UF College of Engineering
Department of Computer & Information Science & Engineering
UNIVERSITY of FLORIDA

Technical Report

UF CISE–TR 472, 2009

Conic Section Classifier: A Novel Conecpt Class with a Tractable Learning Algorithm

Santhosh Kodipaka, Arunava Banerjee, Baba C. Vemuri

Submitted to IEEE Trans. on Pattern Analysis and Machine Intelligence – March 16, 2009

CVGMI

Center for Vision, Graphics and Medical Imaging

http://www.cise.ufl.edu/research/cvgmi/

E331, CSE Building, PO BOX 116120, Gainesville, FL 32611, USA; +1 (352) 392 5770

Conic Section Classifier: A New Concept Class with a Tractable Learning Algorithm

Santhosh Kodipaka, Arunava Banerjee, and Baba C. Vemuri
April 10, 2009

Abstract—In several computer vision and medical diagnosis applications, features used for supervised learning are often high-dimensional and the available samples are sparse. This leads to a severely under-constrained learning problem. One can approach this either by reducing the feature dimensionality or by limiting the classifier to a simpler concept class. We propose a new concept class suited for such data sets, that is based on conic sections. Each class is represented by a conic section in the input space, described by its focus (point), directrix (hyperplane) and eccentricity (value). Class labels are assigned to data-points based on the eccentricities attributed to them by the class descriptors. The concept class can represent non-linear discriminant boundaries with merely four times the number of parameters as a linear discriminant. Learning involves updating the class descriptors. We also present a tractable learning algorithm for binary classification. For each descriptor, we track its feasible space that results in identical labeling for classified points. We show favorable learning performance compared to many state-of-the-art classifiers on several data sets.

Index Terms—Machine learning, Concept learning, Classifier design and evaluation, Geometric algorithms

1 Introduction

Many notable problems in medical diagnosis, object recognition, text-categorization, etc., can be posed in the general framework of supervised learning theory. The learning problem in such instances can be formulated as follows: One is given a dataset of N labeled tuples $\{\langle X_1, y_1\rangle, \ldots, \langle X_N, y_N\rangle\}$, where each X_i is a data point represented in some input space $\mathcal{X}$, and y_i is it's associated class label from an output space $\mathcal{Y}$. The input space $\mathcal{X}$ in which the data points, X_i's, lie can be any appropriately defined space with some degree of structure. In the majority of cases $\mathcal{X}$ is set to be the Euclidean space $\mathbb{R}^M$. $\mathcal{Y}$ is in general a two-element output space. The ultimate goal of learning is to find the best function $f : \mathcal{X} \to \mathcal{Y}$ (called the *concept*) that minimizes the *expected risk* on $\mathcal{X}$. A lower bound on the *expected risk*, given a set of training data, is furnished by the *empirical risk* or *training error* defined as $\frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(y_i \neq f(X_i))$, where $\mathbb{I}$ is the indicator function.

Formulated as above, the learning problem is still ill-posed since there exist uncountably many functions, f's, that yield zero empirical risk. To make this problem well-defined, one further restricts the f's to a particular class of functions, known as the *concept class*, and subsequently identifies the best member of that class which minimizes the empirical risk. Such a *classifier* might however over-fit the training data and perform poorly on as yet unseen data points. A notion of generalizability for a given classifier is therefore introduced as a regularizer to control the expected risk on unseen data. Since the distribution of data in $\mathcal{X}$ is not known a priori, a conservative bound on the generalization capacity of a classifier can be quantified as a function of the classifier's empirical error and a formalization of the complexity of the classifier's concept class. Although Statistical Learning Theory [1] does provide formal bounds for generalization error, such bounds are often weak. The common practice therefore is to estimate the generalization error via such protocols as the holdout method, cross-validation and bootstrapping, as reviewed in [2]. The classifier from the concept class that yields the least generalization error, empirically measured using one of the techniques above, is chosen for the purpose of future classification.

This research was in part supported by NIH RO1 NS046812 to BCV.
Authors are with the Department of Computer & Information Science & Engineering, University of Florida, Gainesville, FL 32611. Email: {snsk,arunava,vemuri}@cise.ufl.edu

1.1 Motivation

Without detailed prior knowledge regarding the nature of a dataset, it is not possible in principle to predict which of a given set of concept classes will yield the smallest generalization error. Practitioners therefore resort to applying as many classifiers with different concept classes as possible, before choosing the one that yields the least generalization error. *Every new concept class with a corresponding tractable learning algorithm is consequently a potential asset to a practitioner since it expands the set of classifiers that can be applied to a dataset.*

The learning task becomes remarkably difficult when the number of training samples available is far fewer than the number of features used to represent each sample. We encounter such high dimensional sparse datasets in several applications like the diagnosis of Epilepsy based on brain MRI scans [3], the diagnosis of various types of Cancer from micro-array gene expression

data [4], spoken letter recognition [5] and object recognition from images [6], to name only a few. The supervised learning problem is severely under constrained when one is given N labeled data points that lie in $\mathbb{R}^M$ where $N \ll M$. This situation arises whenever the "natural" description of a data point in the problem domain is very large and the cost of collecting large number of labeled data points is prohibitive.

In such scenarios, learning even a simple classifier such as a linear discriminant is under-constrained because one has to solve for $M+1$ parameters given only N constraints. Additional objectives, such as maximizing the functional margin of the discriminant, are usually introduced to fully constrain the problem. The learning problem becomes progressively difficult as the concept class gets richer, since such concepts require larger number of parameters to be solved for, given the same number of constraints. This often leads to overfitting and the generalization capacity of the classifier suffers.

There are two kinds of traditional solutions to this quandary. In the first approach, the classifier is restricted to the simplest of concept classes like the Fisher Discriminants [7], the Linear Support Vector Machine (SVM), etc. In the second approach, the dimensionality of the dataset is reduced either by a prior feature selection [8], [9] or by projecting the data onto discriminative subspaces [10]. The criterion for projection may or may not incorporate discriminability of the data, such as in PCA versus Large Margin Component Analysis [11], respectively. The assumption underlying the second approach is that there is a smaller set of compound features that is sufficient for the purpose of classification. Our principal contribution in this paper, expands the power of the first approach noted above, by presenting a novel concept class along with a tractable learning algorithm, well suited for high-dimensional sparse data.

1.2 Synopsis

We introduce a novel concept class based on conic sections in Section 2. Each member class in the dataset is assigned a conic section in the input space, parameterized by its focus point, directrix plane and a scalar valued eccentricity. The eccentricity of a point is defined as the ratio between its distance to a fixed focus and to a fixed directrix plane. The focus and directrix descriptors of each class attribute eccentricities to all points in the input space $\mathbb{R}^M$. A data point is assigned to the class to which it is closest in eccentricity value. The concept class is illustrated in Figure-1. The resultant discriminant boundary for two-class classification turns out to be a pair of polynomial surfaces of at most degree 8 in $\mathbb{R}^M$ and thus has finite VC dimension [12]. Yet, it can represent these highly non-linear discriminant boundaries with merely four times the number of parameters as a linear discriminant. In comparison, a general polynomial boundary of degree d requires $O(M^d)$ parameters, where M is the dimensionality of the input space.

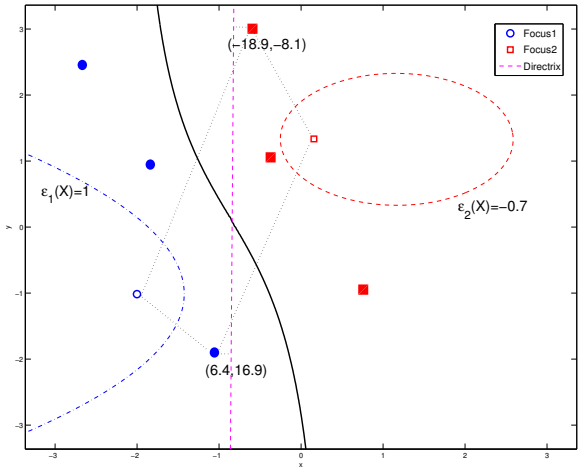


Fig. 1. Overview of the concept class. Circles and squares are data points from classes 1, & 2. The parabola and the ellipse are the class conic sections with eccentricities 1, & -0.7. Both the classes share a common directrix line here. The faint dotted lines represent distances of points to the foci and directrices. For one point in each class, the eccentricities attributed by the class descriptors are shown. Points are assigned to the class to which they are closest to, in eccentricity value. The thick curve is the resultant discriminant.

Given a labeled dataset, learning involves arriving at appropriate pair of conic sections (i.e., their directrices, foci, and eccentricities) for the classes, that reduces empirical risk and results in a discriminant that is simpler and hence more generalizable.In Section 3, we present a tractable geometric algorithm for binary classification, that updates the class descriptors in an alternating manner. This paper expands upon preliminary results presented in [13] by tracking larger feasible spaces for the class conic section descriptors. We demonstrate the efficacy of our technique in Section 4, by comparing it to well known classifiers like Linear and Kernel Fisher Discriminants and kernel SVM on several real datasets. Our classifier consistently performed better than LFD, as desired. In the majority of cases, it out-performed state-of-the-art classifiers. We discuss concluding remarks in Section 5. A list of symbols used to represent different geometric entities in this paper are given in the Appendix.

1.3 Related Work

Conic sections have been used extensively in several Graphics and Computer Vision problems like curve fitting [14], [15], and recovering conics from images [16] to infer structure from motion, etc. The principal reasons for this usage is that a large variety of curves can be represented with very few parameters and that the conic sections are very common in occurrence. Within the domain of supervised learning, there is one instance in

which this notion was used. One can obtain a conic section by intersecting a cone with a plane at a certain angle. The angle is equivalent to the eccentricity and when varied results in different conic sections. This notion was combined with neural networks in [17] to learn such an angle at each node. However, the other descriptors, namely the focus and directrix are fixed at each node unlike our approach.

Support Vector Machines (SVM) and Kernel Fisher Discriminant (KFD) with polynomial kernel also yield polynomial boundaries like our method. The set of discriminants due to these classifiers can have a non-empty intersection with those due to the conic section concept class. That is, they do not subsume the boundaries that result from the latter and vice-versa. We emphasize here that there is no known kernel equivalent to the conic section concept class for SVM or KFD, and hence the concept class is indeed novel. A detailed comparison to these concept classes is presented in Section 2.1.

1.4 Contributions

In this paper, we introduce a concept class based on conic sections accompanied by a geometric learning algorithm. The concept class has finite VC dimension, can represent highly non-linear boundaries with merely $4 * (M + 1)$ parameters, and subsumes linear discriminants. In the learning phase for two-class classification, we track a feasible space for each descriptor in $\mathbb{R}^M$that results in identical labeling of classified points. The feasible space is represented as a compact geometric object, from which desirable descriptor updates are chosen. We reduce the computation of linear subspace in which the compact geometric object lies, into that of a Gram-Schmidt orthogonalization. We also employ a *stiffness* criterion that is used to pursue simpler discriminants instead of highly non-linear ones, thereby performing model selection in the learning phase. The performance of the concept class on several high dimensional sparse datasets is comparable to and sometimes better than state-of-the-art techniques.

2 The Conic-Section Concept Class

A conic section in $\mathbb{R}^2$ is defined as the locus of points whose distance from a given point (the *focus*) and that from a given line (the *directrix*), form a constant ratio (the *eccentricity*). Different kinds of conic sections such as ellipse, parabola and hyperbola, are obtained by fixing the value of the eccentricity to < 1, $= 1$, and > 1, respectively. Conic sections can be defined in higher dimensions by making the directrix a hyperplane of co-dimension 1. Together, a fixed focus point and directrix hyperplane generate an *eccentricity function* (Eqn.1) that attributes to each point $X \in \mathbb{R}^M$ a scalar valued eccentricity defined as:

$$\varepsilon(X) = \frac{\|X - F\|}{b + Q^T X} \; ; \; where \; \|Q\| = 1 \tag{1}$$

Hereafter, we use $\|.\|$ to denote the Euclidean $\mathbb{L}_2$ norm. $F \in \mathbb{R}^M$ is the focus point and $(b + Q^T X)$ is the orthogonal distance of X to the directrix represented as $\{b, Q\}$, where $b \in \mathbb{R}$ is the offset of the directrix from the origin and $Q \in \mathbb{R}^M$ is the unit vector that is normal to the directrix. The locus of points that correspond to $\varepsilon(X) = e$ is an axially symmetric conic section in $\mathbb{R}^M$. At $e = 0$, the conic section collapses to the focus point. As $|e| \to \infty$, it becomes the directrix hyperplane itself.

We are now in a position to formally define the concept class for binary classification. Each class, $k \in \{1, 2\}$, is represented by a distinct conic section parameterized by the descriptor set: focus, directrix and eccentricity, as $C_k = \{F_k, (b_k, Q_k), e_k\}$. For any given point X, each class attributes an eccentricity $\varepsilon_k(X)$, as defined in Eqn.1, in terms of the descriptor set C_k. We refer to $\langle \varepsilon_1(X), \varepsilon_2(X) \rangle$ as the class *attributed eccentricities* of X. We label a point as belonging to that class k, whose class eccentricity e_k is closest to the sample's class attributed eccentricity $\varepsilon_k(X)$, as in Eqn.2. The label assignment procedure is illustrated in Figure-1.

$$class(X) = argmin_k(|\varepsilon_k(X) - e_k|) \tag{2}$$

$$g(X) = |\varepsilon_1(X) - e_1| - |\varepsilon_2(X) - e_2| \tag{3}$$

The resultant discriminant boundary (Eqn.3) is the locus of points that are equidistant to the class representative conic sections, in eccentricity. The discriminant boundary is defined as $\mathcal{G} \equiv \{X : g(X) = 0\}$ where $g(X)$ is given by Eqn.3. The discriminant boundary equation can be expanded to:

$$(\frac{r_1}{h_1} - e_1) = \pm(\frac{r_2}{h_2} - e_2), \tag{4}$$

$$\Rightarrow ((r_1 - e_1 h_1)h_2) = \pm((r_2 - e_2 h_2)h_1), \quad where$$

$$r_k(X) = \sqrt{(X - F_k)^T (X - F_k)} \tag{5}$$

$$h_k(X) = X^T Q_k + b_k \tag{6}$$

Upon re-arranging the terms in Eqn.4 and squaring them, we obtain a pair of degree-8 polynomial surfaces, in X, as the discriminant boundary.

$$(\ (r_1 h_2)^2 + (r_2 h_2)^2 - ((e_1 \mp e_2) h_1 h_2)^2\ )^2 - (2 r_1 r_2 h_1 h_2)^2 = 0 \tag{7}$$

Depending upon the choice of the conic section descriptors, $\{C_1, C_2\}$, the resultant discriminant can yield lower order polynomials as well. The boundaries due to different class conic configurations in $\mathbb{R}^2$, are illustrated in Figure-2. When the normals to the directrices, Q_1, Q_2, are not parallel, the discriminant is highly non-linear (Figure-2(a)). A simpler boundary is obtained when directrices are coincident, as in Figure-2(b). We obtain linear boundaries when either directrices perpendicularly bisect the line joining the foci (Figure-2(c)) or the directrices are parallel and the eccentricities are equal (Figure-2(d)). A list of symbols used in this paper are given in the Appendix.

The concept class, referred to as Conic Section classifier (CSC) hereafter, has several notable features. Learning involves arriving at conic descriptors so that the

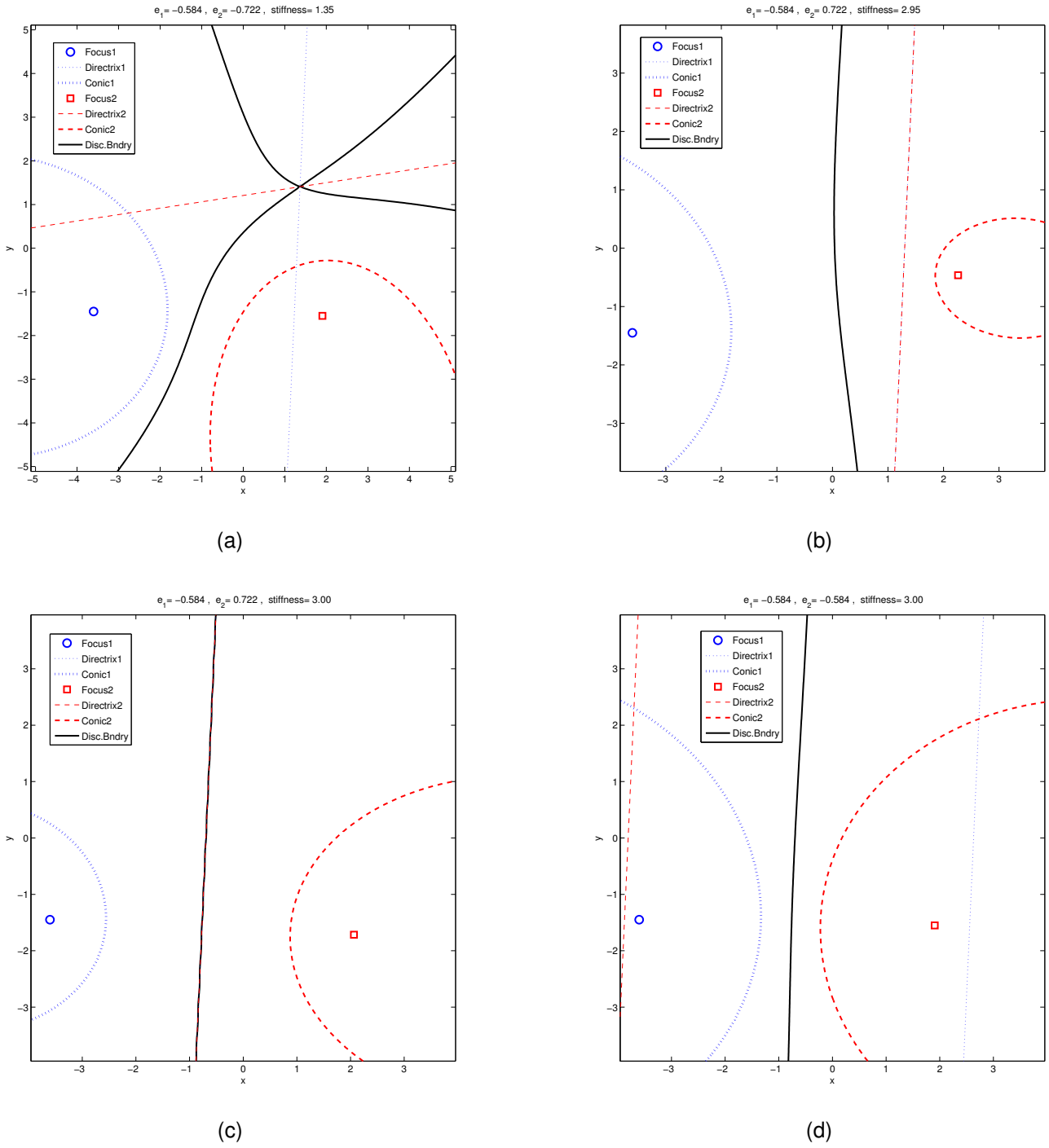


Fig. 2. Discriminant boundaries for different class conic configurations in $\mathbb{R}^2$: (a) Non-linear boundary for a random configuration. (b) Simpler boundary: directrices are coincident. Sign of e_2 is flipped to display its conic section. (c) Linear boundary: directrices perpendicularly bisect the line joining foci. (d) Linear boundary: directrices are parallel and eccentricities are equal. (See Sec-2)

given data samples are well separated. Our learning technique, presented in Section 3, pursues boundaries that are simple and ensures large functional margins. Our intent is not to fit conic sections to samples from each class, but learn a generalizable boundary between classes with fewer parameters. Regardless of the dimensionality of the input space, the discriminant is always linear under certain conditions. The linear discriminants can be arrived at, when the directrices for the two classes are identical, the foci lie symmetrically on the two opposite sides of the directrix, the line joining the foci is normal to the directrix and/or the class eccentricities are equal and lie in a certain range near zero. The concept class therefore subsumes linear discriminants. We can obtain boundaries ranging from simple to complex, by varying the class conic descriptors. The number of parameters necessary to specify discriminant boundaries due to the conic section concept class is $4*(M+1)$. This is far less than the M^2 parameters required for even a general quadratic surface.

2.1 Comparisons to other Classifiers

We compare CSC to Support Vector Machine (SVM) and Kernel Fisher Discriminants (KFD) with polynomial kernels as they appear to be related to CSC in the type of discriminants represented. For both the classifiers, the discriminant boundary can be defined as:

$$b + \sum_{i=1}^{N} w_i (X^T X_i + 1)^d = 0 \tag{8}$$

where w is a weight vector. Here the decision surface is a linear combination of N degree-d polynomial surfaces defined from each of the data points X_i. The methods differ in their generalization criteria to arrive at the weights w_i. Note that the discriminants due to CSC (Eqn.7) cannot be expressed in terms of the boundary due to (Eqn.8). There could be a non-empty intersection between the set of the polynomial surfaces represented by CSC and those due to Eqn.8. We point out that there is no kernel which matches this concept class, and therefore, the concept class is indeed novel.

KFD seeks boundaries that maximize the Fisher criterion [7], *i.e.* maximize inter-class separation while minimizing within class variance. The learning criterion used in SVM is to pursue large functional margin, resulting is lower VC dimension [18] and thereby improving generalizability. CSC uses a similar criterion and in fact goes a step further. The degree of the polynomial kernel is a model-selection parameter in kernel based classifiers like SVM and Kernel Fisher Discriminants (KFD). As CSC learning will involve arriving at simpler boundaries for the same constraints on training samples, the degree of the polynomial is also being learnt in effect. (See Figure-2).

The advantage of SVM over CSC is that learning in the former involves a convex optimization. The optimization in KFD is reduced to that of a matrix inverse problem for binary classification. However, the equivalent numerical formulation for CSC turns out to be non-convex and intractable. We therefore use novel geometric approaches to represent the entire feasible space for constraints on each of the conic descriptors and then pick a local optimum. In fact, we reduced the feasible space pursuit problem in CSC into that of a Gram-Schmidt orthogonalization [19] (Section 3.3.2).

When SVM deals with high-dimensional sparse datasets, most of the data points end up being support vectors themselves, leading to about $N*(M+1)$ parameters whereas CSC employs only $4*(M+1)$ parameters. The boundary due to KFD also involves the same number of parameters as SVM. In summary, CSC has some unique benefits over the state-of-the-art techniques that makes it worth exploring. These include incorporating quasi model-selection into learning, and shorter description of the discriminant boundary. We have also found that CSC out-performed SVM and KFD with polynomial kernel in many classification experiments as listed in Table-2.

3 The Learning Algorithm

We introduce a novel incremental algorithm for the two-class Conic Section Classifier in this section. We assume a set of N labeled samples $P = \{\langle X_1, y_1\rangle, \ldots, \langle X_N, y_N\rangle\}$, where $X_i \in \mathbb{R}^M$ and the label $y_i \in \{-1, +1\}$, to be sparse in a very high dimensional input space such that $N \ll M$. Learning involves finding the conic section descriptors, $\{C_1, C_2\}$, that can minimize empirical learning risk and simultaneously result in simpler discriminant boundaries. The empirical risk, L_{err}, is defined as:

$$L_{err} = \frac{1}{N} \sum_i \mathbb{I}(\ y_i \cdot g(X_i) > 0\ ) \tag{9}$$

where $\mathbb{I}$ is the indicator function. A brief description of the algorithm is presented next.

3.1 Overview

In the learning phase, we perform a constrained update to one conic descriptor at a time, holding the other descriptors fixed; *the constraint being that the resultant boundary continues to correctly classify points that are already correctly classified in previous iteration*. The feasible space for each descriptor within which these constraints are satisfied will be referred to as its *Null Space*. We pick a solution from each *Null Space* in a principled manner such that one or more misclassified points are learnt. The sets of descriptors that will be updated alternately are $\{\langle e_1, e_2\rangle, F_1, F_2, \{b_1, Q_1\}, \{b_2, Q_2\}\}$.

To begin with, we initialize the focus and directrix descriptors for each class such that the *Null Spaces* are large. The initialization phase is explained in detail in Section 3.5. The subsequent learning process is comprised of two principal stages. In the first stage, given fixed foci and directrices we compute attributed eccentricities

Input: Labeled Samples P
Output: Conic Section Descriptors C_1, C_2

Initialize the class descriptors $\{F_k, \{b_k, Q_k\}\}, k \in \{1, 2\}$
repeat
  Compute $\langle \varepsilon_1(X_i), \varepsilon_2(X_i) \rangle \ \ \forall X_i \in P$
  Find the best *class-eccentricities* $\langle e_1, e_2 \rangle$
  for each descriptor $\{F_1, F_2, \{b_1, Q_1\}, \{b_2, Q_2\}\}$
    Determine the classifying range for $\langle \varepsilon_{1i}, \varepsilon_{2i} \rangle$
    Find its feasible space due to these constraints.
    for each misclassified point X_{mc}
      Compute a descriptor update to learn X_{mc}
    end for
    Pick updates with least *empirical* error
    Then pick an update with largest *Stiffness*
  end for
until the descriptors C_1, C_2 converge

Fig. 3. Learning the class descriptors C_1, C_2

(Eqn.1) for each point X_i, denoted as $\langle \varepsilon_{1i}, \varepsilon_{2i} \rangle$. We then compute an optimal pair of class eccentricities $\langle e_1, e_2 \rangle$, that minimizes the empirical risk L_{err} in $O(N^2)$ time, as described in Section 3.2.1. For a chosen descriptor (focus or directrix) to be updated, we find feasible intervals of desired attributed eccentricities, such that $y_i \cdot g(X_i) > 0 \ \forall X_i$, *i.e.,* the samples are correctly classified (Section 3.2.2).

In the second stage, we solve for the inverse problem. Given *class-eccentricities*, we seek a descriptor solution that causes attributed eccentricities to lie in the desired feasible intervals. This results in a set of geometric constraints on the descriptor, due to Eqn.4, that are dealt with in detail in Section 3.3. We compute the entire *Null Space*, defined as the equivalence class of the given descriptor that results in the same label assignment for the data.For each misclassified point, we pick a solution in this *Null Space*, that learns it with a large margin while ensuring a simpler decision boundary in $\mathbb{R}^M$. In Section 3.4.1, we introduce a *stiffness* criterion to quantify the extent of the non-linearity in a discriminant boundary. Among the candidate updates due to each misclassified point, an update is chosen that yields maximum *stiffness*, *i.e.*, minimal non-linearity. The second stage is repeated to update the foci $\{F_1, F_2\}$ and the directrices $\{\{b_1, Q_1\}, \{b_2, Q_2\}\}$, one at a time. The two stages are alternately repeated until either the descriptors converge or there can be no further improvement in classification and *stiffness*. Note that all through the process, *the learning accuracy is non-decreasing since a previously classified point is never misclassified due to subsequent updates.* A summary of the algorithm is listed in Figure-3. We discuss the first phase in detail in the following section.

3.2 Learning in Eccentricity Space

The focus and directrix descriptors of both the classes induce a non-linear mapping, $\varepsilon^*(X)$, due to Eqn.1, from the input space into $\mathbb{R}^2$, as:

$$\varepsilon^*(X) = \langle \varepsilon_1(X), \varepsilon_2(X) \rangle \tag{10}$$

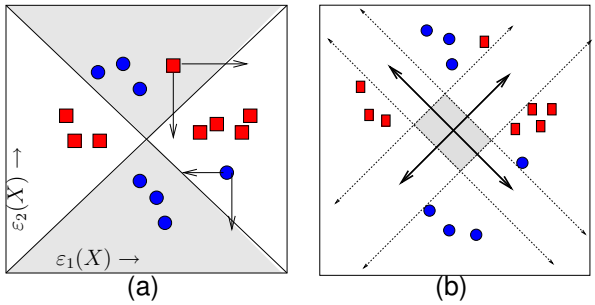


Fig. 4. (a) Shaded regions in this *ecc-Space* belong to the class with label -1. Learning involves updating the eccentricity maps so as to shift the misclassified points into desired regions. (b) The discriminant boundary is the pair of thick lines. Within the shaded rectangle, any choice of *class eccentricity* descriptors (the point of intersection of two thick lines) results in identical classification.

This space of attributed eccentricities will be referred to as the eccentricity space (*ecc-Space*). We defined this space so as to determine the best pair of class eccentricities simultaneously. In Figure-4(a), the x and y axes represent the maps $\varepsilon_1(X)$ and $\varepsilon_2(X)$ respectively, as defined in Eqn.1. For any given choice of class eccentricities, e_1 and e_2, the discriminant boundary equivalent in *ecc-Space* , $|\varepsilon_1(X) - e_1| - |\varepsilon_2(X) - e_2| = 0$, becomes a pair of mutually orthogonal lines with slopes $+1, -1$, respectively, as illustrated in the figure. These lines intersect at $\langle e_1, e_2 \rangle$, which is a point in *ecc-Space* . The lines divide *ecc-Space* into four quadrants with opposite pairs belonging to the same class. It should be noted that this discriminant corresponds to an equivalent non-linear decision boundary in the input space $\mathbb{R}^M$. We use *ecc-Space* only as a means to explain the learning process in the input space. The crucial part of the algorithm is to learn the eccentricity maps 1 for each class by updating the foci and directrices. In this section, we first present an $O(N^2)$ time algorithm to find the optimal class eccentricities. Next, we determine resultant constraints on the focus and directrix descriptors due to the classified points.

3.2.1 *Finding Optimal Class-Eccentricities* $\langle e_1, e_2 \rangle$

We now present an $O(N^2)$ algorithm to find the optimal pair of class-eccentricities resulting in the least possible empirical error (Eqn.9), given fixed foci and directrices. The discriminant boundary (Eqn.3) in *ecc-Space* is completely defined by the location of class eccentricities. Consider a pair of orthogonal lines with slopes $+1$ and -1 respectively, passing through each mapped sample in *ecc-Space*, as illustrated in Figure-4(b). Consequently, these pairs of lines partition the *ecc-Space* into $(N+1)^2$ 2D rectangular intervals. We now make the critical observation that *within the confines of such a 2D interval, any choice*

of a point that represents class eccentricities results in identical label assignments (see Figure-4(b)). Therefore, the search is limited to just these $(N+1)^2$ intervals. The interval that gives the smallest classification error is chosen. The cross-hair is set at the center of the chosen 2D interval to obtain large functional margin. Whenever, there are multiple 2D intervals resulting in the least empirical error, the larger interval is chosen so as to obtain a larger margin.

3.2.2 Geometric Constraints on Foci and Directrices

Having fixed the class eccentricities, the learning constraint on attributed eccentricities for each X_i, with a functional margin $\delta > 0$, is given by Eqn.11. Assuming that the descriptors of class 2 are fixed, the constraint on $\varepsilon_1(X_i)$ due to Eqn.11 is derived in Eqn.12. Now, if we are interested in updating only focus F_1, the constraints on F_1 in terms of distances to points, X_i, are given by Eqn.13.

$$y_i\left(|\varepsilon_1(X_i) - e_1| - |\varepsilon_2(X_i) - e_2|\right) > \delta \quad (11)$$

$$\Rightarrow y_i\left(|\varepsilon_1(X_i) - e_1|\right) > (\delta + y_i\Delta\varepsilon_{2i}) = w_{1i}$$

$$l_{1i} = (y_i e_1 - w_{1i}) > y_i\varepsilon_1(X_i) > (y_i e_1 + w_{1i}) = u_{1i} \quad (12)$$

$$l_{1i} > y_i\frac{||F_1 - X_i||}{h_{1i}} > u_{1i} \quad (13)$$

Here $\Delta\varepsilon_{2i} = |\varepsilon_2(X_i) - e_2|$, and h_{1i} is the distance to the directrix hyperplane of class 1 (Eqn.4). In Eqn.13, the only unknown variable is F_1. Similarly, we can obtain constraints for all the other descriptors. Whenever the intervals due to the constraints are unbounded, we apply bounds derived from the range of attributed eccentricities in the previous iteration. The margin, δ, was set to 1% of this range.

In the second stage of the learning algorithm, we employ a novel geometric technique to construct the *Null Space* of say, F_1, for distance constraints (Eqn.13) related to the currently classified points. Next, we pick a solution from the *Null Space* that can learn a misclassified point by satisfying its constraint on F_1, if such a solution exists. The learning task now reduces to updating the foci and directrices of both the classes alternately, so that the misclassified points are mapped into their desired quadrants in *ecc-Space*, while the correctly classified points remain in their respective quadrants. Note that with such updates, *our learning rate is non-decreasing*. In the next section, we construct *Null Spaces* for the focus and directrix descriptors. In Section 3.4 we deal with learning misclassified points.

3.3 Constructing *Null Spaces*

Assume that we have N_c classified points and a point X_{mc} that is misclassified. We attempt to modify the discriminant boundary by updating one descriptor at a time in $\mathbb{R}^M$ such that the point X_{mc} is correctly classified. The restriction on the update is that all the classified points remain in their respective class quadrants in *ecc-Space* , *i.e*, their labels not change. In this section, we construct feasible spaces for each of the focus and directrix descriptors within which the labeling constraint is satisfied.

3.3.1 The Focus Null Space

Here, we consider updating F_1, the focus of class 1. For ease of readability, we drop the reference to class from here on, unless necessary. First, we construct the *Null Space* within which F adheres to the restrictions imposed by the following N_c quadratic constraints, due to Eqn.13,:

$$r_{li} < \|F - X_i\| < r_{ui}\ \forall i \in 1, 2, \ldots N_c \quad (14)$$

where r_{li}, and r_{ui} are lower and upper bounds on the distance of X_i to F. In effect, each point X_i requires F to be at a certain distance from itself, lying in the interval (r_{li}, r_{ui}). Next, we need to pick a solution in the *Null Space* that satisfies a similar constraint on X_{mc} so as to learn it, and improve the generalization capacity of the classifier (Section 3.4). While this would otherwise have been an NP-hard problem (like a general QP problem), the geometric structure of these quadratic constraints enables us to construct the *Null Space* in just $O(N^2M)$ time. Note that by assumption, the number of constraints $N \ll M$.

The *Null Space* of F with respect to each constraint in Eqn.14 is the space between two concentric hyperspheres in $\mathbb{R}^M$, referred to as a *shell*. Hence, the *Null Space* for all the constraints put together is the intersection of all the corresponding shells in $\mathbb{R}^M$. This turns out to be a complicated object. However, we can exploit the fact that the focus in the previous iteration, denoted as F°, satisfies all the constraints in Eqn.14 since it resulted in N_c classified points. To that end, we first construct the locus of all focus points, F', that satisfy the following equality constraints:

$$\|X_i - F'\| = \|X_i - F^\circ\| = r_i\ ,\ \forall i \in 1 \ldots N_c \quad (15)$$

Note that such an F' will have the same distances to the classified data points X_i as the previous focus F°, so that the values of the discriminant function at X_i remain unchanged, *i.e.*, $g(X_i, F') \equiv g(X_i, F^\circ)$. Later, we will use the locus of all F' to construct a subspace of the *Null Space* related to Eqn.14 that also has a much simpler geometry.

3.3.2 Intersection of Hyperspheres

The algorithm we discuss here incrementally builds the *Null Space* for the equality constraints in Eqn.15; *i.e.*, the locus of all foci F' that are at distance r_i to the respective classified point X_i. The *Null Space* is initialized as the set of feasible solutions for the first equality constraint in Eqn.15. It can be parameterized as the hypersphere $S_1 = (r_1, X_1)$, centered at X_i with radius r_i. Next, the second equality constraint is introduced, the *Null Space* for which, considered independently, is the hypersphere $S_2 = (r_2, X_2)$. Then the combined *Null Space* for the two

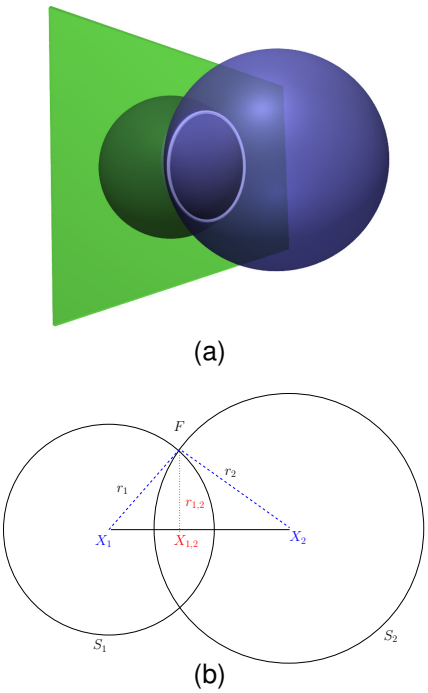


Fig. 5. (a)Intersection of two hypersphere *Null Spaces*, $S_1(r_1, X_1)$ and $S_2(r_2, X_2)$ lying in a hyperplane. Any point on the new *Null Space* (bright-circle) satisfies both the hypersphere (distance) constraints. (b) $S_1 \cap S_2$ can be parameterized by the hypersphere $S_{\{1,2\}}$ centered at $X_{1,2}$ with radius $R_{1,2}$.

constraints is the intersection of the two hyperspheres, $S_1 \cap S_2$.

As illustrated in Figure-5(a), the intersection of two spheres in $\mathbb{R}^3$ is a circle that lies on the plane of intersection of the two spheres. The following solution is based on the analogue of this fact in $\mathbb{R}^M$. We make two critical observations: *the intersection of two hyperspheres is a hypersphere of one lower dimension, and this hypersphere lies on the intersecting hyperplane of the original hyperspheres*. Each iteration of the algorithm involves two steps. In the first step, we re-parameterize the combined *Null Space*, $S_1 \cap S_2$, as a hypersphere $S_{\{1,2\}}$ of one lower dimension lying in the hyperplane of intersection $H_{\{1,2\}}$. Based on the geometry of the problem and the parameterization of S_1 and S_2, shown in Figure-5(b), we can compute the radius and the center of the new hypersphere $S_{\{1,2\}} = (r_{\{1,2\}}, X_{\{1,2\}})$ in $O(M)$ time, given by Eqns.16-18. We can also determine the intersecting hyperplane $H_{\{1,2\}}$ represented as $\{b_{\{1,2\}}, Q_{12}\}$. The first descriptor $b_{\{1,2\}}$ is the displacement of $H_{\{1,2\}}$ from the origin and the other descriptor is the unit vector normal to $H_{\{1,2\}}$. In fact, $Q_{\{1,2\}}$ lies along the line joining X_1 and X_2. The parameters of the new hypersphere $S_{\{1,2\}}$ and the hyperplane $H_{\{1,2\}}$ are computed as :

$$\begin{aligned} Q_{1,2} &= (X_2 - X_1)/\|X_2 - X_1\| && (16)\\ X_{1,2} &= X_1 + Q_{1,2}Q_{1,2}^T(F^\circ - X_1) && (17)\\ r_{1,2} &= \|X_{1,2} - F^\circ\| \\ b_{1,2} &= -Q_{1,2}^T X_{1,2} && (18) \end{aligned}$$

In the second step, the problem for the remaining equality constraints is reposed on the hyperplane $H_{\{1,2\}}$. This is accomplished by intersecting each of the remaining hyperspheres $S_3, ., S_{N_c}$ that correspond to the samples $X_3, ., X_{N_c}$, with $H_{\{1,2\}}$, in $O(NM)$ time. Once again, based on the geometry of the problem, the new centers of the corresponding hyperspheres can be computed using Eqn.17 and their radii are given by:

$$r'_i = \sqrt{r_i^2 - ((X_i - F^\circ)^T Q_{1,2})^2}$$

In short, the intersection of the N_c hyperspheres problem is converted into the intersection of $(N_c - 1)$ hyperspheres in the hyperplane $H_{\{1,2\}}$, as summarized below:

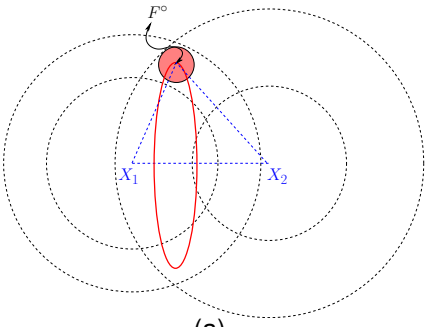
$$\begin{aligned} S_1 \cap S_2 &\rightarrow S_{\{1,2\}} \in H_{\{1,2\}} \\ S_i \cap H_{\{1,2\}} &\rightarrow S'_i \in H_{\{1,2\}} \qquad \forall i = 3, ., N_c \\ S_1 \cap S_2 ... \cap S_{N_c} &\rightarrow S_{1,2} \cap S'_3 .. \cap S'_{N_c} \in H_{1,2} \qquad (19) \end{aligned}$$

The problem is now transparently posed in the lower dimensional hyperplane $H_{\{1,2\}}$ as a problem equivalent to the one that we began with, except with one less hypersphere constraint. The end result of repeating this process $(N_c - 1)$ times yields a *Null Space* that satisfies all the equality constraints (Eqn.15), represented as a single hypersphere lying in a low dimensional linear subspace and computed in $O(N^2M)$ time. It should be observed that all the intersections thus far are feasible and that the successive *Null Spaces* have non-zero radii since the equality constraints have a feasible solution *a priori*, *i.e.*, F°.

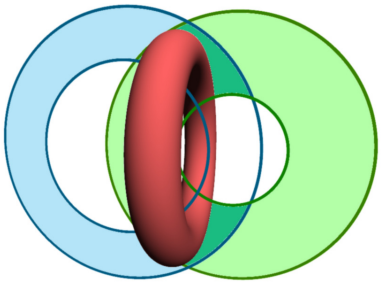
Upon unfolding Eqns.16,17 over iterations, we notice that the computation of the normal to the hyperplane of each intersection and that of the new center can be equivalently posed as a Gram-Schmidt orthogonalization [19]. The process is equivalent to the QR decomposition of the following matrix:

$$\begin{aligned} \mathbf{A} = \begin{bmatrix} \tilde{X}_2 & \tilde{X}_2 & \dots & \tilde{X}_{N_c} \end{bmatrix} = \mathbf{Q}R \qquad (20)\\ \text{where, } \tilde{X}_i = (X_i - X_1) \\ C_e = \pi(X_1) = X_1 + \mathbf{Q}\mathbf{Q}^T(F^\circ - X_1) \qquad (21)\\ r_e = \|F^\circ - C_f\| \end{aligned}$$

The function $\pi(X)$ in Eqn.21 projects any point into a low dimensional linear subspace, $\mathcal{H}$, normal to the unit vectors in $\mathbf{Q}$ (Eqn.20) and contains F°. $\mathcal{H}$ can be defined as the linear subspace $\{X \in \mathbb{R}^M : \mathbf{Q}^T(X - F^\circ) = 0\}$. We use existing stable and efficient QR factorization routines to compute $\mathcal{S}^e$ in $O(N^2M)$ time. *It is noteworthy that the final* Null Space *due to the equality constraints in Eqn.15 can be represented as a single hypersphere* $\mathcal{S}^e = (r_e, C_e) \subset \mathcal{H}$. The center and radius of $\mathcal{S}^e$ can be computed using Eqn.21. The geometry of $\mathcal{S}^e$ enables us to generate sample focus points that always satisfy the equality constraints. In the next section, we pursue larger regions within which the inequality constraints on the focus due to Eqn.14 are satisfied.



(a)



(b)

Fig. 6. Cross-section of *shell-shell* intersection in $\mathbb{R}^3$. The red disc in (a) is the largest disc centered at F° that is within the intersection. The thick circle passing through F° and orthogonal to the cross-section plane, is the locus of all foci with same distance constraints as F°. A product of the disc and the thick circle results in a toroid, as in (b), which is a tractable subregion lying within the intersection of *shells*.

3.3.3 Intersection of Shells

Consider the intersection of two *shells* centered at X_1, X_2, in $\mathbb{R}^3$, as illustrated in Figure-6. We first compute the largest disc centered at the previous focus, F°, which is guaranteed to lie within the intersection. Next, we revolve the disc about the line joining X_1 and X_2. The resultant object is a doughnut like toroidal region, that can be defined as a product of a circle and a disc in $\mathbb{R}^3$, as illustrated in Figure-6(a). The circle traced by F° is the locus of all foci F that are at the same distances to points X_1, X_2, as F°. We pursue this idea in $\mathbb{R}^M$. The final *Null Space* , say $\mathcal{S}^f$, that satisfies the inequality constraints on F (Eqn.14) can be defined as :

$$\mathcal{S}^f \equiv \{F = F' + \beta U : F' \in \mathcal{S}^e, \|U\| = 1, \beta \in [0, \alpha)\} \quad (22)$$

where F' is a point from the feasible space, $\mathcal{S}^e$, due to the equality constraints on F (Eqn.15), and $\alpha > 0$ is the radius of the largest solid hypersphere (ball) at all F' that satisfies the inequality constraints. In this manner, we intend to add a certain α thickness to the locus of F'.

We can compute the radius, α, of the largest disc at F° from Eqns.14,22. Let $r_i = \|X_i - F^\circ\|$, $F = F^\circ + \beta U$ be a point on the ball at F°. Due to triangle inequality between F, F° and any point X_i, $\forall \beta \in [0, \alpha)$ we have:

$$\begin{aligned} & |r_i - \beta| \leq \|(X_i - F^\circ) - \beta U\| \leq r_i + \beta \\ \Rightarrow \quad & r_{li} < |r_i - \beta| \leq \|X_i - F\| \leq r_i + \beta < r_{ui} \\ \Rightarrow \quad & \beta < r_i - r_{li}, \ \beta < r_{ui} - r_i \\ \Rightarrow \quad & \alpha = \min\{(r_{ui} - r_i), (r_i - r_{li})\}_{i=1\ldots N_c} \end{aligned} \quad (23)$$

Here $\beta < r_i$ so as to satisfy Eqn.14. We can thus compute α from Eqn.23 in just $O(N)$ time given the distance bounds r_{li}, and r_{ui}. With very little computational expense, we can track a larger *Null Space* than that due to the equality constraints (Eqn.15).

3.3.4 The Directrix Null Space

Here, we present a two-step technique to construct the *Null Space* for a directrix that satisfies the learning constraints in Eqn.12. First, we list constraints on the directrix so that the classified points remain in their quadrants when mapped into *ecc-Space* . Second, we reduce the problem of constructing the resultant feasible space into that of a focus *Null Space* computation.

The constraints on the hyperplane descriptor set $\{b, Q\}$, due to those on attributed eccentricities in Eqn.12, $\forall i = 1 \ldots N_c$, are :

$$h_{li} < \left(b + Q^T X_i\right) < h_{ui} \quad \text{with } Q^T Q = 1 \quad (24)$$

$$\Rightarrow \quad h_{li} - h_{u1} < Q^T (X_i - X_1) < h_{ui} - h_{l1}$$

$$\Rightarrow \quad \tilde{h}_{li} < Q^T \tilde{X}_i < \tilde{h}_{ui} \quad (25)$$

where h_{li}, and h_{ui} are lower and upper bounds on the distance of X_i to the hyperplane $\{b, Q\}$. We assume every other descriptor is fixed except the unknown $\{b, Q\}$. Upon subtracting the constraint on X_1 from the other constraints, we obtain Eqn.25, where $\tilde{X}_i = (X_i - X_1)$, $\tilde{h}_{li} = (h_{li} - h_{u1})$, and $\tilde{h}_{ui} = (h_{ui} - h_{l1})$. We can now convert the distances to hyperplane constraints to those of distances to a point, by considering $\|Q - \tilde{X}_i\|$ as in Eqn.26. Let $z_i = (1 + \|\tilde{X}_i\|^2)$. The resultant *shell* constraints on Q, $\forall i = 1 \ldots N_c$ are in Eqn.27

$$\|Q - \tilde{X}_i\|^2 = 1 + \|\tilde{X}_i\|^2 - 2Q^T \tilde{X}_i \quad (26)$$

$$\Rightarrow \quad (z_i - 2\tilde{h}_{ui}) < \|Q - \tilde{X}_i\|^2 < (z_i - 2\tilde{h}_{li}) \quad (27)$$

Thus given N_c inequality constraints on point distances to Q and given the previous Q°, we use the solution from the focus update problem to construct a *Null Space* for Q. The only unknown left is b which lies in an interval due to X_1 in Eqn.24, given Q. We choose b to be the center of that interval, as $b = (h_{u1} + h_{l1})/2 - Q^T X_1$, so that the margin for X_1 is larger in *ecc-Space*.

3.4 Learning Misclassified Points

In order to keep the learning process tractable, we learn one misclassified point, X_{mc}, at a time. The final *Null Space* , $\mathcal{S}^f$, for the inequality constraints on F can be defined as all $F = F' + \beta U$, where $F' \in \mathcal{S}^e \subset \mathcal{H}$, $\beta \in [0, \alpha)$, and U is any normal vector. To determine if $\mathcal{S}^f$ has a solution that can learn X_{mc}, we intersect $\mathcal{S}^f$ with a

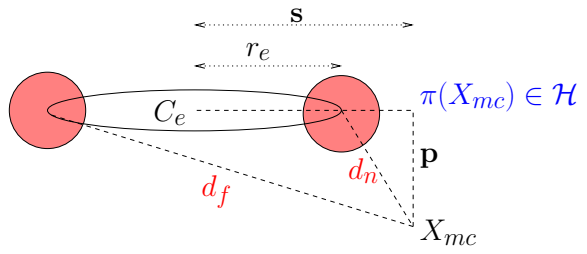


Fig. 7. The cross-section of the toroidal *Null Space*, $\mathcal{S}^f$, lying in the plane spanned by $\{X_{mc}, \pi(X_{mc}), C_e\}$. The expressions for distances d_n and d_f in terms of p,s, and r_e are given in Eqn.28

shell corresponding to the inequality constraint (Eqn.14) of X_{mc}, say $r_l < \|F - X_{mc}\| < r_u$. This is equivalent to checking if the range of $\|F - X_{mc}\|, \forall F \in \mathcal{S}^f$ intersects with the interval (r_l, r_u). This range can be determined by computing the distance to the nearest and farthest points on $\mathcal{S}^e$ to X_{mc}, as in Eqn. 28. First, we project X_{mc} into the linear subspace $\mathcal{H}$, in which $\mathcal{S}^e$ lies, using Eqn.21 to obtain $\pi(X_{mc})$. Let $p = \|X_{mc} - \pi(X_{mc})\|$ and $s = \|C_e - \pi(X_{mc})\|$. We then have :

$$d_n = \sqrt{(s - r_e)^2 + p^2}$$
$$d_f = \sqrt{(s + r_e)^2 + p^2} \quad (28)$$
$$d_n - \alpha < \|F - X_{mc}\| < d_f + \alpha \quad (29)$$

where (r_e, C_e) are the radius and center of $\mathcal{S}^e$. See Figure-7 to interpret these relations. Now the intersection of $\mathcal{S}^f$ and the *shell* corresponding to X_{mc} is reduced to that of the intervals: $(d_n - \alpha, d_f + \alpha) \cap (r_l, r_u)$. If they don't intersect, we can easily pick a (focus) point on $\mathcal{S}^f$ that is either nearest or farthest to the *shell* related to X_{mc}, so as to maximally learn X_{mc}, *i.e.*, change $r(X_{mc})$ which in turn maximizes $y_{mc} \cdot g(X_{mc})$. If they do intersect, there are only a few cases of intersections possible due to the geometry of $\mathcal{S}^f$ and the *shell*. Each such intersection turns out to be or contains another familiar object like $\mathcal{S}^f$, a *shell*, or a solid hypersphere (ball). Whenever the intersection exists, we pick a solution in it that maximizes $\sum y_i\, g(X_i)$ for the other misclassified points, *i.e.*, a solution closer to satisfying all the inequality constraints put together. In this manner, we compute a new update for each misclassified point. Next, we describe a criterion to pick an update, from the set of candidate updates, that can yield simplest possible discriminant.

3.4.1 Stiffness Measure

We introduce a *stiffness* measure that quantifies the extent of the non-linearity of the conic section classifier. It is defined as:

$$\Gamma(C_1, C_2) = |Q_1^T Q_2| + |Q_1^T F_{1,2}| + |Q_2^T F_{1,2}| \quad (30)$$

where, $F_{1,2}$ is the unit normal along the line joining the foci. The maximum of this measure corresponds to the configuration of conic sections that can yield a linear discriminant as discussed in Section 2. The measure is defined for focus and directrix descriptors. It can be used to pick an update yielding the largest *stiffness* so that the resultant discriminant can be most generalizable. We noticed that the non-linearity of the discriminant boundary is primarily dependent upon the angle between the directrix normals, Q_1 and Q_2. In Figure-2, the stiffness measures for different types of boundaries are listed. The discriminant boundary becomes simpler when the directrix normals are made identical. Moreover, the functional dependencies between the distance constraints which determine the discriminant, in Eqn.4, become simpler if the line joining the foci is parallel to the directrix normals. It stems from the observation that every point in the hyperplane that perpendicularly bisects the line joining the foci, is at equal distance from the foci ($r_1(X) \equiv r_2(X)$). Linear discriminants can be guaranteed when the stiffness is maximum and the class eccentricities are either equal in magnitude or near zero. Two configurations that yielded linear discriminants are illustrated in Figures-2(c),2(d). From the final *Null Space* of the descriptor being updated, we determine a descriptor update favored by each misclassified point. Among the competing updates, we choose the update with the least *empirical* error and maximum *stiffness*. Thus, the large *stiffness* pursuit incorporates a quasi model selection into the learning algorithm.

3.5 Initialization

Our learning algorithm is equivalent to solving a highly non-linear optimization problem that requires the final solution to perform well on both seen and as yet unseen data. Naturally, the solution depends considerably on the choice of initialization. As expected, random initializations converged to different conic descriptors leading to inconsistent performance. We observed that owing to the eccentricity function (Eqn.1), the *Null Spaces* become small (at times, vanishingly small) if the focus or directrix descriptors are placed very close to the samples. We found the following data-driven initializations to be consistently effective in our experiments. The initialization of all descriptors were done so as to start with a linear discriminant obtained from either linear SVM or Fisher [7] classifier or with the hyperplane that bisects the line joining the class means. The class eccentricities were set to $\langle 0, 0 \rangle$. The foci were initialized to lie far from their respective class samples. We initialized normals $Q_1 = Q_2$ with that in the initial linear discriminant. Now, b_1, b_2 were initialized to be either equal or far from their respective class clusters. If the data was not well separated with the current initialization, the foci and directrices were pushed apart until they were outside the smallest sphere containing the samples.

3.6 Discussion

One of the core characteristics of our algorithm is that after each update any point that is correctly classified by the earlier descriptors is not subsequently misclassified.

TABLE 1
Details of data used in experiments.

Dataset	Features	Samples	Class 1	Class 2
Epilepsy	216	44	19	25
Colon Tumor	2000	62	40	22
Leukemia	7129	72	47	25
CNS	7129	60	21	39
ETH-Objects	16384	20	10	10
TechTC38	1683	143	74	69
Isolet-BC	617	100	50	50

This is due to two reasons. First, we begin with an initialization that gives a valid set of assignments for the class attributed eccentricities. This implies that the *Null Space* for the classified points is non-empty to begin with. Second, any descriptor update chosen from the *Null Space* is a feasible solution that satisfies the distance constraints introduced by the correctly classified points. Moreover, the current descriptor to be updated is also a feasible solution, *i.e.*, in the worst case the *Null Space* collapses to the current descriptor. *Therefore, the learning rate of CSC in non-decreasing*. The radius of the final *Null Space* for each descriptor, monotonically decreases as more points are correctly classified, as can be deduced from Eqn.21. Also, the order of classified samples processed does not affect the final *Null Space* .

A key contribution of our learning technique is the tracking of a large set of feasible solutions as a compact geometric object. From this *Null Space* we pick a set of solutions to learn each misclassified points. From this set, we pick a solution biased towards a simpler discriminant using a stiffness criterion so as to improve upon generalization. The size of the margin, δ, in *ecc-Space* also gives a modicum of control over generalization.

4 EXPERIMENTS

We evaluated the classifier on several real datasets concerning cancer class discovery, epilepsy diagnosis, and recognition of objects and spoken alphabets. We begin with a brief overview of the datasets, and the classifiers used for comparison with Conic Section classifier (CSC). Next, we discuss implementation details of CSC followed by a review of the results.

4.1 Datasets

We conducted experiments on a variety of datasets involving medical diagnosis, object recognition, text-categorization and spoken letter recognition. All the datasets considered have the peculiar nature of being high dimensional and sparse. This is due to the fact that either the collection of samples is prohibitive or that the features obtained for each sample can be too many, especially when there is no clear way to derive a reduced set of meaningful compound features.

The dimensionality and feature size details of the datasets used are listed in Table-1. Epilepsy data [3] consists of the shape deformation between the left and right hippocampi for 44 epilepsy patients. First, we computed the displacement vector field in 3D representing the non-rigid registration that captures the asymmetry between the left and right hippocampi. We found the joint 3D histogram of the (x, y, z) components of the displacement vector field to be a better feature set for classification. We used a $6 \times 6 \times 6$ binning of the histograms in 3D. The task was to categorize the localization of the focus of epilepsy to either the left (LATL) or right temporal lobe (RATL).

The Colon Tumor [20], the Leukemia [4], and the CNS [21] datasets are all gene-expression datasets. The gene-expression scores computation is explained in their respective references. In the Colon Tumor data the task is to discriminate between normal and tumor tissues. The leukemia data consists of features from two different cancer classes, namely acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL). The CNS dataset contains treatment outcomes for central nervous system embryonal tumor on 60 patients, which includes 21 survivors and 39 failures.

From the ETH-80 [6] object dataset, we chose 10 profile views of a dog and a horse. The views are binary images of size 128×128. From the TechTC-38 [22] text-categorization dataset, we classify the documents related to Alabama vs. Michigan localities (id: 38 [22]), given word frequencies as their features. We dropped features (words) that are either very infrequent or too common. The Isolet-BC dataset is a part of the Isolet Spoken Letter Recognition Database [5], [23], which consists of features extracted from speech samples of B and C from 25 speakers.

4.2 Classifiers and Methods

We implemented a faster version of the Linear Fisher Discriminant (LFD) [7] as described in Yu & Yang [24]. This technique exploits the fact that high-dimensional data has singular scatter matrices and discards the subspace that carries no discriminative information. Support Vector Machines (SVM) [25] and Kernel Fisher Discriminants (KFD) [26] broadly represented the non-linear category. Both employ the kernel trick of replacing inner products with Mercer kernels. Among linear classifiers, we chose LFD and linear SVM. We used *libSVM* [27], a C++ implementation of SVM using Sequential Minimal Optimization [28] and our own MATLAB implementation of KFD. Polynomial (PLY) and Radial Basis (RBF) Kernels were considered for SVM and KFD.

We performed *stratified* 10-fold cross validation (CV) [2] in which the samples from each class are randomly split into 10 partitions. Partitions from each class are put into a fold, so that the label distribution of training and testing data is similar. In each run, one fold is withheld for testing while the rest is used for training and the process is repeated 10 times. The average test-error is reported as the generalization error estimate of the classifier. The experimental results are listed in Table-2.

TABLE 2
Classification results for CSC, (Linear & Kernel) Fisher Discriminants and SVM.

Dataset	CSC	LFD	KFD-PLY	KFD-RBF	SVM-PLY	SVM-RBF
Epilepsy	**91.50**	78.00	80.50	85.00	84.50	84.50
Colon Tumor	**88.33**	85.24	78.57	83.57	**88.33**	83.81
Leukemia	95.71	92.86	**98.57**	97.32	97.14	95.71
CNS	71.67	50.00	63.33	**75.00**	65.00	65.00
ETH-Objects	**90.00**	85.00	**90.00**	55.00	85.00	**90.00**
TechTC38	**74.14**	67.05	71.24	71.90	72.10	60.24
Isolet-BC	**95.00**	91.00	94.00	94.00	94.00	94.00

TABLE 3
Parameters for results reported in Table. 2. *Not a parameter.

Dataset	CSC			KFDPLY	KFDRBF	SVMPLY	SVMRBF
	Q	$\delta\%$	Stiffness*	degree	radius	degree	radius
Epilepsy	Means	1	2.9997	4	3	5	.006
Colon Tumor	LFD	.01	2.9955	1	5	2	.5
Leukemia	LFD	1	2.9793	1	70	1	3
CNS	Means	.1	2.9454	5	.07	1	.001
ETH-Objects	LFD	.01	2.9884	3	300	1	2
TechTC	Lin-SVM	1	2.9057	1	500	1	300
Isolet-BC	Lin-SVM	1	3.0	4	.3	1	.07

The best parameters for each classifier were empirically explored using grid search. We searched for the best degree for polynomial kernels from $1 \ldots 5$, beyond which the classifiers tend to learn noise. We computed the largest distance between a sample pair as the base-size for selecting the radius of the RBF kernels. The radius was searched in an exponentially spaced grid between 10^{-3} to 10^5 times the base-size.

4.3 Conic Section Classifier

We implemented CSC in MATLAB. The computationally intensive aspect in the learning is tracking the *Null Space* for the equality constraints. Since this has been reduced to *QR* decomposition of a matrix composed from the sample vectors, we used the MATLAB `qr` routine which is numerically more stable and faster than having to compute the hypersphere intersections separately. Whenever possible, we cached the *QR* decomposition results to avoid recomputations. We have an $O(N^2)$ algorithm to compute the optimal class-eccentricities, that will be used in the first iteration. This gave us the best learning accuracy for the initial configuration of CSC. This could result in a non-linear boundary to start with. Then the stiffness guided choice of updates ensured that we pursue simpler discriminants without decreasing the learning rate. In subsequent iterations, we searched for better eccentricity descriptors in the immediate 10×10 neighborhood intervals of the previous class eccentricities, in *ecc-Space*. This avoids jumping from the current global optima to the next, after learning the foci and directrix descriptors, thereby making the algorithm more stable. We found that the descriptors converged to a local minima typically within 75 iterations of the learning algorithm listed in Figure-3. The parameters of CSC involve the choice of initialization, described in Section 3.5, and the functional margin δ. The δ value is fixed to be at a certain percentage of the extent of attributed eccentricities, $\varepsilon^*(X)$. We searched for the best percentage among $[1\%, .1\%, .01\%]$.

4.4 Classification Results

Classification results are listed in Table-2. Conic Section Classifier (CSC) performed significantly better than the other classifiers for the Epilepsy data. In the gene-expression datasets, CSC was comparable to others with Leukemia and Colon Tumor data, but not with the CNS data which turned out be a tougher classification task. CSC fared slightly better than the other classifiers in text-categorization and spoken letter recognition data. In fact, CSC has consistently performed substantially better than the LFD, as it is desirable by design. It is also interesting to note that none of the SVM classifiers actually beat CSC. This empirically proves that CSC is able to represent more generalizable boundaries than SVM for all the data considered.

The parameters used in the experiments are listed in Table-3. The column related to Q, lists that the normals to directrices were initialized from those due to either linear SVM , or LFD, or the line joining the means of the samples from the same class. The column related to δ denotes the margin as percentage of the range of class attributed eccentricities. We report the average *stiffness* of CSC over 10-folds in each experiment. Note that this is not a parameter of CSC. We included this in the table for comparison with the degree of the polynomial kernels. The stiffness values for all experiments were near its maximum (3.0). The standard deviation of the stiffness was less that 0.02 for all the data, except for CNS ($.08$), and Isolet-BC ($.1$) datasets. Hence, *stiffness* does guide descriptor updates towards yielding simpler discriminants for the same or higher learning accuracy. The best degree for other polynomial kernel classifiers

also turned out to be 1, in several cases. Except for the Leukemia data, CSC performed better than linear SVM and linear Fisher classifiers in all cases.

We performed experiments on a quad core 3.8 GHz 64-bit Linux machine. Each 10-fold CV experiment on gene-expression datasets took under 4 minutes for all the classifiers including CSC. For the other datasets, run times were less than 2 minutes. Since we used a faster variant of LFD [24], it took under a second for all the datasets. The run times for CSC were better than the other classifiers with RBF kernels. However, the search for the optimal model parameters adds a factor of 5 to 10. From the experimental results in Table-2, it is evident that Conic Section classifier out-performed or matched the others in a majority of the datasets.

5 Conclusions

In this paper, we introduced a novel concept class based on conic section descriptors, that can represent highly non-linear discriminant boundaries with merely $O(M)$ parameters, and that subsumes linear discriminants. We provided a tractable supervised learning algorithm in which the set of feasible solutions, called the *Null Space*, for a descriptor update is represented as a compact geometric object. The computation of the *Null Space* related to quadratic equality constraints on class descriptors is reduced to that of a Gram-Schmidt orthogonalization. We introduced a *stiffness* criterion that quantifies the extent of the non-linearity in the discriminant boundary. In each descriptor *Null Space* , we pick solutions that learn misclassified points and yield simpler discriminant boundaries, due to the *stiffness* criterion. Thus *stiffness* enables the classifier to perform model selection in the learning phase. As the learning problem is equivalent to a non-linear optimization problem that is not convex, our method is prone to local minima as well.

We tested the resultant classifier against several state-of-the-art classifiers on many public domain datasets. Our classifier was able to classify tougher datasets better than others in most cases, as validated in Table-2. The classifier in its present form uses axially symmetric conic sections. The concept class, by definition applies to the multi-class case as well. The learning algorithm needs to incorporate equivalent boundary representation in *ecc-Space* , among other aspects. In future work, we intend to extend this technique to multi-class classification, to conic sections that are not necessarily axially symmetric, and explore pursuit of conics in kernel spaces.

Appendix
List of Symbols

N, M	Number of samples and features.
X	A point in $\mathbb{R}^M$.
F	A focus point.
$\langle y_i, X_i \rangle$	A labeled data sample. $y_i \in \{-1, +1\}$
$\{b, Q\}$	Directrix hyperplane descriptors.
C_k	Descriptors of class k: $\{F_k, \{b_k, Q_k\}, e_k\}$
e_k	The scalar valued class eccentricity.
$\varepsilon_k(X)$	The eccentricity function given C_k.
$\mathcal{G}$	$\{X \in \mathbb{R}^M : g(X) \equiv 0\}$ (decision boundary)
$r_k(X)$	Distance to focus : $\|X - F_k\|$
$h_k(X)$	Distance to directrix : $b_k + Q_k^T X$
$S_i(r_i, X_i)$	Hypersphere centered at X_i with radius r_i
shell	Space between two concentric hyperspheres.
$\mathcal{S}^e(r_e, C_e)$	$\cap_{i=1}^{N_c} S_i$ for N_c classified points.
$\mathcal{H}$	A linear subspace in which $\mathcal{S}^e$ lies.
$\mathcal{S}^f$	Toroidal *Null Space* defined in Eqn.22.

Acknowledgments

This research was in part supported by NIH RO1 NS046812 to BCV. The authors would like to thank Jeffrey Ho for his participation in insightful discussions.

References

[1] V. Vapnik, *Statistical Learning Theory*. John Wiley and Sons, New York, 1999.
[2] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *International Joint Conference on Aritificial Intelligence*, 1995, pp. 1137–1145.
[3] N. Vohra, B. C. Vemuri, A. Rangarajan, R. L. Gilmore, S. N. Roper, and C. M. Leonard, "Kernel fisher for shape based classification in epilepsy," *MICCAI*, pp. 436–443, 2002.
[4] T. Golub *et al.*, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, 1999.
[5] R. Cole, Y. Muthusamy, and M. Fanty, "The ISOLET spoken letter database," Oregon Graduate Institute, Tech. Rep. CSE 90-004, 1990.
[6] B. Leibe and B. Schiele, "Analyzing appearance and contour based methods for object categorization," in *Computer Vision and Pattern Recognition*, vol. 2, 2003, pp. 409–415.
[7] R. Fisher, "The use of multiple measurements in taxonomic problems," in *Annals of Eugenics (7)*, 1936, pp. 111–132.
[8] A. Jain and D. Zongker, "Feature selection: Evaluation, application, and small sample performance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 153–158, 1997.
[9] P. Somol, P. Pudil, and J. Kittler, "Fast branch and bound algorithms for optimal feature selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 7, pp. 900–912, 2004.
[10] K.-C. Lee, J. Ho, and D. J. Kriegman, "Acquiring linear subspaces for face recognition under variable lighting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 684–698, 2005.
[11] L. Torresani and K. chih Lee, "Large margin component analysis," in *Advances in Neural Information Processing Systems 19*. Cambridge, MA: MIT Press, 2007, pp. 1385–1392.
[12] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.
[13] A. Banerjee, S. Kodipaka, and B. C. Vemuri, "A conic section classifier and its application to image datasets," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 1, pp. 103–108, 2006.
[14] F. L. Bookstein, "Fitting conic sections to scattered data," *Computer Graphics and Image Processing*, vol. 9, no. 1, pp. 56–71, 1979.

[15] T. Pavlidis, "Curve fitting with conic splines," *ACM Trans. Graph.*, vol. 2, no. 1, pp. 1–31, 1983.
[16] L. Quan, "Conic reconstruction and correspondence from two views," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 2, pp. 151–160, 1996.
[17] G. Dorffner, "A unified framework for MLPs and RBNFs: Introducing conic section function networks," *Cybernetics and Systems*, vol. 25, no. 4, 1994.
[18] D. Hush and C. Scovel, "On the vc dimension of bounded margin classifiers," *Machine Learning*, vol. 45, no. 1, pp. 33–44, 2001.
[19] L. N. Trefethen and D. Bau, *Numerical Linear Algebra*. SIAM: Society for Industrial and Applied Mathematics, 1997.
[20] U. Alon *et al.*, "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," in *Proc. Nat. Acad. Sc . USA*, vol. 96, 1999, pp. 6745–6750.
[21] S. L. Pomeroy *et al.*, "Prediction of central nervous system embryonal tumour outcome based on gene expression," *Nature*, vol. 415, no. 6870, pp. 436–442, 2002.
[22] E. Gabrilovich and S. Markovitch, "Text categorization with many redundant features: using aggressive feature selection to make svms competitive with c4.5," in *ICML '04: Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004.
[23] A. Asuncion and D. Newman, "UCI machine learning repository," 2007.
[24] H. Yu and J. Yang, "A direct lda algorithm for high-dimensional data with application to face recognition," *Pattern Recognition*, vol. 34, no. 12, pp. 2067–2070, 2001.
[25] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.
[26] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller, "Fisher discriminant analysis with kernels," *Neural Networks for Signal Processing IX*, pp. 41–48, 1999.
[27] R.-E. Fan, P.-H. Chen, and C.-J. Lin, "Working set selection using second order information for training support vector machines," *Journal of Machine Learning Research*, vol. 6, pp. 1889–1918, December 2005.
[28] J. C. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT Press, Cambridge, MA, USA, 1999, pp. 185–208.

UF College of Engineering
Department of Computer & Information Science & Engineering
UNIVERSITY *of* FLORIDA

CVGMI

Center for Vision, Graphics and Medical Imaging

http://www.cise.ufl.edu/research/cvgmi/

E331, CSE Building, PO BOX 116120, Gainesville, FL 32611, USA; +1 (352) 392 5770