Feature Extraction and Support-Vector Classification of FLIR Imagery

Shaorong Chang, Balaji Krishnapuram, Paul Runkle, Lawrence Carin, Sandor Der and Nasser Nasrabadi

1Department of Electrical and Computer Engineering
Duke University
Durham, NC 27708-0291

2Army Research Laboratory
Adelphi, MD 20783

Abstract – Algorithms are considered for classifying vehicles based on forward-looking infrared (FLIR) imagery. The image features are designed to focus on target subregions, with these ideally characterized by distinct thermal properties. In one implementation the subregions are determined manually, and in the second we utilize an independent-components analysis (ICA). The features are processed via a support vector machine (SVM), with two distinct approaches used to convert the binary SVM classifier into the multi-target classifier of interest. The algorithms are examined using measured FLIR imagery.

I. Introduction

Classification of targets based on forward-looking infrared (FLIR) imagery is complicated by the fact that the associated imagery is a strong function of the target-sensor orientation (pose) [1-7]. Due to this nonstationarity of FLIR imagery as a function of target-sensor orientation, it is difficult to design a single classifier for a given target, and therefore we define target classes. A target class represents a set of target-sensor orientations over which the associated FLIR imagery is approximately stationary. Each target is characterized by multiple classes. Moreover, the imagery associated with a given target class can vary significantly, representative of variation in the target thermal properties. For example, the FLIR image of a target, at a fixed target-sensor pose can vary
significantly depending on the target history (e.g. whether the engine is on, how long the engine has been on/off, target motion, etc.).

One of the principal challenges in processing FLIR imagery involves designating a feature vector, from which one achieves data compression. The objective is to retain as much of the discriminatory information as possible from the original image, while simultaneously reducing the degrees of freedom. In the work reported here the features are tied to particular subregions on the target, each of which ideally has distinctive thermal properties. For each such subregion a set of templates are designed, based on a Karhunen-Loeve (KL) transform [8] of the training data (as in a principal-components analysis [9]). The dimensionality of the feature vector is tied to the number of subregions considered.

We investigate two techniques for designation of the subregions. In one approach the subregions are designated explicitly, based on a manual analysis of the associated FLIR imagery. The extraction of a given target subregion can be viewed as implementing a simple weighting filter: those parts of the image associated with a desired subregion are weighted by one, with the remaining portions of the imagery weighted by zero. Each subregion has an associated weighting filter. As an alternative to this manual approach, we have also designed weighting filters based on an independent-components analysis (ICA) [10-12] of the available training data. This process emphasizes those portions of the FLIR image for which the associated ICA weighting filter is large, while de-emphasizing the other regions. In this case the weighting filter is no longer binary, and the number of ICA components used is analogous to the number of subregions considered in the manual approach.

Having constituted a feature vector, the classification is performed via a support vector machine (SVM), such being the subject of considerable interest recently in the classification community [13-18]. An SVM is typically applied for binary classification problems, while here we are interested in classifying multiple target classes. We consider two techniques for implementing the SVM-based classification of $N_c$ classes ($N_c>2$). In
one we build $N_c$ SVMs, one for each class. Classifier $m$ is used to specify a decision surface in feature space between class $m$ and all the other $N_c-1$ classes. As we discuss in Sec. IV, this approach has limitations, principally because the size of the associated feature vector grows linearly with $N_c$. We therefore also consider an alternative approach, in which we design $N_c(N_c-1)/2$ simpler SVM classifiers, for which the feature-vector dimensionality does not grow with increasing $N_c$. While the number of such SVM classifiers grows in a quadratic fashion with $N_c$, each associated SVM is considerably simpler than in the former approach. The performance of these two approaches is examined based on the processing of measured FLIR imagery.

The remainder of the text is organized as follows. In Sec. II we discuss the manual technique by which the target subregions are used to design a feature vector, with the automated ICA-based approach discussed in Sec. III. The SVM is discussed in Sec. IV, and example results are presented in Sec. V. The research is summarized and conclusions are discussed in Sec. VI.

II. Manual Feature Extraction

A. Feature templates

As indicated above, a target class is constituted by a set of target-sensor orientations over which the associated FLIR imagery is relatively stationary statistically. Assume $N$ targets, each composed of $K$ classes, for a total of $N_c=KN$ classes. We also assume that the images associated with class $m$ are composed of $P_{m-1}$ subcomponents (subregions), with this made more explicit below. We derive representative templates for each of the $P_{m-1}$ target subcomponents, as well as a separate template representative of the overall target image. We therefore employ a total of $P_m$ template sets for a given target class, with $P_m$ in general class dependent (in the examples presented in Sec. V, $P_m$ is the same for all target classes).
As an example, we have $P_m=4$ sets of templates for the image in Fig. 1. Each subcomponent corresponds to a localized region of the overall image, with such linked to particular components of the target. In particular, each subcomponent corresponds to a different portion of the vehicle (e.g., the front cabin, wheels, etc.). A subcomponent is expected to have associated thermal characteristics, with such also related to the thermal properties of adjoining subcomponents. There is flexibility in the choice of the parameter $P_m$ as well as in the choice of the specific target subcomponents. The objective is to delineate particular subregions on the target that have characteristic thermal properties (for example the tires). Knowledge of the underlying thermal physics can be utilized to define such relatively isolated subcomponents on the vehicle, although here the decomposition is performed in a somewhat *ad hoc* manner, based on known target subregions. An automated approach is discussed in Sec. III.

**B. EXM filter formulation**

From the training data associated with class $m$, assume we have $N_m$ realizations of each of the $P_m$ regions. Using this data, we form a set of $N_m$ Wiener filters [19,20] for each of the $P_m$ class-dependent regions (a distinct filter designed from each of the $N_m$ training images). These filters have been termed expansion-matching (EXM) filters [19,20], where here $\Omega_n^{m,p}$ represents the EXM filter belonging to class $m$, region $p$, as derived from the $n^{th}$ training image. The utility of an EXM filter, *vis-à-vis* a matched filter, is discussed in [19,20]. We have found improved performance using EXM filters (relative to matched filters), because the EXM filters emphasize the edges in the templates [19,20]. The $N_m$ EXM filters associated with region $p$ for class $m$ are denoted as $\Sigma_{m,p}$, where

$$\Sigma_{m,p} = [\Omega_1^{m,p}, \Omega_2^{m,p}, ..., \Omega_{N_m}^{m,p}], \quad 1 \leq m \leq M, 1 \leq p \leq P_m$$

(1)

To reduce the number of templates associated with region $p$ and class $m$, we perform a Karhunen-Loeve (KL) transform [8]. In particular, we compute the eigenvectors and eigenvalues of the matrix
\[ [R_{\text{test}}] = \frac{1}{N_m} \sum_{i=1}^{N_m} \Omega_i^{m,p} \Omega_i^{m,p^T} \]  

(2)

where superscript \( T \) represents transpose. The KL transform produces an orthonormal set of basis functions \( \Phi_i^{m,p} \) for the given set of EXM filters. Note that we have chosen to perform the KL transform after designing the \( N_m \) EXM filters, since the resulting set of eigenvectors are orthonormal.

The eigenvectors are arranged in descending order of eigenvalues \( \eta_i^{m,p} \), and we retain the eigenvectors associated with the largest \( N_{eig} \) eigenvalues [8]. For the imagery discussed in Sec. V, we have typically employed \( N_{eig} \) to be on the order of 10% of \( N_m \). The set of \( N_{eig} \) eigen templates corresponding to class \( m \) and subregion \( p \) is represented as \( E_{m,p} \)

\[
E_{m,p} = [\Phi_1^{m,p}, \Phi_2^{m,p}, ..., \Phi_{N_{eig}}^{m,p}], \quad 1 \leq m \leq M, \ 1 \leq p \leq P_m
\]  

(3)

C. Feature vector

Assume we are to classify an image \( I \) as one of \( N_c \) classes. When examining whether this image is associated with class \( m \), we compute the \( P_m \)-dimensional feature vector

\[
C_m = [\text{corr}_1^{m}, \text{corr}_2^{m}, ..., \text{corr}_{P_m}^{m}] \]  

(4)

where

\[
\text{corr}_p^{m} = \max_{\text{neighborhood } p} \left[ \sum_{j=1}^{N_m} \frac{\langle I, \Phi_j^{m,p} \rangle}{\langle I^p, I^p \rangle^{\frac{1}{2}} \langle \Phi_j^{m,p}, \Phi_j^{m,p} \rangle^{\frac{1}{2}}} \right] \]  

(5)

\( \langle I^p, I^p \rangle \) is the energy in \( I \) over the support of the filters associated with region \( p \).

Each of the \( P_m \) regions associated with class \( m \) corresponds to a physical location on the target, with known relative positioning. Referring to (5), when computing the correlation for subregion \( p \) we only search for the maximum correlation over a localized
portion of the image consistent with that region. To define the appropriate portions of the image for definition of the $p$-dependent region, it is important to align the image appropriately. In this context, we first perform a correlation with eigen EXM filters associated with target component $p=1$, corresponding to the overall target shape (see Fig. 1). After finding the maximum correlation with this set of eigen EXM filters, the target image is repositioned in the center of the domain. After this alignment is performed, the corresponding subcomponent regions are defined, in which we perform searches for the maximum correlations for subregions regions $p=2$ through $p=P_m$.

III. Automatic Template Design for Feature Extraction: Independent Components

The procedure developed in Sec. II results in features tied to physical portions of the vehicle, with the goal of isolating regions of the vehicle with characteristic thermal properties. There is significant human intervention required, to determine the number of regions $P_m$, as well as in delineating the subregions themselves. It is possible to tie this decomposition to the underlying, fundamental thermodynamics of the vehicle, although this has not been attempted here. In this section we consider an alternative approach, based on an independent-components analysis (ICA) [10-12].

Assume that a given vehicle is characterized by a finite set of heat sources. For example, the engine, the exhaust pipe, tires heated by friction, etc. The temperature of any given exterior portion of the vehicle (seen via a FLIR imager) can be approximated as a linear mixture of the heat produced by each of the aforementioned sources, with the mixing dictated by the thermodynamics by which heat travels from a given source to a particular exterior region on the target. Under the assumption of linearity, let the matrix $M_{qi}$ represent the temperature at point $q$ on the vehicle exterior, when source $i$ has unit amplitude (and all other sources are off). When all sources are on, the matrix $M_{qi}$ yields the overall target temperature at all pixel points $q$. In particular, if we assume a FLIR image is characterized by $Q$ pixels, and that class $m$ is composed of $I_m$ sources, then the matrix components $M_{qi}$ constitute a $Q \times I_m$ matrix $M$. Let $q$ represent a $Q \times 1$ vector denoting the FLIR-image pixel values arranged in a vector (here we assume
monochromatic imagery), and \( \mathbf{i} \) represents an \( I_m \times 1 \) vector denoting the temperature of each of the \( I_m \) sources. Under the aforementioned assumptions, we have \( \mathbf{q} = \mathbf{M} \mathbf{i} \), where the vector \( \mathbf{i} \) constitutes the random temperature of the sources, with each realization of \( \mathbf{i} \) reflecting the random target history. The statistical variation of \( \mathbf{i} \) constitutes the statistical variation of the FLIR imagery associated with a given target class (manifested in the random FLIR imagery \( \mathbf{q} \)).

The above discussion was based on the assumption that distinct heat sources can be identified and that the target thermal properties are linear. Both of these assumptions are expected to be reasonable in many scenarios. We now add an additional assumption, the validity of which may deserve further study (nevertheless, we have realized quality results when it has been invoked, as shown in Sec. V). In particular, we assume that the components of \( \mathbf{i} \) are statistically independent, implying that the heat sources act independently. When the tires heat due to movement (friction), the engine and exhaust pipe will also likely be warm. However, when the vehicle stops, the engine can remain on (stay warm), while the tires cool. Moreover, after the engine is turned off, the properties characteristic of engine, tire and exhaust-pipe cooling, for example, are likely to be unrelated.

Under the assumption that the components of \( \mathbf{i} \) are statistically independent, the matrix equation \( \mathbf{q} = \mathbf{M} \mathbf{i} \) is ideally suited for an independent-components analysis (ICA) [10-12]. In particular, given an ensemble of statistical FLIR imagery \( \mathbf{q} \) (here representative of the FLIR pixel values, for a given target class), and for a chosen number of sources \( I_m \), the ICA algorithm determines that (deterministic) matrix \( \mathbf{M} \) that most makes the components of the associated \( \mathbf{i} \) as statistically independent as possible. Each column of \( \mathbf{M} \), \( \{M_{q1}, q=1, 2, \ldots Q\} \), represents the FLIR pixel values on the target when source \( i \) has unit amplitude and all other sources are set to zero. Moreover, given FLIR imagery \( \mathbf{q} \), \( \mathbf{i} = \mathbf{M}^{-1} \mathbf{q} \) yields the estimated strength (temperature) of each source, where \( \mathbf{M}^{-1} \) is implemented as a pseudo inverse. The ICA algorithm has been discussed in many papers [10-12], and therefore we do not present algorithmic details here.
The ICA procedure employed here is depicted in Fig. 2. The matrix $M$ and sources $i$ are unknown. However, using multiple training images $q$, and assuming that the elements of $i$ are statistically independent, the ICA algorithm solves for $M^{-1}$ [10-12], from which we compute the desired matrix $M$. In Fig. 2 we consider $I_m=4$ independent components, and the column labeled “training FLIR images” represent only four of the 64 training FLIR images used by the ICA algorithm in this example. The four “learned filters” represent the four rows of $M^{-1}$, and the “learned basis images” represent the four columns of $M$. For illustrative purposes, the 64 training images cover a relatively limited $10^\circ$ range of target-sensor orientations ($\pm 5^\circ$ about broadside), and therefore the ICA components are relatively “clean”, i.e. the columns of $M$ look like the target shape. As we discuss in Sec. V, each target class considered in the FLIR classification problem encompass target-sensor orientations of significantly greater extent than $10^\circ$, and therefore the learned basis images are no longer as clean as those depicted in Fig. 2.

Assume training data associated with class $m$, for which we assume $I_m$ sources. Let $g_{im}$ represent the $i$th column of $M$, normalized for unit energy ($g_{im}^T g_{im}=1$). We now consider developing a feature vector associated with class $m$. When processing a given image under test, $I$, we weight $I$ by the associated elements of $g_{im}$ (recall that the elements of $g_{im}$ correspond to pixels on the FLIR target image). In particular, for the $I_m$ ICA components, we generate $I_m$ new matrices $\hat{I}_{im}$, where each element of $\hat{I}_{im}$ is a product of the respective elements in $I$ and $g_{im}$. Large elements in $g_{im}$ correspond to regions in $I$ that are emphasized in the associated $\hat{I}_{im}$, with the converse true for small elements in $g_{im}$.

The algorithm now proceeds similarly to that discussed in Sec. II. In particular, given training data, we design $I_m$ ICA-based weighting images $g_{im}$ for class $m$. All $N_m$ training images are then weighted with the $g_{im}$, yielding $I_m N_m$ images. Each of these images is used to design an associated EXM filter (see Sec. II). The $N_m$ EXM filters associated with the $i$th $g_{im}$ are then compressed into $N_{\text{eig}}$ images ($N_{\text{eig}}<<N_m$), via the KLT algorithm, as in Sec. II. If necessary, a preliminary alignment of the imagery can be performed, in the manner discussed in Sec. II.
When generating a feature vector associated with class $m$ for test image $I$, we first weight $I$ by the associated set of images $g_{im}$ (yielding $I_m$ new images $I_i$, with $i=1, 2, \ldots, I_m$). Inner products are then taken between $I_i$ and each of the associated set of KLT-generated eigen templates (from the above paragraph, each $g_{im}$ has an associated set of eigen-templates). The feature vector associated with ICA component $g_{im}$ is the sum of these inner products, analogous to (5). For $I_m$ ICA components we therefore generate a feature vector $C_m = [\text{corr}_1^m, \text{corr}_2^m, \ldots, \text{corr}_n^m]$, as in (4).

From the above discussion, we now note the principal distinction between the procedure in Sec. II and that based on ICA. In Sec. II we identified subregions on the target, with these effected by weighting the image under test by a binary filter (the weighting filter is one in subregions of interest, and zero elsewhere). This same weighting procedure is applied to the training data to design the KLT-based templates, from which the features are defined. In the ICA-based procedure the templates are no longer binary, but rather are based on the columns of the ICA matrix $M$ (i.e. $g_{im}$).

Recall that $g_{im}$ reflects the FLIR imagery of target class $m$, when source $i$ has unit amplitude (and all other sources off). Within ICA, the spatial position of the source is not made explicit. Nevertheless, within the context of the FLIR problem, as indicated above, it is anticipated that sources are likely tied to physical components of the vehicle (such as the engine, tires, etc.). For a given such source $i$, it is anticipated that the associated heat distribution about the target surface will be concentrated in the vicinity of the source. For example, for a source $i$ corresponding to the tires, the associated FLIR signature $g_{im}$ is likely strongest in the vicinity of the tires, and weak elsewhere. In this sense each $g_{im}$ denotes target subregions of thermal importance, such subregions having motivated the manual decomposition discussed in Sec. II. Using ICA, the target subregions are determined automatically, based on the training data, and the associated weighting functions $g_{im}$ are no longer binary. The principal limitation of this procedure is that one must a priori choose the number of ICA components $I_m$; i.e., this parameter is not
selected automatically via ICA [10-12]. However, based on training data, one can estimate a reasonable value for $I_m$.

IV. Support-Vector Classifier

Support vector machines (SVMs) have received significant attention recently in the context of classification problems [13-18]. We provide below a brief summary, in which we discuss the particular manner in which SVMs have been utilized in this work. In particular, an SVM is a binary classifier. The $k$th example of a feature vector, $V_k$, is identified by the associated label $y_k$, where $y_k = \pm 1$ represents the binary class label. We consequently have the set of data $(V_k, y_k)$. Assume we are given a set of labeled training observations $T = \{(V_1, y_1), (V_2, y_2), \ldots (V_L, y_L)\}$ where labels $y_i \in \{-1,+1\}$ and $V_i \in O$, where $O$ is the space of the observed feature vectors.

A. SVM basics

Let $\chi_i = \varphi(V_i)$ be the feature vector obtained after a general feature mapping $\varphi$ (the dimensionality of the vectors $V_i$ and $\chi_i$ are in general different, and often the dimensionality of $\chi_i$ is much larger than that of $V_i$, even possibly of infinite dimension [13]). If $w$ represents the vector normal to a hyperplane classifier in the space $X$ of $\chi_i$, and $b$ is the distance of the origin of the coordinate axis measured from the hyperplane, then we would like to have a margin of separation of at least $1/\|w\|$ from the hyperplane to the nearest data point among $\chi_i$

$$y_i[<w, \chi_i> + b] \geq 1, \forall i=1, \ldots, L$$

The expression $<w, \chi_i>$ represents an inner product between the vectors $w$ and $\chi_i$.

We note that this can always be achieved by scaling the vector $w$ appropriately if the two classes are linearly separable in their feature space. It may be easily seen that in the event of such a scaling of $w$, $1/\|w\|$ is a measure of separation between the two classes due to the hyperplane [13]. However, in practice there may not be a hyperplane that can
separate the two classes completely, especially in the presence of noise. Consequently, we permit some classification errors to exist in some of the labels declared by the SVM. In other words, 

$$y_i[<w, x>i + b] \geq 1 - \xi_i, \quad \forall \ i = 1, \ldots, L$$

(7)

and 

$$\xi_i \geq 0 \quad \forall \ i = 1, \ldots, L.$$ 

(8)

Thus a classifier \( \{w, b\} \) which generalizes well can be found by controlling both the classification margin (via \( ||w|| \)) and the number of training errors. This is achieved by minimizing the objective function

$$\pi(w, \xi) = \frac{||w||^2}{2} + \lambda \sum_{i=1}^{L} \xi_i$$

(9)

subject to constraints (7) and (8) for some constant \( \lambda > 0 \) determining the tradeoff between training errors and generalization capability. It is useful to observe that (9) is a quadratic programming problem with linear constraints. It can be solved using the method of Lagrange multipliers, and hence reduced to its dual formulation as illustrated in [13]:

$$\text{Max} \quad W(\alpha) = \sum_{i=1}^{L} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{L} \alpha_i \alpha_j y_i y_j <x_i, x_j>$$

(10)

Subject to \( \lambda \geq \alpha_i \geq 0, \ i = 1, \ldots, L \), and \( \sum_{i=1}^{L} \alpha_i y_i = 0 \)

(11)

We have used the SMO algorithm [16] to solve this constrained optimization problem to identify the optimal \( w, b \) of the hyperplane. Then the SVM classifier is obtained using

$$w = \sum_{i=1}^{L} y_i \alpha_i x_i, \quad \text{such that} \quad$$

$$f(x) = \text{sign} \left( \sum_{i=1}^{L} y_i \alpha_i <x, x_i> + b \right)$$

(12)

$$f(V) = \text{sign} \left( \sum_{i=1}^{L} y_i \alpha_i <\varphi(V), \varphi(V_i)> + b \right)$$

(13)
Those \( V_i \) for which \( \alpha_i > 0 \) are called support vectors. Thus, only the support vectors affect the final decision function of the SVM.

The function \( K(V, V_i) = <\phi(V), \phi(V_i)> \) is a nonlinear measure of similarity between the observations \( V \) and \( V_i \) and is called the kernel function. We note that if the kernel is given, then we do not need to find \( \phi(V) \) explicitly. This property enables us to even use an infinite-dimensional feature space for the purpose of classification [i.e., the vector \( \phi(V) \) may be of infinite dimension, since it is not computed explicitly, rather we only require the kernel \( K(V, V_i) \)]. The explicit minimization of \( ||w|| \) along with the training error allows us to surmount the “curse of dimensionality”, to achieve remarkably good classification performance [13].

We also note that regularization theory has been traditionally used in order to improve generalization performance in pattern recognition problems, naturally raising the question of the relative merits of the SVM scheme used in the paper. In this context Smola and Scholkopf [16] have described the general conditions for an equivalence relationship between regularization theory and the SVM technique described here. More specifically, they prove that the Green's functions associated with regularization operators are suitable kernels for SVMs with equivalent regularization properties.

This result shows why mapping to very high-dimensional feature spaces can still provide good results. However, the advantages of the SVM technique over regularization networks are due to the solution sparsity (very few of the data points are support vectors having non-zeros coefficients \( \alpha_i \)), and the consequent computational advantages thereof.

**B. Multi-class classification**

As indicated above, the SVM is designed as a binary classifier (\( y_k = \pm 1 \)), while here we are interested in classifying multiple target classes. Assume that we are interested in \( N_c \) classes (\( N_c > 2 \)). We have implemented the multi-class SVM classifier in two distinct ways.
In the first approach we design $N_c$ SVM classifiers. In the context of the feature vector discussed in Sec. II, assume that all $N_c$ classes are characterized by the same number of subregions $P_m$. Moreover, in the context of the ICA-based feature vector discussed in Sec. III, assume that all classes are characterized by the same number of thermal sources $I_m$. For a given image under test, the feature vector (4) is formed for all $N_c$ classes (using the appropriate set of templates discussed in Secs. II and III), and the $N_c$ feature vectors are concatenated to form a single feature vector of dimension $P_mN_c$ or $I_mN_c$, using the techniques in Secs. II and III, respectively. To give a sense of the size of this cumulative feature vector, for the results presented in Sec. V, $N_c=40$ and $P_m$ and $I_m$ are typically less than five. Using the notation above, the concatenated feature vector is denoted $V$.

For such feature vectors, of dimension $P_mN_c$ or $I_mN_c$, we design $N_c$ SVM classifiers. For the classifier associated with class $m$, the “+1” case corresponds to class $m$ and the “-1” case corresponds to all other classes. The SVM associated with class $m$ seeks a decision surface in feature space that separates features $V$ associated with class $m$ from features associated with all other classes. Using the notation from (13), let $f_m(V)$ represent the SVM classifier associated with class $m$. A given feature vector $V$ is associated with that target for which the corresponding $f_m(V)$ is largest [13] (i.e., a given feature vector is associated with class $k$ of $f_k(V) > f_m(V)$ for all $m \neq k$).

As has been discussed in [16,17], the above procedure yields final decision boundaries that are often different than those found in the original set of SVMs $f_m(V)$. Moreover, as the number of classes $N_c$ increases, the size of the associated feature vector $V$ (here $P_mN_c$ or $I_mN_c$) increases. We therefore also consider the following alternative approach to implementing a multi-class SVM, based on $N_c(N_c-1)/2$ simpler classifiers. In particular, consider target classes $k$ and $m$. In this case the cumulative feature vector is only a concatenation of the feature vectors associated with classes $k$ and $m$ (see Secs. II and III). In particular, using the technique in Sec. II the feature vector $V$ is of dimension $2P_m$, and using the technique in Sec. III it is of dimension $2I_m$. An SVM is designed for
the two classes \( k \) and \( m \), where “+1” corresponds to class \( k \), and “-1” corresponds to class \( m \) (for example). In this manner we build \( N_c(N_c-1)/2 \) such binary classifiers. Each class is compared to all \( N_c-1 \) other classes, and we declare the image under test to be associated with that class to which such a binary classification occurs most frequently (\( i.e. \), via simple “voting”). Other similar procedures can be used to make the final classification, based on the \( N_c(N_c-1)/2 \) binary classifiers.

The disadvantage of the second approach discussed above is that the number of classifiers, \( N_c(N_c-1)/2 \), grows in a quadratic fashion with the number of classes \( N_c \). However, the feature-space dimension is much smaller than that in the original classifier. In practice we have found that the two approaches require comparable computational effort to test, but that the second approach is much faster when performing training (because of the reduce feature-vector dimension). We also note that each of the \( N_c(N_c-1)/2 \) binary classifiers used in the second approach retain the decision boundaries employed in the original set of SVMs [16,17].

V. Example Results

A. Problem description

We consider classifying FLIR images of ten distinct vehicles (see [7] for a further discussion of the data). The data contains a total of 13,861 training images (approximately 1,400 images per target). The testing data exists for only five of the targets, consisting of a total of 3,485 images. This represents the realistic case in which one may design classifiers for a large number of targets, but only see a subset of such in practice. The training data corresponds to the targets in a relatively clutter-free environment, while the testing data is cluttered. Typical examples of the FLIR imagery are shown in Fig. 3.

The training images were obtained at 5° intervals around the targets from a scaled 2-km viewing distance. We define four classes for each target, each class representative
of a contiguous set of target-sensor orientations over which the FLIR imagery is approximately stationary. Let $0^\circ$ be defined as looking at the front end of a vehicle. As shown in Fig. 4, class 1 of a target is defined as FLIR images of the front of a vehicle (angle $0$-$30^\circ$ and $330$-$360^\circ$), class 2 represents the left side (angle $35$-$145^\circ$), class 3 represents the rear (angle $150$-$210^\circ$), and class 4 represents the right side (angle $215$-$325^\circ$). Since there are four classes for each target and ten targets in total, there are $N_c=40$ classes.

### B. Eigen-filter selection

First addressing the feature vector discussed in Sec. II, four sets of EXM filters are developed for each image class ($P_m=4$): one set of EXM filters for the entire image, with the other three for target subregions, as in Fig. 1. We perform a KL transform [8], and upon setting $N_{eig}$ such that more than 90% of the energy in the original set of filters is extracted by the eigen-templates, on average $N_{eig}=46$ for the eigen-template set corresponding to the entire images and on average $N_{eig}=20$ for the eigen-templates associated with the three subimages. When more energy is maintained (i.e. when more eigen-templates are employed), better classification performance is obtained, but more computer memory and time are required in algorithm training and testing. This trade-off is shown explicitly in Fig. 5. In particular, in Fig. 5 we plot classification performance using the feature vectors discussed in Sec. II, as a function of the relative percentage of eigen-templates used. Results are shown when algorithm testing is performed on the testing and training data, and we see that the performance stops improving significantly after retaining approximately 10% of the principal eigen-templates. In Fig. 5 we have used the first type of multi-class SVM discussed in Sec. IVB, and results are plotted as the average correct classification of the target identity. More specifically, in these results a target is deemed correctly classified even if the wrong particular target class is assigned (of four, for a given target). The significant deterioration in performance seen when the testing and training data are distinct (Fig. 5) is reflective of the relative complexity of the testing data, as indicated in Fig. 3 [7].
In the following results we choose $N_{eig}$ such that 90% energy is maintained (corresponding to using around the top 7% eigen-filters, as presented in Fig. 5).

C. Templates design

We have sought to designate target subregions by dividing the images into several physically meaningful subimages (as shown in Fig. 1). This is intuitively straightforward. However, for complex images it is difficult to decide how many templates should be used to obtain quality classification performance. We first tried to use only the “coarse template” representative of the overall images (i.e. $P_m=1$, and no subregions were defined). This corresponds to a traditional principal-components-like [9] analysis. However, the average recognition rate for this case is only 65.4% (quantified on testing data, in the manner discussed in Fig. 5). If we add three additional subregions ($P_m=4$), as indicated in Fig. 1, the average recognition rate increases to 77.2% (see Fig. 5). The associated confusion matrix is shown in Table 1 (see [7] for a further discussion of the data and targets). For these data we found that $P_m=4$ yielded best results. However, as indicated in Sec. III, actually defining the subregions takes particular human intervention and care, thus motivating the automated ICA-based procedure discussed in Sec. III, the results from which we now discuss.

D. ICA continuous templates design

The ICA procedure defines a set of images $g_{im}$ with which a given FLIR is weighted (see Sec. III). In Figs. 6 and 7 we present example $g_{im}$ for $I_m=2$ and $I_m=3$, respectively (all for the same example target class). Within ICA one must choose the number of ICA components $I_m$, and we investigate classification performance using different numbers of components. Figure 8 summarizes the average correct classification rate as a function of $I_m$. We note from Fig. 8 that the classification performance is relatively consistent as a function of $I_m$, with performance better than that found using the technique of Sec. II (in particular, as indicated above, for $P_m=4$ the average correct classification is 77.2%). Remarkably, when we use only one ICA template per class, the
average recognition rate rises to 83.5%. The associated confusion matrix is shown in Table 2.

E. SVM design

The results discussed above were obtained by designing one SVM per class [i.e., “one per class” (OPC)]. Moreover, we considered several different kernel functions [\( K(V, V_i) = \langle \phi(V), \phi(V_i) \rangle \) from (13)], including radial basis function and polynomial kernels [13]. We found that all kernels yielded comparable results, and that the simple linear kernel yielded performance similar to the nonlinear kernels. However, as discussed further below, the different kernels required different numbers of support vectors. The fact that the linear kernel performed well implies that a simple hyperplane (in the original feature space) yields adequate results for this data and for the selected features. All results presented here are based on a linear kernel.

Based on the results in Fig. 8, for the pair-wise classifier (the second classifier discussed in Sec. IVB) we choose \( I_m=1 \), this yielding simple two-dimensional feature vectors that are easily visualized. As discussed in Sec. IVB, for each pair of classes, we design a binary classifier. In this case, since there are \( N_c=40 \) classes in total, we have to train 780 binary classifiers. However, recall from Sec. IV that the increased number of classifiers is compensated for significantly by the reduction in the feature-vector size. In particular, for \( I_m=1 \), the OPC approach operates on 40-dimensional while the PWC operates on two-dimensional feature vector. It is also important to note that each PWC SVM need only employ training data from the two classes in question, while each OPC must use training data from all classes at once. However, for the OPC support vector machine, we only need train 40 classifiers instead of 780 classifiers in PWC SVM training. For \( I_m=1 \), we achieved 81.3% correct classification with the PWC algorithm (for both linear and nonlinear kernels), while as indicated above the OPC yielded 83.5%. These results are likely indistinguishable, given the size of the training and testing set. Note, as indicated above, the training phase is markedly faster with PWC \textit{vis-à-vis} the
OPC (the computational effort is comparable for PWC and OPC testing, although the former was faster).

As indicated in the previous discussion, the linear SVM kernel (representative of a hyperplane) yields results comparable to that of the nonlinear kernels. For the PWC with $I_m=1$, the small size of the feature vector (two-dimensional) affords the opportunity for simple visualization of this phenomenon. In Fig. 9 we plot the feature vectors associated with example two-class data. In Fig. 9(a) we show the decision boundary using a linear SVM kernel (a simple inner product in the original feature space) and in Fig. 9(b) we show the results from a polynomial kernel. Although the decision boundaries are similar, the number of required support vectors (circled in green) is much smaller for the nonlinear polynomial kernel. For the data considered we found that the linear-kernel SVM used approximately 20% of the training features as support vectors, while the polynomial kernel used less than 10%.

**F. Comparison with other algorithms**

In this paper we have concentrated on feature selection and on an SVM classifier. It is of interest to consider algorithm performance to that of previous approaches. This same data set has been investigated by other approaches, and the results presented here reflect the best observed to date. For example, in [7] the authors considered classification based on linear-vector quantization (LVQ), with which optimal average classification performance was approximately 75% (when the algorithm is tested on the testing data).

**VI. Conclusions**

We have addressed the problem of target classification based on the processing of forward-looking infrared (FLIR) imagery. A principal focus has been directed toward defining an appropriate feature set, based on subregions on the target. This has been motivated by the expectation that different portions of the vehicle are likely characterized by distinct thermal properties. In one approach the subregions were effected manually,
requiring careful human design. In the second technique the subregions were defined more loosely, within the context of an independent-components analysis (ICA) [10-12]. The size of the feature vector is tied to the number of explicit target subregions or ICA components. The algorithm performance has been examined based on measured FLIR imagery [7], with results comparing favorably to previous classification approaches [7].

Once the feature vector was constructed, the classification was performed via a support-vector machine (SVM) [13-18]. The SVM is traditionally applied to binary classification, and we have considered two procedures for performing multi-class classification. In one approach we design a single classifier for each class, designed to separate that target class from all others. In the second approach we implement a large set of SVMs, considering two classes at a time. The latter approach has the advantage of employing a much smaller feature vector, this compensating for the large number of (simpler) classifiers required. On the FLIR imagery considered here we found that the two approaches yielded comparable classification performance, while the two-at-a-time approach is considerably more efficient in the training phase (due to the significantly reduced associated feature vector). Although the SVM is applicable to nonlinear kernels, we found, for the data considered, that a linear kernel (i.e., a simple hyperplane) yielded results comparable to that of more-sophisticated nonlinear kernels [13]. However, the nonlinear kernels required fewer support vectors.

There are several issues of interest for future research. While the measured data set was relatively large (and the training and testing data distinct), the observed phenomena should be considered on a larger data set. Of particular interest concerns the optimal choice of kernel functions for FLIR imagery, and an examination of the robustness of a given selected kernel. The data considered was monochromatic, and it is therefore also of interest to extend the techniques developed here to the case of hyperspectral imagery (multiple wavelengths).
References


Figure Captions

Figure 1. Decomposition of an example FLIR image into subregions. Region 1 corresponds to the overall target, while here regions 2-4 correspond to target subregions (e.g. the tires).

Figure 2. Summary of the procedure used in the ICA-based FLIR analysis. The number of sources $I_m$ are assumed known, but the sources (denoted by the vector $i$ and the associated basis images are to be determined. Using an ensemble of FLIR images, each denoted by the vector $q$, and assuming that the sources are statistically independent, ICA yields the matrix $M^{-1}$, where $q=Mi$. The columns of $M$ represent the desired basis images. Here we consider $I_m=4$ components (and only show four representative images $q$ from the associated training set). The ICA components are computed using FLIR imagery over the angular extent 85°-95° (not from all of class 2, indicated in Fig. 4).

Figure 3. Example measured FLIR imagery (side view). The first two rows correspond to example training data, from vehicles one through ten, and the bottom row represents example testing imagery.

Figure 4. Decomposition of a target into classes, each class corresponding to a set of target-sensor orientations for which the associated FLIR imagery is statistically stationary. Position 0° corresponds to the vehicle front.

Figure 5. Classification performance, using $P_m=4$ subregions per class (see Fig. 1). A set of eigen-templates are designed for each subregion, and here classification results are presented as a function of the percentage of eigen-filters employed. Results are shown for testing on the training data, and for testing on a distinct set of data (test data).

Figure 6. Example images $g_{im}$ associated with taking $I_m=2$ independent components, within an independent-components analysis (ICA). We consider class 2 of the same target considered in Fig. 2. The example FLIR image from this class (left) indicates that the imagery of a target is often complex, as opposed to the “clean” example images in Fig. 2.

Figure 7. As in Fig. 6, using $I_m=3$ independent components.
Figure 8. Classification of the SVM classifier, as a function of the number of ICA terms used to denote the target “subregions”. In these computations we employed 40 SVM classifiers (one for each class).

Figure 9. Example distribution of two-dimensional features associated with using \( I_m=1 \) ICA component per target class. The features correspond to two example target classes. The support vectors are circled and the SVM decision boundary is denoted. (a) Linear SVM kernel, (b) polynomial SVM kernel.

Table Captions

Table 1. Confusion matrix for the SVM classifier based on using \( P_m=4 \) manually delineated target subregions (see Sec. II). Classifiers are designed for ten vehicles, and testing data is available for a subset of five vehicles.

Table 2. As in Table 1, but now the target subregions are effected via ICA (see Sec. III), and here \( I_m=1 \) ICA components are utilized.
Figure 2
Figure 4
Figure 6
Figure 7

Component 1  Component 2  Component 3
Figure 9(a)
Figure 9(b)
<table>
<thead>
<tr>
<th></th>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
<th>V6</th>
<th>V7</th>
<th>V8</th>
<th>V9</th>
<th>V10</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>72.6</td>
<td>6.2</td>
<td>2.2</td>
<td>0.7</td>
<td>2.0</td>
<td>0.7</td>
<td>4.3</td>
<td>1.4</td>
<td>8.3</td>
<td>1.5</td>
</tr>
<tr>
<td>V2</td>
<td>3.1</td>
<td>80.1</td>
<td>2.0</td>
<td>0.3</td>
<td>1.3</td>
<td>0.6</td>
<td>2.5</td>
<td>0.8</td>
<td>4.8</td>
<td>3.8</td>
</tr>
<tr>
<td>V3</td>
<td>1.5</td>
<td>2.1</td>
<td>74.2</td>
<td>1.5</td>
<td>3.2</td>
<td>1.2</td>
<td>6.5</td>
<td>1.3</td>
<td>4.0</td>
<td>4.5</td>
</tr>
<tr>
<td>V4</td>
<td>1.9</td>
<td>1.0</td>
<td>0.5</td>
<td>85.1</td>
<td>1.2</td>
<td>0.7</td>
<td>1.6</td>
<td>0.7</td>
<td>6.6</td>
<td>0.7</td>
</tr>
<tr>
<td>V7</td>
<td>4.5</td>
<td>0.9</td>
<td>8.1</td>
<td>1.0</td>
<td>1.1</td>
<td>1.5</td>
<td>73.2</td>
<td>1.1</td>
<td>5.0</td>
<td>3.4</td>
</tr>
</tbody>
</table>

Table 1
<table>
<thead>
<tr>
<th></th>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
<th>V6</th>
<th>V7</th>
<th>V8</th>
<th>V9</th>
<th>V10</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>69.1</td>
<td>8.1</td>
<td>3.2</td>
<td>1.4</td>
<td>3.0</td>
<td>0.7</td>
<td>6.7</td>
<td>0.9</td>
<td>6.4</td>
<td>0.5</td>
</tr>
<tr>
<td>V2</td>
<td>1.7</td>
<td>86.1</td>
<td>2.0</td>
<td>0.6</td>
<td>1.6</td>
<td>0.6</td>
<td>2.4</td>
<td>0.4</td>
<td>2.7</td>
<td>2.1</td>
</tr>
<tr>
<td>V3</td>
<td>0.6</td>
<td>1.4</td>
<td>84.8</td>
<td>0.5</td>
<td>1.4</td>
<td>0.3</td>
<td>5.4</td>
<td>0.4</td>
<td>2.3</td>
<td>2.8</td>
</tr>
<tr>
<td>V4</td>
<td>0.7</td>
<td>0.2</td>
<td>0.9</td>
<td>90.3</td>
<td>1.7</td>
<td>0.4</td>
<td>1.7</td>
<td>0.4</td>
<td>3.1</td>
<td>0.7</td>
</tr>
<tr>
<td>V7</td>
<td>1.0</td>
<td>1.1</td>
<td>4.6</td>
<td>0.5</td>
<td>0.3</td>
<td>0.5</td>
<td>87.2</td>
<td>0.3</td>
<td>2.6</td>
<td>2.0</td>
</tr>
</tbody>
</table>

Table 2