. We would like also to thank Prof. L.-F. Alzuguir for the PIV data-set.

## References

[1] Y. Jang, R. P. Botchen, A. Lauser, D. S. Ebert, K. P. Gaither, and T. Ertl, "Enhancing the Interactive Visualization of Procedurally Encoded Multifield Data with Ellipsoidal Basis Functions," *Computer Graphics Forum*, vol. 25, no. 3, pp. 587–596, 2006.

[2] V. Vapnik, S. Golowich, and A. Smola, "Support vector method for function approximation, regression estimation, and signal processing," in *Advances in Neural Information Processing Systems 9*. MIT Press, 1997, pp. 281–287.

[3] R. Schaback and H. Wendland, "Kernel techniques: From machine learning to meshless methods," *Acta Numerica*, vol. 15, pp. 543–639, 2006.

[4] F. Mussa-Ivaldi, "From basis functions to basis fields: Vector field approximation from sparse data," *Biological Cybernetics*, vol. 67, pp. 479–489, 1992.

[5] M. Lage, F. Petronetto, A. Paiva, H. Lopes, T. Lewiner, and G. Tavares, "Vector field reconstruction from sparse samples," in *Sibgrapi*, 2006, pp. 297–304.

[6] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H.-P. Seidel, "Multi-level partition of unity implicits," *Transactions on Graphics*, vol. 22, no. 3, pp. 463–470, 2003.

[7] B. Schölkopf, J. Giesen, and S. Spalinger, "Kernel methods for implicit surface modeling," in *Advances in Neural Information Processing Systems 17*, 2005, pp. 1193–1200.

[8] F. Steinke, B. Schölkopf, and V. Blanz, "Support vector machines for 3D shape processing," *Computer Graphics Forum*, vol. 24, no. 3, pp. 285–294, 2005.

[9] B. K. P. Horn, *Robot vision*. MIT Press, 1986.

[10] J. Colliez, F. Dufrenois, and D. Hamad, "Optic flow estimation by support vector regression," *Engineering Applications of Artificial Intelligence*, vol. 19, no. 7, pp. 761–768, 2006.

[11] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer, 2000.

[12] B. Schölkopf and A. J. Smola, *Learning with Kernels*. MIT Press, 2002.

[13] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, pp. 199–222, 2004.

[14] L. Zhang, W. Zhou, and L. Jiao, "Wavelet support vector machine," *Systems, Man and Cybernetics*, vol. 34, no. 1, pp. 34–39, 2004.

[15] J. Shawe-Taylor and N. Cristianini, *Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[16] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001.

[17] R.-E. Fan, P.-H. Chen, and C.-J. Lin., "Working set selection using the second order information for training SVM," *Journal of Machine Learning Research 6*, pp. 1889–1918, 2005.

[18] J. Stam, "Stable fluids," in *Siggraph*, 1999, pp. 121–128.