Algebraic Curve Fitting Support Vector Machines

Christian J. Walder, Brian C. Lovell, and Peter J. Kootsookos

Intelligent Real-Time Imaging and Sensing Group,
School of Information Technology and Electrical Engineering,
The University of Queensland, Brisbane, Queensland 4072, Australia.
{walder, lovell, kootsookos}@itee.uq.edu.au

Abstract. An algebraic curve is defined as the zero set of a multivariate polynomial. We consider the problem of fitting an algebraic curve to a set of vectors given an additional set of vectors labelled as interior or exterior to the curve. The problem of fitting a linear curve in this way is shown to lend itself to a support vector representation, allowing non-linear curves and high dimensional surfaces to be estimated using kernel functions. The approach is attractive due to the stability of solutions obtained, the range of functional forms made possible (including polynomials), and the potential for applying well understood regularisation operators from the theory of Support Vector Machines.

1 Motivation

Algebraic curves provide a powerful basis for a range of geometrical analysis problems, including shape recognition and non-iterative shape registration, largely due to the capacity for deriving geometric invariants [5, 3, 6]. Geometric invariants are those properties of an algebraic curve that are not affected by a particular group of transformations, for example the affine group.

The fitting of an algebraic curve to a set of vectors has proven to be a difficult problem, with simple approaches such as least squares having little practical value due to the instability of the solutions obtained. In an attempt to improve stability a number of improvements have been made to the least squares approach. Two of the more successful approaches are the so-called 3L method of Lei, Blane and Cooper [2], and the Gradient-1 algorithm of Tasdizen, Tarel and Cooper [7]. Both of these methods require some geometrical knowledge in addition to the known set of points on the curve. The 3L method requires a sets interior and exterior points equidistant to the curve of interest. The Gradient-1 method requires the normal vector of the desired curve to be given for each data vector on the curve. Our method requires similar information to the 3L method, but without the equidistance constraint. Our approach is equivalent to a combination of Support Vector Machine (SVM) classification [9] of the interior/exterior points, in combination with a least squares penalty on the vectors that lie on the curve. Various classes of functions can be produced by the method, including the implicit polynomials considered by the 3L and Gradient-1 algorithms.

2 Least Squares Fitting

Perhaps the simplest approach to algebraic curve fitting is the minimisation of the least squares criterion. That is, given a set of points $\{\tilde{z}_i\}_{1\leq i\leq n}$ (subject to noise) that are known to lie on a curve, and a set of functions $\mathcal{F}$, the least squares heuristic chooses the function defined by $\min_{f\in\mathcal{F}}\sum_{i=1}^{n}(f(\tilde{z}_i))^2$. The main problem with this method is that it does not penalise extraneous parts of the curve that lie far from the data. Our modification to this simple and largely ineffective formulation turns out to be a variation of the standard hard-margin SVM classifier, and we begin by reviewing this method.

3 Support Vector Machines

The formulation of Vapnik's hard-margin SVM classifier is derived by first considering the problem of two-class linear data classification [9]. That is, given a data set $\{(\tilde{x}_i, y_i)\}_{1\leq i\leq n}$ of training samples $\tilde{x}_i \in \mathbb{R}^d$ belonging to classes labelled by $y_i \in \{-1, 1\}$, we wish to find a hyper-plane that separates the two classes. The SVM approach to this problem finds the separating hyper-plane with maximum distance to the $\tilde{x}_i$, while preserving the separation of the two classes. If we denote the hyper-plane by the set $\{\tilde{x} \in \mathbb{R}^d | <\tilde{w}, \tilde{x}> + b = 0\}$, then finding the optimal parameters $\tilde{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$ is equivalent to minimising $<\tilde{w}, \tilde{w}>$ subject to $y_i(<\tilde{w}, \tilde{x}_i> + b) \geq 1, i = 1 \ldots n$.

The most important property of this formulation is that the derivation of the Wolfe dual [1] results in an equivalent optimisation problem, but one in which the data vectors only appear in the form of inner products with one another. As a result of this, the so-called "kernel trick" [9] can be applied in order to find non-linear separating hyper-surfaces. We now demonstrate how this approach can be combined with the least squares criterion in order to perform algebraic curve fitting.

4 Kernel Based Algebraic Curve Fitting

Here we have a set of points, $\{\tilde{z}_1, \tilde{z}_2, \ldots \tilde{z}_{N_z}\} \subset \mathbb{R}^d$ that lie on our hyper-surface, as well as a set $\{(\tilde{x}_j, y_j)\}_{1\leq j\leq N_x}$ of labelled points $\tilde{x}_j \in \mathbb{R}^d$ with associated labels $y_j \in \{1, -1\}$, where $\tilde{x}_j$ is interior to the hyper-surface of interest if $y_j = 1$, and exterior if $y_j = -1$.

Our proposed method of utilising this information is to perform SVM classification of the $\tilde{x}_j$ with a squared error penalty on the value of the function at the $\tilde{z}_i$. To this end, we begin with the following optimisation problem, which is equivalent to hard-margin linear SVM classification of the $\tilde{x}_j$, with a penalty proportional to c associated with the square of the functional value ($f(\tilde{z}_i) =< \tilde{w}, \tilde{z}_i > +b$) at each $\tilde{z}_i$:

$$\begin{array}{l}
\underline{\text{Minimise}}\ _{\tilde{w},b} \\
< \tilde{w}, \tilde{w} > +c\sum_{i=1}^{N_z} \xi_i^2 \\
\underline{\text{Subject To}}: \\
< \tilde{w}, \tilde{z}_i > +b \leq \xi_i, \quad i = 1 \dots N_z \\
< \tilde{w}, \tilde{z}_i > +b \geq -\xi_i, \quad i = 1 \dots N_z \\
y_j(< \tilde{w}, \tilde{x}_j > +b) \geq 1, \quad j = 1 \dots N_x
\end{array}$$

Note that choosing $c = 0$ makes the first two constraints superfluous, and we are left with the standard hard-margin SVM. We now proceed to derive the Wolfe dual problem [1]. In the following equations, the summations over i are to be taken from 1 to N_z, and those over j from 1 to N_x. The Lagrangian function is then:

$$\begin{aligned}
L = \tfrac{1}{2} < \tilde{w}, \tilde{w} > +\tfrac{1}{2}c\sum_i \xi_i^2 - \sum_i \alpha_i(\xi_i - < \tilde{w}, \tilde{z}_i > -b) \\
- \sum_i \alpha_i^*(\xi_i + < \tilde{w}, \tilde{z}_i > +b) - \sum_j \gamma_j[y_j(< \tilde{w}, \tilde{x}_j > +b) - 1] \qquad (1)
\end{aligned}$$

By the convexity of the problem, at the optimal solution we have:

$$\frac{\partial L}{\partial \tilde{w}} = \tilde{0} = \tilde{w} + \sum_i(\alpha_i - \alpha_i^*)\tilde{z}_i - \sum_j \gamma_j y_j \tilde{x}_j \qquad (2)$$

$$\frac{\partial L}{\partial b} = 0 = \sum_i(\alpha_i - \alpha_i^*) - \sum_j \gamma_j y_j \qquad (3)$$

$$\frac{\partial L}{\partial \xi_i} = 0 = c\xi_i - (\alpha_i + \alpha_i^*) \qquad (4)$$

So $\tilde{w}$ has the "support vector expansion":

$$\tilde{w} = \sum_j \gamma_j y_j \tilde{x}_j - \sum_i \beta_i \tilde{z}_i$$

where we have written $\beta_i = \alpha_i - \alpha_i^*$. We now use the above relations to eliminate the primal variables from (1). Putting (2) and (3) into (1) leads to:

$$L = \tilde{e}^{\mathsf{T}}\tilde{\gamma} + \tfrac{1}{2}c\sum_i \xi_i^2 - \sum_i(\alpha_i + \alpha_i^*)\xi_i - \tfrac{1}{2}\tilde{\beta}^{\mathsf{T}}H_z\tilde{\beta} - \tfrac{1}{2}\tilde{\gamma}^{\mathsf{T}}H_x\tilde{\gamma} + \tilde{\gamma}^{\mathsf{T}}H_{xz}\tilde{\beta}$$

where $\tilde{e}$ is a vector of ones, and we have defined the matrices H_x, H_z and H_{xz} with elements defined as follows:

$$\begin{aligned}
[H_x]_{j,j'} &= y_j y_{j'} < \tilde{x}_j, \tilde{x}_{j'} > \\
[H_z]_{i,i'} &= < \tilde{z}_i, \tilde{z}_{i'} > \\
[H_{xz}]_{i,j} &= y_j < \tilde{z}_i, \tilde{x}_j >
\end{aligned}$$

Now we use (4), which implies $\frac{1}{2}c\sum_i \xi_i^2 - \sum_i(\alpha_i + \alpha_i^*)\xi_i = -\frac{1}{2}\sum_i(\alpha_i + \alpha_i^*)\xi_i$, and we have:

$$L = \tilde{e}^{\mathsf{T}}\tilde{\gamma} - \tfrac{1}{2}\sum_i \frac{(\alpha_i + \alpha_i^*)^2}{c} - \tfrac{1}{2}\tilde{\beta}^{\mathsf{T}}H_z\tilde{\beta} - \tfrac{1}{2}\tilde{\gamma}^{\mathsf{T}}H_x\tilde{\gamma} + \tilde{\gamma}^{\mathsf{T}}H_{xz}\tilde{\beta} \qquad (5)$$

Since the solution depends on $\alpha_i - \alpha_i^*$ rather than either of the α_i or α_i^* individually, the positivity constraint on the Lagrange multipliers α_i and α_i^* allows us to remove an unnecessary degree of freedom by constraining $\alpha_i \alpha_i^* = 0$, and so that we can write $\frac{(\alpha_i + \alpha_i^*)^2}{c} = \frac{|\beta_i|^2}{c} = \frac{\beta_i^2}{c}$, and the expression for L becomes:

$$L = \tilde{e}^{\mathsf{T}}\tilde{\gamma} - \tfrac{1}{2}\tilde{\beta}^{\mathsf{T}}(H_x + C)\tilde{\beta} - \tfrac{1}{2}\tilde{\gamma}^{\mathsf{T}}H_z\tilde{\gamma} + \tilde{\gamma}^{\mathsf{T}}H_{xz}\tilde{\beta} \tag{6}$$

where C is a diagonal matrix with entries $\frac{1}{c}$.

The final matrix form of our dual problem is then:

$$\begin{array}{l} \underline{\text{Minimise}}\ _{\tilde{\beta},\tilde{\gamma}} \\ \tfrac{1}{2}\tilde{\beta}^{\mathsf{T}}(H_x + C)\tilde{\beta} + \tfrac{1}{2}\tilde{\gamma}^{\mathsf{T}}H_z\tilde{\gamma} - \tilde{\gamma}^{\mathsf{T}}H_{xz}\tilde{\beta} - \tilde{e}^{\mathsf{T}}\tilde{\gamma} \\ \underline{\text{Subject To:}} \\ \tilde{e}^{\mathsf{T}}\tilde{\beta} - \tilde{y}^{\mathsf{T}}\tilde{\gamma} = 0, \\ \tilde{\gamma} \geq \tilde{0} \end{array}$$

As is the case with the primal problem, setting c to zero leads to an equivalence with the standard hard-margin SVM. To see this, note that as the c approaches zero, any non-zero β_i will cause the objective function above to approach the value ∞. This guarantees that at the optimal solution all the β_i will equal zero. The remainder of the problem (containing only the $\tilde{\gamma}$ terms) is equivalent to the Lagrangian dual form of the standard hard-margin SVM problem, as can be seen by comparing with [9].

Note that the optimal value of b can be found directly from the Karush-Kuhn-Tucker (K.K.T.) optimality conditions [1], which are in this case:

$$\begin{aligned} \alpha_i(\xi_i - <\tilde{w}, \tilde{z}_i> - b) &= 0 \\ \alpha_i^*(\xi_i + <\tilde{w}, \tilde{z}_i> + b) &= 0 \\ \gamma_j[y_j(<\tilde{w}, \tilde{x}_j> + b) - 1] &= 0 \end{aligned}$$

Since only the inner products of the data appear in the dual problem, we can find non-linear hyper-surfaces by using a kernel function K that corresponds to an inner product after mapping under Φ to some feature space $\mathcal{H}$, that is,

$$K(\tilde{x}, \tilde{x}') = <\phi(\tilde{x}), \phi(\tilde{x}')>_{\mathcal{H}}, \forall \tilde{x}, \tilde{x}' \in \mathbb{R}^d$$

The point, familiar to the SVM community, is that we do not need to know explicitly the mapping $\Phi : \mathbb{R}^d \rightarrow \mathcal{H}$ in order to find non-linear hyper-surfaces, we simply replace the inner products $< \tilde{x}, \tilde{x}' >$ in our dual problem with kernel function evaluations $K(\tilde{x}, \tilde{x}')$, and the corresponding hyper-surface is the set $\{\forall \tilde{x} | f(\tilde{x}) = 0\}$, where $f(\tilde{x})$ has become:

$$f(\tilde{x}) = \sum_j \gamma_j y_j K(\tilde{x}_j, \tilde{x}) - \sum_i \beta_i K(\tilde{z}_i, \tilde{x}) + b$$

5 Results and Discussion

The method has been implemented using Platt's Sequential Minimal Optimisation algorithm [4] to solve the Quadratic Programming (QP) problem. First, we construct from our data set the matrices appearing in the final form of our dual problem. These matrices are passed to the QP solver, which returns the optimal values of the β_i and γ_j. Next, we solve for b, which is attained directly from the K.K.T. conditions (see Sec. 4), and we have completely defined our implicit function according to the final equation given in Sec. 4. In each of the two following subsections, we investigate the results obtained using one of two different kernel functions.

5.1 Polynomial Kernel Function

The kernel function used in this subsection is the so-called "polynomial kernel" [9]:

$$K(\tilde{x}, \tilde{x}') = (< \tilde{x}, \tilde{x}' > +1)^d$$

The free parameter $d \in \mathbb{N}^+$ corresponds to the order of our implicit polynomial. In general, choosing a greater value of d makes possible a more complex family of curves.

In the first test, simple morphological operations were used to preprocess the binary image of Figure 1 into sets of interior, exterior, and edge points, as marked in Figure 2 by the crosses, plus-signs and dots, respectively. The figure also includes the zero contours of the resultant function for various values of c. An 8th order polynomial kernel was used for all the curves of Figure 2, that is, we chose $d = 8$. Note that the same image was used in a similar test in both [2] and [7]. We have added noise to the $\tilde{x}_j$ in order demonstrate robustness with respect to the location of these vectors. We note in passing however that an equidistant set of $\tilde{x}_j$ (as required by the 3L method) would result in further improvements of stability and accuracy.

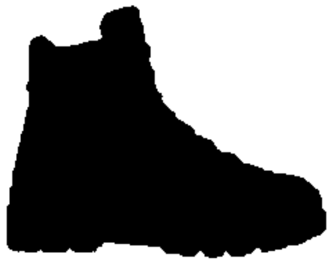
Fig. 1. Original shoe test image.

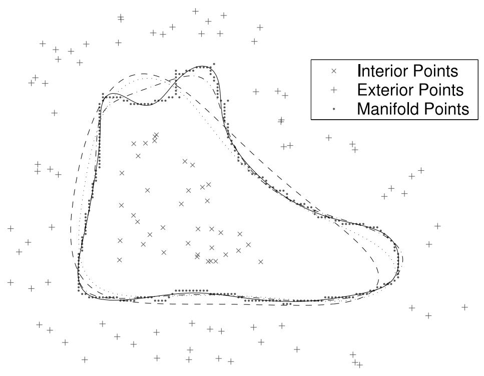

Fig. 2. Results for the polynomial kernel, degree 8, as a function of c (c values - dashed line: 0, dotted: 0.02, dash-dot: 10, solid: 10,000).

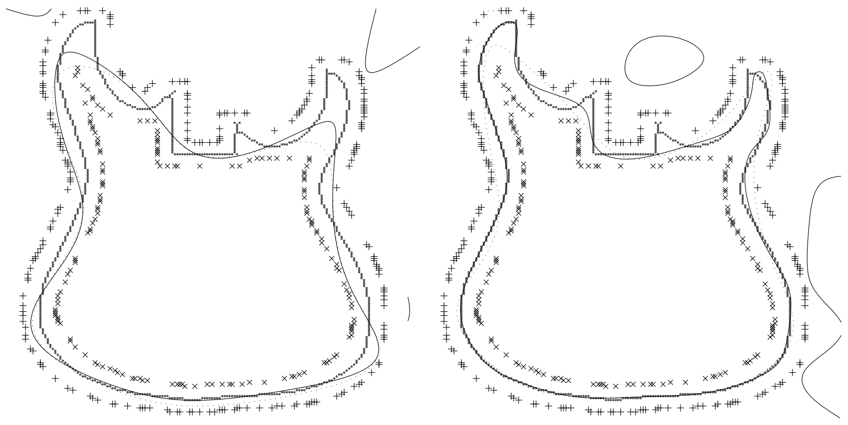

Fig. 3. Results for the polynomial kernel, degree 12, as a function of c (c values – left-hand side – dotted line: 0, solid line: 0.00042184; right-hand side – dotted line: 0.042184, solid line: 42.184).

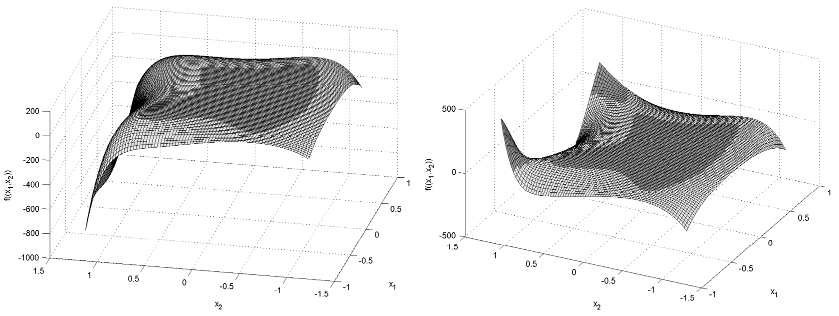

Fig. 4. 3-Dimensional rendering of the implicit polynomials corresponding to the $c = 0$ (left-hand side) and $c = 0.00042184$ (right-hand side) solutions of Figure 3

A more complex data set is depicted in Figure 3, which demonstrates our observation that the desired manifold tends to be approximated poorly by the polynomial kernel (as we increase c) in those regions where the $c = 0$ solution is not already close to the desired manifold. Correspondingly, choosing a “tighter” set of $\tilde{x}_i$ in the neighbouring region tends to improve the performance. In general we found that continually increasing the c value eventually results in the appearance of extraneous components of the zero set, as is the case in Figure 3 for the $c = 0.00042184$ solution, which is also rendered in Figure 4. Note the correspondence between the extraneous parts of the zero set shown in Figure 3, with the three dimensional rendering of the same function in Figure 4.

5.2 Rational Polynomial Kernel Function

In this subsection we improve upon the results in the previous subsection by using instead the following kernel function:

$$k(\tilde{x}, \tilde{x}') = \frac{1}{\varepsilon + \|\tilde{x} - \tilde{x}'\|^2} = \frac{1}{\varepsilon + < \tilde{x} - \tilde{x}', \tilde{x} - \tilde{x}' >}, \varepsilon \in \mathbb{R}^+$$

This kernel function, which does not appear to have been used previously in the context of the SVM, has for our application several advantages over the polynomial kernel of the previous subsection. In particular, it is trivial to show that using this kernel guarantees the boundedness of the resultant algebraic curve, a property which has previously been imposed by more direct means, as for example by Taubin et. al. [8]. Additionally, the kernel function (and therefore the resultant algebraic curve) is independent of the absolute position of the data vectors. Finally the kernel function can be considered (for small ε), as an approximation to the inverse square of Euclidean distance. This is desirable according to the widely held view within the SVM community that a kernel function should represent a good similarity metric for the problem it is being applied to.

Although the kernel function is a rational polynomial, since the denominator is non-zero, we can rewrite our implicit equation as an implicit (non-rational) polynomial. To do this, we simply multiply the original equation by the product of the denominators of each of the summands. That is, as an alternative to the original equation (for simplicity we write ξ_i instead of both β_i and $\gamma_j y_j$):

$$f(\tilde{x}) = \sum_i \xi_i \frac{1}{\varepsilon + < \tilde{x} - \tilde{x}_i, \tilde{x} - \tilde{x}_i >} + b$$

we have the following non-rational polynomial equation, which has an identical zero set:

$$f'(\tilde{x}) = \sum_i \xi_i \prod_{j \neq i} (\varepsilon + < \tilde{x} - \tilde{x}_j, \tilde{x} - \tilde{x}_j >) + b \prod_i (\varepsilon + < \tilde{x} - \tilde{x}_i, \tilde{x} - \tilde{x}_i >)$$

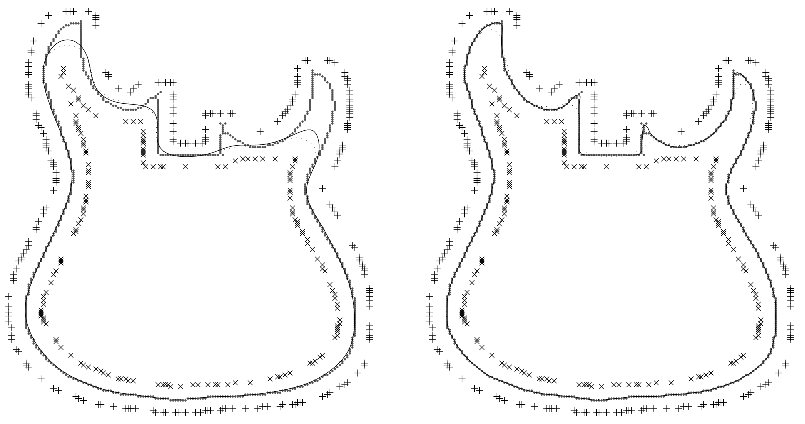

Fig. 5. Results for the rational polynomial kernel, $\epsilon = 0.05$, as a function of c (c values – left-hand side – dotted line: 0, solid line: 0.012296; right-hand side – dotted line: 0.12296, solid line: 122.9618).

It is important to note that the order of the above polynomial depends on the number of non-zero Lagrange multipliers, rather than being chosen a priori as is the case with the polynomial kernel of Sec. 5.1. In fact, the order of the polynomial will be equal to twice the number of non-zero Lagrange multipliers, typically resulting in very high order polynomials.

The improvement in performance afforded by the rational polynomial kernel function is evident in Figure 5, in which the solutions are well behaved under the variation of the c parameter. We have found that the solutions are well behaved under a wide range of c values. As a result, a c value can easily be found that produces good results on a wide range of different data sets. This is demonstrated in Figure 6, where we have used identical processing for all of the test images. The interior and exterior points were obtained from the input image by taking the edges of six-pixel morphological erosions and dilations of the original binary image, respectively. Two thirds of these interior and exterior points were then selected at random and discarded.

Although this simplistic approach produced a relatively poor set of points on the guitar image (ie. the “hole” in the guitar is unaccounted for), the method succeeded in producing the correct topology. The only image which resulted in a solution with an incorrect topology was the butterfly image, however it could be argued that this is the fault of the preprocessing step, and that the cluster of points around the abdomen of the butterfly are nonsensical in terms of defining a contour. Note that the correct topology would be produced for some $c \leq 30$. Moreover, the butterfly example is included here for illustrative purposes only - such errant input data could easily be avoided with a slightly more sophisticated preprocessing step.

6 Conclusions

One of the major difficulties faced by algebraic curve fitting algorithms is the topological instability of the solutions. We have met this problem with the inherent regularisa-

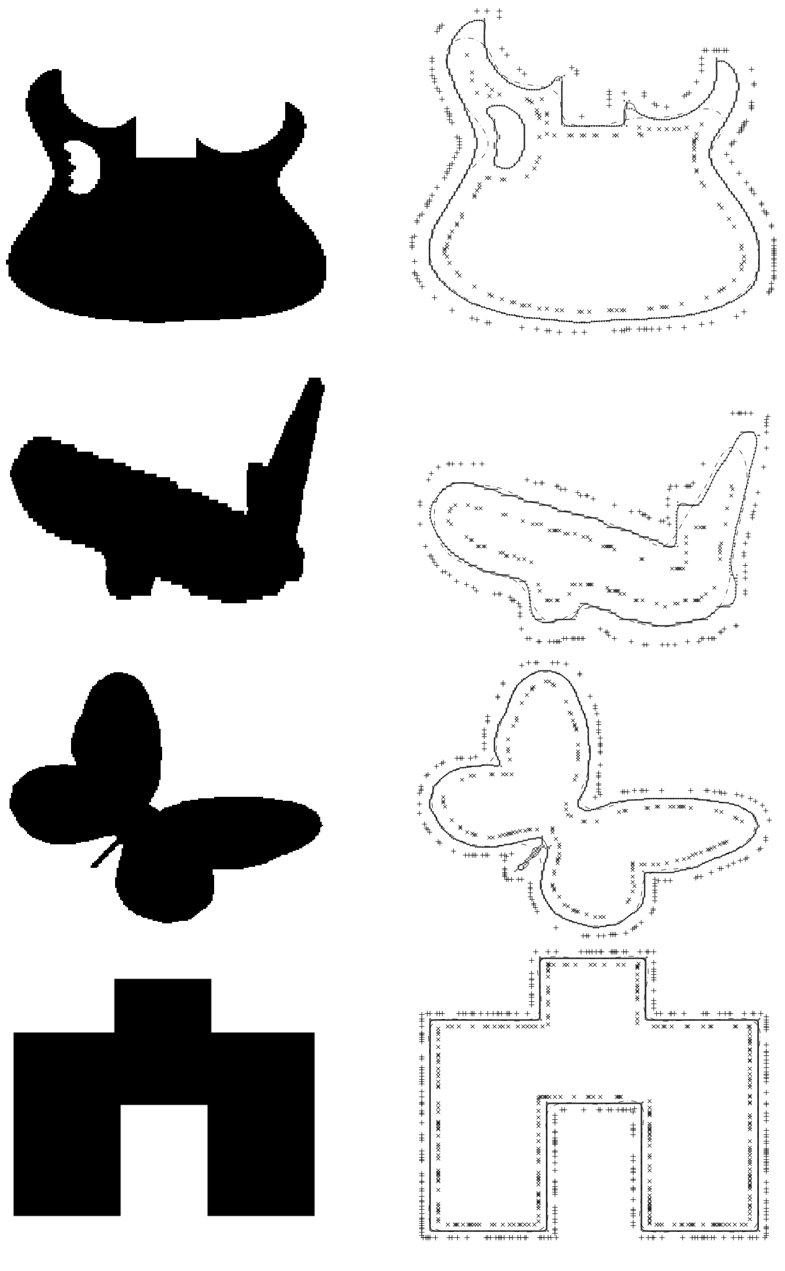

Fig. 6. Results for the rational polynomial kernel, $\epsilon = 0.05$. The original binary images are on the left-hand side. The same parameters were used for all of the results shown on the right-hand side: dash-dot line - $c = 0$, solid line - $c = 30$. The $c = 30$ solutions are largely obscured by the data points. *NB: This figure is best viewed by "zooming in" on an electronic copy of the document.*

tion of the SVM classifier in conjunction with a least squares penalty on the data points. We have given two main options for fitting algebraic polynomial curves.

The first, using the polynomial kernel of Sec. 5.1, allows the order of the polynomial to be chosen a priori. While this approach is useful in simple cases, it is unsuitable for complex shapes that are not coupled with a suitably precise set of interior/exterior points, as the incorrect topology may result.

The second method involves using the rational kernel of Sec. 5.2. This method allows the accurate estimation of complex shapes, with minimal requirements on the accuracy of the interior/exterior points. In this case, the order of the polynomial is determined by the complexity of the data at hand, often resulting in high order polynomials.

Finally, it is interesting to compare the algorithm with the standard hard-margin SVM, in the problem domain of machine learning (data classification). In this perspective the $\tilde{z}_i$ could be considered as training vectors that an expert has labelled "too hard to classify". The algorithm uses this information by trying to find a decision function that places the class membership boundary near the $\tilde{z}_i$. A possible application for the algorithm as a data classifier is in the domain of handwriting recognition, in which a human could label some sample characters as being, say, "either an eight or a six".

References

1. M.S. Bazaraa, H.D. Sherali, and C.M. Shetty. *Nonlinear Programming: Theory and Algorithms*. John Wiley and Sons, Inc., 1997.
2. Z. Lei, Michael M. Blane, and David B. Cooper. 3L fitting of higher degree implicit polynomials. In *Proceedings of Third IEEE Workshop on Applications of Computer Vision*, Sarasota, Florida, USA, 1996.
3. Z. Lei, T. Tasdizen, and D.B. Cooper. Pims and invariant parts for shape recognition. In *Proceedings of Sixth International Conference on Computer Vision*, pages 827–832, Mumbai, India, 1998.
4. J. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines, 1998.
5. J.-P. Tarel, H. Civi, and D. B. Cooper. Pose estimation of free-form 3D objects without point matching using algebraic surface models. In *Proceedings of IEEE Workshop on Model-Based 3D Image Analysis*, pages 13–21, Mumbai, India, 1998.
6. Jean-Philippe Tarel, William A. Wolovich, and David B. Cooper. Covariant conics decomposition of quartics for 2d object recognition and affine alignment. In *Proceedings of the International Conference on Image Processing (ICIP'98)*, pages 818–822, Chicago, Illinois, USA, 1998.
7. T. Tasdizen, J.-P. Tarel, and D.B. Cooper. Algebraic curves that work better. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages II:35–41, Fort Collins, Colorado, USA, 1999.
8. G. Taubin, F. Cukierman, S. Sullivan, J. Ponce, and D.J. Kriegman. Parameterized families of polynomials for bounded algebraic curve and surface fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:287–303, 1994.
9. Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, 1995.