On Rotational Invariance for Texture Recognition

X. Shi and R. Manduchi
Dept. of Computer Engineering, University of California, Santa Cruz
{jennifer,manduchi}@soe.ucsc.edu*

Abstract

In this paper we analyze the effect of rotational invariant operators for texture recognition using cross-validation experiments with different sample sizes. This work presents three main contributions. First, invariant operators for steerable filter banks are derived analytically using the Lie group approach. Second, the use of "randomized invariants" for steerable texture analysis is introduced. Randomized invariants produce classification rates that are intermediate between those of non-invariant and of invariant features. Third, a thorough quantitative analysis is presented, highlighting the relationship between classification performances and training sample size, textural characteristics of the data set, and classification algorithm.

1. Introduction

This paper investigates the properties of invariant operators in the context of texture classification. Invariance to "nuisance" parameters (orientation, scale, illumination, etc.) is normally considered beneficial for computer vision. However, this is not necessarily the case for all vision tasks. While removing the effect of nuisance parameters, invariance may also destroy relevant information, thus increasing ambiguity. When is invariance beneficial then? Previous work [9] argued that a critical factor determining the need for invariance is the size of the training data. When a large amount of well–representative data, under all expected realizations of the nuisance parameters, is available, then, in principle, a classifier should have enough information to make optimal (in Bayesian sense) decisions, and invariance can be detrimental. If, instead, decisions must be taken based on very little training data (as is the case in many object or scene recognition problems), then it is clear that invariance can and should be used.

In this work, we concentrate on the problem of texture–based image classification. Our leading application is scene understanding for outdoor autonomous vehicles [16]. Texture and color cues may provide important information about the materials visible in the scene, which can be used to select optimal paths for a moving robot. In particular, here we consider the issue of in–plane rotational invariance. We begin by describing in Sec. 2 how maximal rank rotational invariant operators can be computed in exact form for texture features extracted by steerable filter banks using the Lie group approach. In Sec. 3 we extend the notion of rotation invariance by considering *randomized invariants*. Randomized invariants are obtained by artificially perturbing a descriptor according to the same type of transformation for which invariance is sought. The idea of randomized invariance is interesting for two main reasons. First, it allows one to approximate an actual invariant without the need for deriving an analytical expression of it, which may be a difficult and sometime impossible task. Second, randomized invariants are parameterizable, meaning that the amount of induced variance can be controlled.

In order to provide an experimental assessment of the effect of rotational invariant and randomized invariant operators, we tested two different classification algorithms with a cross–validation procedure over three different image sets. Two classes were defined for each data set. Both classifiers operate on the features obtained by a bank of multi–scale steerable filters. The two classification algorithms are, respectively: a Maximum Likelihood classifier based on Mixture–of–Gaussian modeling of the class–conditional likelihoods; and an AdaBoost classifier [5]. In particular, a feature selections scheme was used in conjunction with AdaBoost when dealing with randomized invariants. This algorithm, inspired by the popular feature selection scheme of [30], builds a combination of weak classifiers based on different randomized invariants. The idea is that a classifier that uses both non–invariant and invariant features can adapt itself naturally to the characteristics of the data. Our cross–validation experiments with different sizes of training samples are described in Sec. 4.

2. Rotational Invariance for Texture

A mainstream approach to texture analysis is based on the representation of texture patches by low–dimensional local descriptors. A linear filter bank is normally used for this purpose. The vector (descriptor) formed by the outputs of N filters at a certain pixel is a rank–N linear mapping of the graylevel profile within a neighborhood of that pixel. The

*This work was supported by NSF under contract IIS–0222943.

marginal or joint statistics of the descriptors are then used to characterize the variability of texture appearance.

It is customary to choose analysis filters that are scaled and rotated version of one or more prototype kernels (with odd and even symmetry). If the prototype filter kernel is well localized in both the spatial and frequency domain, this approach provides an effective sampling of the local spectrum along the semantically meaningful axes of scale and orientation. This representation is also attractive because it transforms in a predictable way under the action of similarity transformations of the domain (isotropic scaling, in–plane rotation). If $l(x, y, \sigma, \theta)$ is the descriptor component corresponding to a kernel at scale σ and orientation θ, one may expect that, after rescaling of the domain centered at (x, y) by factor $\Delta\sigma$ and rotation around (x, y) by angle $\Delta\theta$, the new output should be approximately equal to $l(x, y, \sigma - \Delta\sigma, \theta - \Delta\theta)$

Scale and rotation invariance has often been advocated for texture analysis. The quest for invariance stems from the fact that the position and orientation of the camera in the scene or of a surface patch with respect to the camera cannot be constrained (generic viewpoint assumption). For example, a rotational invariant texture feature is a quantity that is independent of any rotation of the camera around its optical axis, while a scale invariant feature is unaffected by foreshortening.

Is the hypothesis of complete prior ignorance about the viewpoint acceptable in this context? It is reasonable to argue that, at least for some classes of objects and for some specific applications, it is not. As a simple, intuitive example, consider the case of an autonomous vehicle that analyzes the terrain in front of itself to figure out which path to take. The different types of terrain cover that can be expected in natural environments (grass, soil, gravel, mud) can, at least in principle, be characterized by their visual texture. For example, the texture descriptors of the class "grass" would typically have high power at vertical orientation. Of course, grass blades will not always look vertical, because, for example, the wind may be blowing, or the vehicle may roll left and right. Still, one may expect that, on average, the response at horizontal orientation should be lower than at vertical orientation. In other words, the orientation at which a generic texture patch is seen, is not an uniform random variable in $[-\pi, \pi]$. Note that, if the texture orientation of a certain class *is* uniform, then, for a given scale, the marginal densities of the filter outputs at different orientations will be identical.

A rotational invariant operator maps all values of the descriptor vector corresponding to the planar rotation of a given patch (formally, the *orbit* of the rotation group action on the descriptor [28]) onto a single value. The problem with this operation is that precious information for recognition may be lost in the process. This fairly intuitive notion is a consequence of a very general phenomenon that can be formalized in terms of the classification of a random variable z into a pre–determined set of classes as follows [9]:

Proposition 1: The Bayes rate under 0—1 loss can never decrease as a consequence of a non–invertible transformation of z.

The Bayes rate represents the minimum achievable misclassification rate. Since an invariant operator performs a non–invertible transformation of the descriptor, according to Proposition 1 no benefit may derive from it. In fact, it is easily seen that, if the probability density function (pdf) of the descriptor vector is constant along any orbit of the transformation group action, then the Bayes rate remains unchanged. This agrees with intuition in the case of generic viewpoint assumption: if there is no reason to believe that a patch will be seen at a particular orientation, rotational invariance should do not harm.

Is invariance useless (at best) as Proposition 1 seems to imply? Years of experience with invariant operators in vision have shown that, when properly chosen, invariant operators are actually very useful. For example, [29] observed that rotational invariant texture features seem to produce better results than non–invariant features for the CUReT database. This apparent contradiction may be resolved by taking into account the small sample size phenomenon. The Bayes rate refers to the Bayes classifier, which assumes full knowledge of the statistical description of the data at hand. In fact, one usually has only a limited–size sample of data for training, and therefore the issue of generalization plays an important role. A useful statistical tool, the bias–variance theory for classification [2, 6, 7, 27], was advocated in [9] to describe this phenomenon. Basically, invariant descriptors are "simpler" and therefore generalize better than non–invariant ones, meaning that they are more resilient to the unavoidable variance in the choice of the training sample. Thus, in spite of the fact that invariance may increase ambiguity and therefore the classifier's bias, for small training samples, which have higher associated variance (a main cause of misclassification), invariance may increase the effective classification rate. Indeed, in the case of recognition or matching from just one image, some sort of invariance has been shown to be necessary [21, 15]. However, for other types of tasks and features, this may not always be the case.

One of the goals of this paper is to verify this notion by means of cross-validation experiments on real–world data using rotational–invariant texture features. The next section describes the selected features and invariant operators.

2.1. Invariance and Lie groups

Rotational invariance can be enforced in a number of ways. Perhaps the simpler approach is to choose features that are "naturally" invariant, such as differential invariants [1, 14, 21] or integral invariants (moments) [11, 26]. Another possibility is to start from a given descriptor, and to make it invariant via a suitable transformation. When the feature can be expressed as an explicit function of orientation (such as the result of projection onto a set of angular Fourier basis kernels, implemented by the convolution with a bank of oriented filters), then, referred to the orientation axis, a rotation of the input data transforms into a cyclic shift. Simple translation–invariant operators can then be used, such as the magnitude of the Fourier transform [8] or relative phase components [24]. Another approach is to find the "dominating" orientation and normalize the descriptor with respect to it [13, 12, 20, 15].

A very general approach to the design of invariants was presented in [28]. The idea is to model the effect of "nuisance" parameters (such as rotations) as orbits of Lie group actions. This theoretical framework has two main advantages. First, it directly provides the number of independent invariants, which is equal to the number of independent parameters needed to fix the position of a point in measurement space, minus the orbit dimension. Second, it shows that designing invariants is equivalent to finding the solutions of a system of partial differential equations (PDE), thereby providing a systematic algorithm for the design. Assume that for a given pixel (x, y), the descriptor vector is $(l_0, \ldots, l_{N-1})$. Since the group of planar rotations, *SO(2)*, has dimension 1, we should expect that $N-1$ independent rotation invariants exist. More precisely, a rotational invariant operator is a mapping f from the N–dimensional manifold representing the descriptors' space to a $(N-1)$–dimensional manifold, that solves the following PDE:

$$\sum_{i=0}^{N-1} \frac{\partial f}{\partial l_i} \frac{\partial l_i}{\partial \theta}\bigg|_{\theta=0} = 0 \quad (1)$$

where $\frac{\partial l_i}{\partial \theta}\big|_{\theta=0}$ represents the incremental ratio of the descriptor component l_i as the image is rotated around pixel (x, y). Note in passing that the trivial, rank–0 invariant (a constant f) also solves (1). The goal of invariant design is to find a *maximal rank* descriptor [19], such that the rank of its Jacobian matrix at l is equal to $N-1$. Of course, any function of an invariant descriptor is invariant as well.

Thus, in order to find a rotational invariant, we must be able to (1) express the partial derivatives of the measurements with respect to a rotation, and (2) solve the PDE in (1). In the next section we show that steerable filter banks, a widely used class of texture descriptors, allow us to accomplish both tasks.

2.2. Steerability Invariance

In its most general form, steerability is defined as the property of a function to transform under the action of a Lie group as a linear combination of a set of fixed *basis functions* [10]. We will consider here only steerable filters with basis functions that are the rotated versions of a prototype kernel $h(x, y)$ (*equivariant function space* [10]). We will also assume that the prototype kernel is axially symmetric, so that only rotations between 0 and π are of interest. If $h_i(x, y)$, with $0 \le i < N$, is the version of the kernel rotated by iN/π, then the rotated version of $h(x, y)$ by angle θ can be written as a linear combination of the $h_i(x, y)$, with coefficients that only depend on θ. This also implies that, if $l_i(x, y) \triangleq l(x, y, \theta_i)$ is the filtered version of image $l(x, y)$ rotated by iN/π, then, for a generic angle θ,

$$l(x, y, \theta) = \sum_{i=0}^{N-1} l_i(x, y) k_i(\theta) \quad (2)$$

for suitable *interpolation functions* $k_i(\theta)$ that are independent of l. We will concentrate on steerable prototype kernels that are higher order directional derivatives (in x) of an isotropic Gaussian function $G(x, y)$. Such functions can be written as $(-1)^M H_M(x) G(x, y)$, where $H_M(x)$ is the Hermite polynomial of order M, and M is the order of the derivative. In [4] it was shown that $N = M + 1$ basis functions are sufficient to synthesize $H_M(x)G(x, y)$, and that the interpolation functions in this case are trigonometric polynomials.

A "steerability invariant" operator can be applied on the output of the basis filter bank, transforming the vector $(l_0, \ldots, l_{N-1})$ into the rotational invariant vector $(\bar{l}_0, \ldots, \bar{l}_{N-1})$, where $\bar{l}_i = f_i(l_0, \ldots, l_{N-1})$ and $(f_0, \ldots f_{N-1})$ are independent invariant operators. It is easy to prove that, thanks to the equivariant nature of the chosen filter bank, the PDE in (1) can be expressed in simple form as:

$$\bigtriangledown f^T H \underline{l} \quad (3)$$

where $\bigtriangledown f = (\partial f/\partial l_1, \ldots, \partial f/\partial l_{N-1})^T$, $\underline{l} = (l_0, \ldots, l_{N-1})^T$, and H is a Toeplitz antisymmetric matrix. The solutions of (3) turn out to be (functions of) polynomials in the $\{l_i\}$.

3. Randomized Steerability Invariants

The theory of invariant operators described so far allows only for a dyadic choice: either a feature is invariant to a transformation, or it is not. How could one enforce a "soft" version of invariance? A possible strategy is to add "jittered" versions of the training samples, as in [3, 22, 23]. In other words, one may apply random transformations to the training data, with the transformation parameter (e.g., rotation) moving slightly around the identity (in our case, θ=0).

This strategy was given the name of *randomized invariant* in [9]. Randomized invariants can be easily understood if, in the chosen representation of the feature space, the orbits of transformation group actions are parallel to one of the axes. In this case, the invariant operator simply removes the dimension corresponding to this axis, an operation that corresponds to *marginalization* (i.e., integration along that axis) in terms of the pdf of the feature. The idea of randomized invariants is to substitute marginalization with convolution of the pdf along the chosen axis with a positive, normalized kernel. As the standard deviation of this kernel approaches infinity, this convolution is equivalent to marginalization. As it approaches zero, it is leaves the pdf unchanged. A simple way to realize such a convolution is to add to the training data samples of iid random noise with pdf equal to the chosen kernel. This operation is equivalent to perturbing the descriptor $\underline{l}$ along the transformation orbit by a random quantity. Note that this operation is only performed at training time; the data to be classified is left untouched.

Using equivariant filter banks, it is easy to create randomized invariant features, by using the interpolation functions to steer all basis functions by the same (random) angle θ. In our experiments, the perturbation θ was a white noise with marginal density uniform in $[-r\pi, r\pi]$, where $0 \leq r \leq 1$ is the *randomization index*.

4. Experiments

We used three different image sets for our experiments, all of which containing outdoor images. The first data set is the Outex-NS-00000 set in the Outex image database[1] [18]. This set contains 20 images, which were originally hand–labeled into 6 classes. We retained one such class (“road”) and considered one cumulative class (“vegetation”) for three other original classes (“trees”, “bushes”, “grass”). The remaining classes were neglected.

The second data set, named ROAD and shown in Fig. 2, contains 10 images taken by a robot moving in natural environments. These images were used already in [9] for texture classification analysis. The images were hand–labeled into two different terrain cover classes: “soil” and “low vegetation”[2]. The third data set, named TREES and shown in Fig. 4, contains 14 images taken in a redwood forest. Two classes, “tree trunks” and “undergrowth” were identified. Note that the texture of “tree trunks” has very strong directionality characteristics.

We used combinations of even (M=2) and odd (M=3) steerable filters, in their non–invariant, invariant and randomized invariant form, at two different scales, to extract texture features for our classification experiments. Thus, the dimension of the feature vector was $2 \cdot (3 + 4) = 14$ in the non–invariant case, $2 \cdot (2+3) = 10$ in the invariant case.

We ran a number of cross–validation experiments, by randomly selecting a training sample from the pool of all features for the labeled image areas, and testing the classifier designed using such sample on the remaining data[3]. For each chosen size of the training sample, we ran 90 tests, each time randomly selecting the training data. The average correctness rate of the tests, as well as the results’ standard deviation, was computed and plotted for a number of different training data portions.

Two different classifier were used for our experiments. The first one was a Maximum Likelihood (ML) classifier based on Mixture–of–Gaussians (MoG) modeling of the conditional likelihoods. The number of Gaussian modes for each class and for each training sample size was chosen using the cross–validation procedure of [25]. A minimum of 2 and a maximum of 5 modes were allowed. The second classifier was based on the AdaBoost algorithm [5] a well–known technique based on the combination of a number of “weak classifiers”, which has good generalization properties.

The two classifiers were used with identical modalities in cross–validation experiments with both non–invariant and invariant features. For the case of randomized invariant features, two different approaches were used. For the ML classifier, we chose a fixed value for the randomization index ($r = 0.25$). For the AdaBoost classifier, we used the following strategy, inspired by the feature selection scheme of [30]. The randomization index r was initially quantized in 101 steps between 0 and 1, thus including the non–invariant and 100 randomized invariant features. A weak linear classifier was built for each such feature, and the classifier using the feature with smallest associated misclassification rate (as tested on the training data set) was chosen, while the associated feature was removed from the set. The training data was then weighted based on the classification results as in the canonical AdaBoost scheme, and the procedure was iterated. Convergence was declared when either a maximum number of iterations (in our experiments, 20) was reached, or when the best–performing weak classifier using any remaining feature was only slightly better than random guessing. The main difference with respect to the original feature selection scheme of [30] is that the concept of different “features” only applies to the classifier design. In fact, each weak classifier operates on the same feature space (the reader is reminded that randomized invariants are only a tool for classifier design, and that the test data is not perturbed).

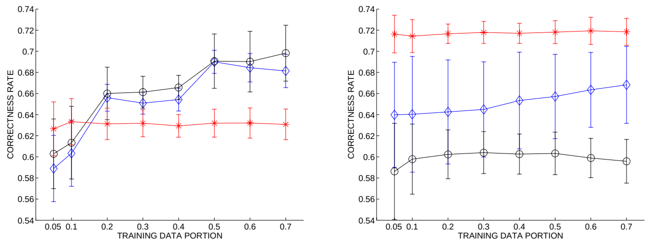
The results of our experiments are shown in Figs. 1, 3

[1]http://www.outex.oulu.fi

[2]The label files for the ROAD and TREES image sets are available on request.

[3]Similar tests, but using whole images for training, were also conducted, obtaining comparable error rates.

and 5. We leave the analysis of such results to the next section.



(a): ML Classifier (b): AdaBoost Classifier

Figure 1: Cross–validation classification experiments on the Outex data set with different proportions of training data. The plot shows the average correctness rate over all tests, together with $\pm$ 1 standard deviation marks. Circles: non–invariant; Stars: invariant; Diamonds: randomized invariant.

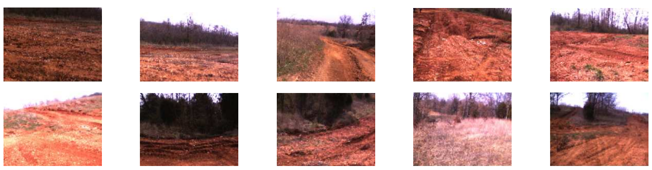
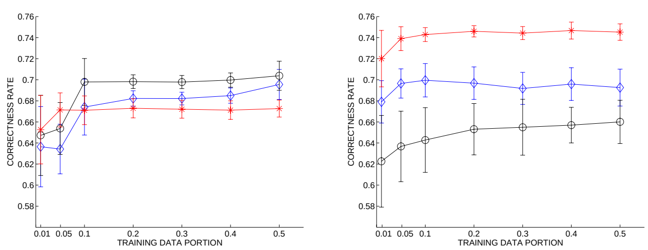


Figure 2: The ROAD image set.



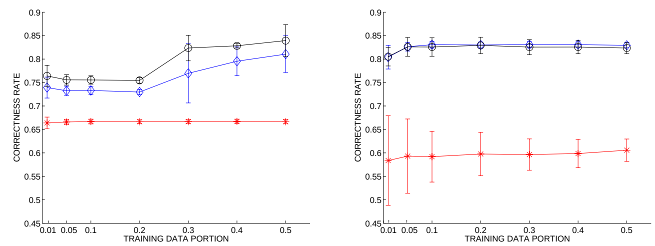
(a): ML Classifier (b): AdaBoost Classifier

Figure 3: Cross–validation classification experiments on the ROAD data set with different proportions of training data (see caption of Fig. 1).

5. Discussion and Conclusions

The experiments of the previous section highlighted a number of interesting properties of rotational invariant operators for texture. First of all, it is seen that the effect of rotational invariance depends dramatically on the directionality characteristics of the texture in the data set. This is clearly seen in the TREES data set, which contains one class ("tree trunks") that has a strong vertical texture. In this case, rotational invariance removes important discriminative information, thereby reducing the correct classification rate.

Figure 4: The TREES image set.



(a): ML Classifier (b): AdaBoost Classifier

Figure 5: Cross–validation classification experiments on the TREES data set with different proportions of training data (see caption of Fig. 1).

For the remaining data sets, the ML and the AdaBoost classifiers showed quite different characteristics. In the case of the ML classifier, the correctness rate decreases (as expected) with the size of the training sample, as a consequence of increased variance. For large enough training samples, the non–invariant classifier is shown to outperform the invariant one. However, the situation is reversed for smaller training sets. This confirms the intuitive notion that invariant operator can improve generalization. Indeed, a cross–over point can be seen in Figs. 1 and 3 for a certain training sample size. The situation is very different in the case of the AdaBoost classifier, which, at least for the training sample sizes considered in this work, always performs better with the invariant features. The exact reason for this discrepancy is the object of current research. Note also that the best AdaBoost classifier for each data set (i.e., using invariant features in the Outex and ROAD sets, and non–invariant features in the TREES set) always outperforms the best ML classifier. On the other hand, the worst AdaBoost classifier is worse than the worst ML classifier for each set. Also, AdaBoost seems to be more sensitive to the choice of the training data, as shown by the higher variance of the correctness rate results.

For what concerns the use of randomized invariants, it is shown that in the ML classification case, randomized invariants give intermediate performance between the non-invariant and the invariant case when the training sample is sufficiently large, but degrade the classification rate for small samples. For the AdaBoost case, the proposed feature selection algorithm yields correctness rates that are consistently placed between the non–invariant and the invariant case for the Outex and the ROAD sets, and almost identical to the non–invariant case for the TREES set. In the latter case, inspection of the histogram of the randomization indices chosen by the algorithm shows that only low indices were selected. Hence, the proposed randomized invariant feature selection scheme looks like a viable and "safe" strategy when one is uncertain about the textural characteristics of the data to be classified, and thus about whether rotational invariance should or should not be used.

References

[1] P.E. Danielsson, "Rotation-invariant linear operators with directional response", *5thIntl Conf. Patt. Rec.*, Miami, 1980.

[2] P. Domingos, "A unified Bias-Variance decomposition for zero-one and squared loss", *AAAI/IAAI*, 564–569, 2000.

[3] H. Drucker, R. Schapire, and P. Simard, "Boosting performances in neural networks", *Intl. Journal of Pattern Recogn. and Artif. Intell.*, 7:705–719, 1993.

[4] W.T. Freeman and E.H. Adelson, "The design and use of steerable filters", *IEEE Trans. PAMI*, 13(9):891–906, 1991.

[5] Y. Freund and R. Schapire, "A short introduction to boosting", *J. Japan.Soc. for Artif. Intel.*, 14(5): 771-780, 1990

[6] J.H. Friedman, "On bias, variance, 0/1–loss, and the curse-of-dimensionality", *Data Mining and Knowledge Discovery*, 1: 55-77, 1997.

[7] S.L. Geman, E. Bienenstock and R. Doursat, "Neural networks and bias/variance dilemma", *Neural Computation* 4: 1-58, 1992.

[8] H. Greenspan, S. Belongie, R. Goodman, P. Perona, S. Rakshit, and C. H. Anderson, "Overcomplete steerable pyramid filters and rotation invariance", *IEEE. CVPR*, 222-228, 1994 .

[9] X. Shi and R. Manduchi, "Invariant operators, small samples, and the Bias-Variance dilemma", *IEEE CVPR*, Washington DC, June 2004..

[10] Y. Hel-Or and P. Teo, "Canonical decomposition of steerable functions," *IEEE CVPR*, 1996.

[11] M-K. Hu, "Visual pattern recognition by moment invariants", *IRE Trans. on Information Theory*, IT-8:179-187, 1962.

[12] M. Kass and A. Witkin, "Analyzing oriented patterns", *Comp. Vision Graphics Image Proc.*, 37:362385, 1987.

[13] H. Knutsson and G.H. Granlund, "Texture analysis using two-dimensional quadrature filters", *IEEE Workshop on Comp. Arch. Patt. Anal. Image Database Mgmt*, 388397, 1983.

[14] J.J. Koenderink and A.J. van Doorn, "Representation of local geometry in the visual system, *Biological Cybernetics*, 55:367375, 1987.

[15] D.G. Lowe. "Distinctive image features from scale-invariant keypoints", *IJCV*, 60(2):91–110, November 2004.

[16] R. Manduchi, A. Castano, A. Talukder, and L. Matthies, "Obstacle detection and terrain classification for autonomous off–road navigation", *Autonomous Robots*, 18, 81-102, 2005.

[17] K. Mikolajczyk and C. Schmid, " A performance evaluation of local descriptors", *IEEE CVPR*, 2003.

[18] T. Ojala, T. Maenpaa, M. Pietikainen, J. Viertola, J. Kyllonen and S. Huovinen, "Outex – New framework for empirical evaluation of texture analysis algorithms", *ICPR*, Quebec, Canada, 2002.

[19] P.J. Olver, *Applications of Lie groups to differential equations*, Springer–Verlag, 1993.

[20] P. Perona and J. Malik, "Detecting and localizing edges composed of steps, peaks and roofs", *Proc. ICCV*, 1990.

[21] C. Schmid and R. Mohr, "Local grayvalue invariants for image retrieval", *IEEE Trans. PAMI*, 5(19):530–4, May 1997.

[22] B. Schölkopf, C. Burges, and V. Vapnik, "Incorporating invariances in support vector learning machines", *Artificial Neural Networks – ICANN'96*, 1996.

[23] P. Simard, Y. Le Cun, and J. Denker, "Efficient pattern recognition using a new transformation distance", *NIPS*, 5, 1993.

[24] E. P. Simoncelli, "A rotation-invariant pattern signature", *IEEE ICIP*, Lausanne, Switzerland, 1996.

[25] P. Smyth, "Clustering using Monte Carlo cross–validation", *Proc. KDD*, 126-133, 1996.

[26] C.Teh and R.T. Chin, "On image analysis by the method of moments, *IEEE Trans. PAMI*, 10(4):496-513, 1988.

[27] R. Tibshirani, "Bias, variance and predicition error for classification rules", Tech. Report, Dept. of Prev. Medicine and Biostatistics, Univ, of Toronto, 1996.

[28] L. Van Gool, T. Moons, E. Pauwels and A. Ooserlinck, "Vision and Lie's approach to invariance", *Image and Vision Computing*, 13(4):259–77, 1995.

[29] M. Varma and A. Zisserman, "A statistical approach to texture classification from single images", *IJCV*, 2005.

[30] P. Viola and M. Jones, "Robust real-time object detection", *IJCV*, 2002