Pairwise Constraints for Matching, Perceptual Grouping and Recognition

Marius Dan Leordeanu
mleordea@andrew.cmu.edu

Thesis Proposal
March 10th, 2008

Committee Members
Martial Hebert (chair)
Rahul Sukthankar
Fernando de la Torre
David Lowe (external)

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

Object category recognition is a challenging problem in computer vision, which currently receives a growing interest in the field. This problem is almost ill-posed, because there is no formal definition of what constitutes an object category. While people largely agree on common, useful categories, it is still not clear which are the objects' properties that help us group them into such categories. In this thesis we represent the object category models as graphs of features, and focus mainly on the second order relationships between them: pairwise category-dependent (e.g. shape) as well as pairwise perceptual grouping constraints (e.g. geometrical and color based). The main theme of this thesis is that higher order relationships between model features are more important for category recognition than local, first order features. We present several novel algorithms that take full advantage of such pairwise constraints. Firstly, we present our spectral matching algorithm for the Quadratic Assignment Problem (also known as Graph Matching), along with a novel, efficient method for learning the pairwise parameters. Secondly, we present a novel optimization method which can handle nonlinear, complex functions, and present some of its applications in the context of our work. Thirdly, we discuss our object category recognition approach based on shape alone, which uses pairwise geometric constraints only. Next, we explore ways (based on both color and geometry) to establish perceptual grouping relationships between pairs of features, which are category independent. And finally, we talk about how we plan on combining both the category dependent and the perceptual relationships in order to perform object category recognition.

Contents

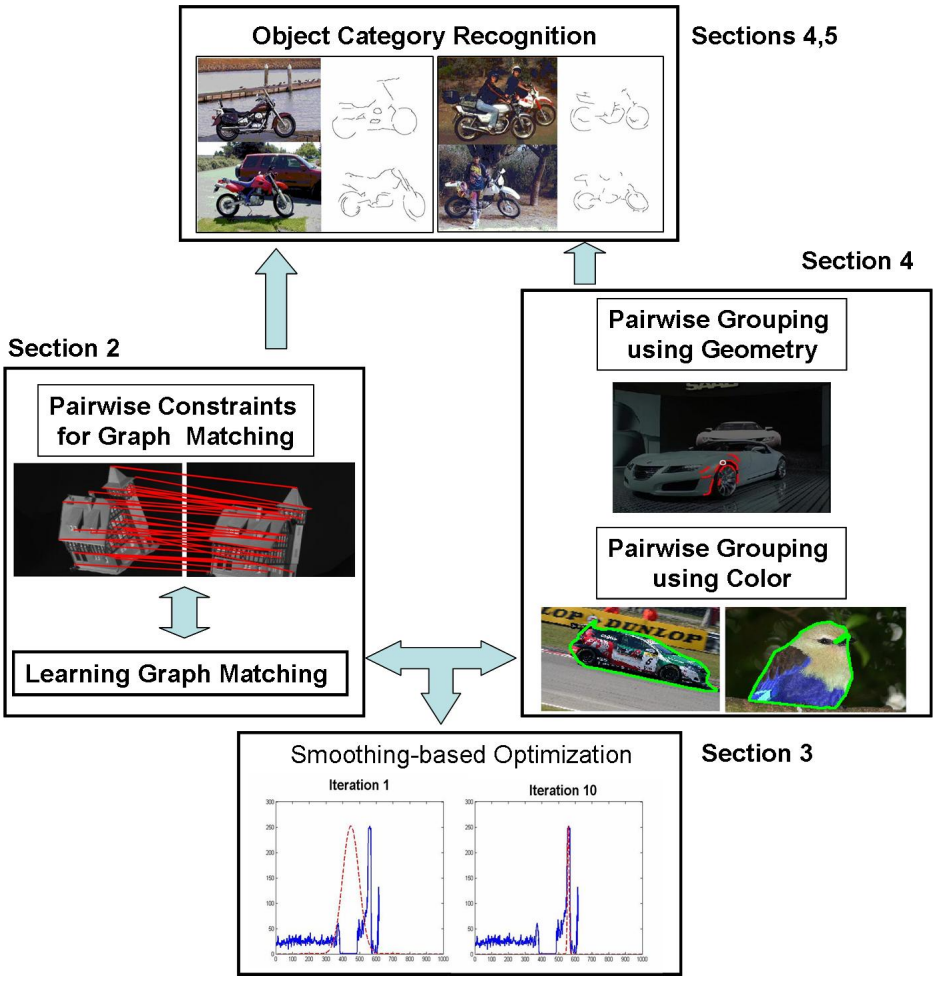

Figure 1. Thesis overview: spectral matching (using pairwise constraints) is combined with perceptual grouping (using both geometric and color based pairwise constraints) for object category recognition. Smoothing-based optimization (Section) can be used for learning the pairwise constraints used by spectral matching

1. Introduction

Object category recognition is a challenging problem in computer vision, which currently receives a growing interest in the field. While humans can accurately distinguish between thousands of categories, current state of the art algorithms are far behind. This problem is almost ill-posed, because there is no formal definition of what constitutes an object category. Philosophically, any arbitrary definition could characterize an object category (*e.g.All things with a white patch*), but such categories are not useful to us, humans. While people largely agree on common, useful categories, it is still not clear which are the objects' properties that help us group them into such categories.

In this thesis we represent these object category models as graphs of features, and focus mainly on the second order (pairwise) relationships between them: category-dependent as well as perceptual grouping constraints. This differs from the popular bag of words model [18], which concentrates exclusively on local features, ignoring the higher-order interactions between them. The main theme of this thesis is that higher order relationships between model features are more important for category recognition than local, first order features. This is consistent with research in cognitive science hypothesizing that human category recognition is based on pairwise relationships between object parts [33]. Studies in human vision show that simple, unary features, without higher order relationships (such as geometric constraints or conjunctions of properties),

are the ones mainly used at the pre-attentive levels, but are not sufficient at higher cognitive levels where object category recognition takes place [65]. The importance of using pairwise relationships between features was recognized early on, starting with Ullman's theory of the correspondence process, which introduced the notion of *correspondence strength* that takes into consideration both the local/unary affinities, but also pairwise interactions between features [47]. Interpretation trees and transformation voting alignment are another example of early work by Lowe [44], Grimson [30], and others [27], [7] using higher order relationships, but they were limited by an explicit parametric model of the global transformation between the input and the model. More recent techniques for category recognition based on graphical models [40] also used higher order constraints, without considering a global transformation.

In this thesis we take full advantage of the power of second order relationships by developing rich constraints between model parts (as well as input features) that are appropriate for the recognition task. We also present our spectral matching algorithm [38], based on such pairwise constraints, which is very efficient due to the rarity of accidental alignments. We try to achieve two goals in approaching the problem:

1. Understand what is significant for solving the task: in this case, understanding that pairwise relationships are very powerful if used properly
2. Take advantage of particular, unique properties of the actual vision problem in order to develop efficient algorithms, such as spectral matching.

We present our matching algorithm in Section 2.1, focusing more on its theoretical and empirical properties than on its application to recognition. In Section 4 we show how to apply this algorithm, within a novel framework, to the object category learning and recognition problem. While most of the work in this area ignores the geometric relationships between features and focuses primarily on the features themselves, we instead argue that the geometric/higher-order relationships are often more powerful than the local features.

We realize that the recognition problem is a high level task, and that it cannot be fully solved by simply matching low-level features. For improving the matching performance and efficiency one has to consider intermediate cues from geometric and color based perceptual grouping, which are object category independent, and could provide powerful *a priori* information that could direct the higher level recognition process onto the right path. In Section 5.1 we present our proposed work on how to perform perceptual and segmentation based grouping. As recognized early on [44], the grouping process gives the recognition algorithm a rough idea of where objects are and how features should be grouped together. After this stage the matching algorithm can use this lower-level information to ignore most of the background clutter. In Section 5 we propose a novel method for integrating the lower-level grouping information with the higher level, model dependent, geometric and appearance-based information into one unified framework.

Figure 1 shows a high level overview of our thesis. Both the matching technique (spectral matching) from Section 2.1 and the grouping approaches proposed in Section 5 are combined for solving the object category recognition problem. For learning the matching parameters we use our novel optimization method for complex nonlinear functions, presented in Section 3.

In Figure 2 we show an alternative overview of our thesis in terms of the completed and the proposed work. The most significant completed work includes:

1. Spectral Matching: an efficient algorithm for feature matching using second order terms. It is basically a solver for the Quadratic Assignment Problem applied to computer vision.
2. Object Recognition without Grouping: a novel approach for semi-supervised learning of object categories using higher order interactions between simple features. The same framework can easily incorporate more complex features

The proposed work (or work in progress) can be summarized as follows (in the order of its importance):

1. Object Recognition with Grouping: a novel, unifying approach for combining object specific information (matching model parts to image features) and image based (model independent) grouping cues
2. An efficient algorithm for grouping (in a soft way) image features based on color distributions and geometric relationships
3. A new general optimization method for complex functions and its applications to grouping and object recognition
4. An efficient method for learning graph matching, specifically designed for the spectral matching algorithm

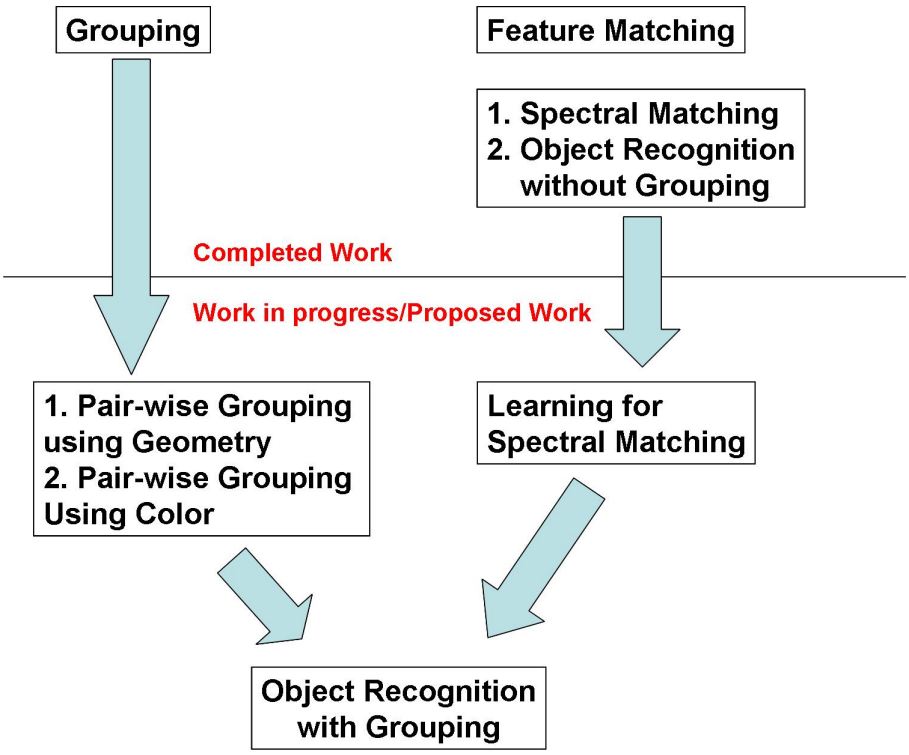

Figure 2. Completed and proposed work

2. Feature Matching using Pairwise Constraints

2.1. Spectral Matching

There are many tasks in computer vision that require efficient techniques for finding consistent correspondences between two sets of features, such as object recognition, shape matching, wide baseline stereo, 2D and 3D registration. In this thesis we present an efficient technique that is suitable for such applications. Our method finds consistent correspondences between two sets of features, by taking in consideration both how well the features' descriptors match and how well their pairwise geometric constraints (or any other type of pairwise relationship) are satisfied. Our formulation can accommodate different kinds of correspondence mapping constraints, such as allowing a data feature to match at most one model feature (commonly used), or allowing a feature from one set to match several features from the other set (used in shape matching [4]).

The features could consist of points, lines, shape descriptors or interest points, depending on the specific application. For problems where the features are non-discriminative ($e.g.$ points), it is the features pairwise geometric information that helps in finding the right correspondence. When discriminative features are extracted ($e.g.$ interest points) then both the geometry and the properties of each individual feature can be used.

The main difficulty of the correspondence problem is its combinatorial complexity. Our approach avoids the combinatorial explosion by taking advantage of the spectral properties of the weighted adjacency matrix M of a graph, whose nodes are the potential assignments $a = (i, i')$ (Section 2) and weights on edges are the agreements between pairs of potential assignments. We use the terms *assignment* and *correspondence* interchangebly .

Our method is based on the observation that the graph associated with M contains:

1. a main strongly connected cluster formed by the correct assignments that tend to establish agreement links among each

other. These agreement links are formed when pairs of assignments agree at the level of pairwise relationships ($e.g.$ geometry) between the features they are putting in correspondence.

2. a lot of incorrect assignments outside of that cluster or weakly connected to it, which do not form strongly connected clusters due to their small probability of establishing agreement links and random, unstructured way in which they form these links.

These statistical properties motivate our spectral approach to the problem. We start by first finding the level of *association* of each assignment with the main cluster, by inspecting the eigenvector of M corresponding to its largest eigenvalue (principal eigenvector). Then we keep rejecting the assignments of low association, until the constraints on the correspondence mapping are met (Section 3). Spectral methods are commonly used for finding the main clusters of a graph, in tasks such as segmentation [63], grouping [46], [60], and change detection [58]. Shapiro and Brady [62] also proposed a spectral technique for correspondence problems, later improved by Carcassoni and Hancock [9], but their formulation is different and it applies only to matching point sets.

In previous spectral methods, the rows and columns of M correspond to single features, and the value at $M(i, i')$ is the affinity of feature i with feature i'. Different from that work, here the rows and columns of M correspond to candidate correspondences, which are pairs of features. Now, $M(a, b)$ contains the affinity of feature pair $a = (i, i')$ with feature pair $b = (j, j')$. Also, the diagonal terms in the typical formulation are not meaningful (they measure affinity of a single feature i with itself). However, here the diagonal terms $M(a, a)$ are meaningful, and they represent the individual pair score between the two features i and i'.

Our problem formulation (Section 2.1.1) also relates to the ones given in [4] and [45]. Maciel and Costeira [45] use the fact that the integral quadratic formulation of the correspondence problem can be reduced to an equivalent concave minimization problem, without changing the optima. However, the complexity of concave minimization is still non-polynomial. Berg and Malik [4], obtain an efficient implementation by specifically designing it to allow several features from one image to match the same feature from the second image, while approximating the quadratic problem with $n+1$ linear programming problems (where n is the number of rows of M). Those papers formulate the problem as an integer quadratic programming problem by embedding the mapping constraints in the general form $Ax = b$. Instead, we relax both the integral and the mapping constraints on x, and use them only after the optimization step. We show that this relaxation makes sense due to the spectral properties of M. In this way we achieve a robust performance that is several orders of magnitude faster than those methods even on small size data sets.

2.1.1 Problem Formulation

Given two sets of features: P, containing n_P data features, and Q, having n_Q model features, a *correspondence mapping* is a set C of pairs (or *assignments*) (i, i'), where $i \in P$ and $i' \in Q$. The features in P and Q that belong to some pair from C are the *inliers*. The features for which there is no such pair in C are the *outliers*. Different problems impose different kinds of *mapping constraints* on C, such as: allowing one feature from P to match at most one feature from Q, or allowing one feature from one set to match more features from the other. Our formulation of the correspondence problem can accommodate different kinds of constraints.

For each candidate assignment $a = (i, i')$ there is an associated score or *affinity* that measures how well feature $i \in P$ matches $i' \in Q$. Also, for each pair of assignments (a, b), where $a = (i, i')$ and $b = (j, j')$, there is an *affinity* that measures how compatible the data features (i, j) are with the model features (i', j'). Given a list L of n candidate assignments, we store the affinities on every assignment $a \in L$ and every pair of assignments $a, b \in L$ in the n x n matrix M as follows:

1. $M(a, a)$ is the affinity at the level of individual assignments $a = (i, i')$ from L. It measures how well the data feature i matches the model feature i'. It is important to note that assignments that are unlikely to be correct (due to a large distance between the descriptors of i and i') will be filtered out and not be included in L. Thus, each such rejection will reduce the number of rows and columns in M by one.

2. $M(a, b)$ describes how well the relative pairwise geometry (or any other type of pairwise relationship) of two model features (i', j') is preserved after putting them in correspondence with the data features (i, j). Here $a = (i, i')$ and $b = (j, j')$. If the two assignments do not agree ($e.g.$ the deformation between (i, j) and (i', j') is too large) or if they are incompatible based on the mapping constraints ($e.g$ $i = j$) we set $M(a, b) = 0$. We assume $M(a, b) = M(b, a)$ without any loss of generality.

We require these affinities to be non-negative, symmetric ($M(a,b) = M(b,a)$), and increasing with the quality of the match (without loss of generality). The candidate assignments $a = (i, i')$ from L can be seen as nodes forming an undirected graph, with the pairwise scores $M(a,b)$ as weights on the edges and the individual scores $M(a,a)$ as weights at the nodes. Then, M represents the affinity matrix of this undirected weighted graph. If the features are highly discriminative, such as SIFT descriptors, then only a small fraction of all possible pairs (i, i') are kept as candidate matches. In this case the size of M and the dimension of the problem search space are considerably reduced. When the features are non-discriminative (such as 2D or 3D points) and there is no a priori information about candidate matches ($e.g.$ constraints on translation), all possible pairs (i, i') can be considered as candidate assignments. In general, M is an n x n, sparse symmetric and positive matrix where $n = kn_P$, and k is the average number of candidate matches for each data feature $i \in P$. Each feature $i \in P$ will usually have a different number of candidate correspondences (i, i'), $i' \in Q$. Thus the number of nodes in this graph (= number of elements in L), adapts based on the actual data and it depends mainly on how discriminative the features's descriptors are.

The correspondence problem reduces now to finding the cluster C of assignments (i, i') that maximizes the inter-cluster score $S = \sum_{a,b \in C} M(a,b)$ such that the mapping constraints are met. We can represent any cluster C by an indicator vector x, such that $x(a) = 1$ if $a \in C$ and zero otherwise. We can rewrite the total inter-cluster score as:

$$S = \sum_{a,b \in C} M(a,b) = x^T M x \tag{1}$$

The optimal solution x^* is the binary vector that maximizes the score, given the mapping constraints:

$$x^* = argmax(x^T M x) \tag{2}$$

The inter-cluster score $x^T M x$ depends mainly on three things: the number of assignments in the cluster, how interconnected the assignments are (number of links adjacent to each assignment) and how well they agree (weights on the links). Previous approaches [4], [45], gave a quadratic programming formulation to this problem by embedding the mapping constraints on x in the general form of $Ax = b$. Instead, we relax both the mapping constraints and the integral constraints on x, such that its elements can take real values in $[0, 1]$. We interpret $x^*(a)$ as the *association* of a with the best cluster C^*. Since only the relative values between the elements of x matter, we can fix the norm of x to 1. Then, by the Raleigh's ratio theorem, x^* that will maximize the inter-cluster score $x^T M x$ is the principal eigenvector of M. Since M has non-negative elements, by Perron-Frobenius theorem, the elements of x^* will be in the interval $[0, 1]$. In Section 3 we describe how we use the mapping constraints to binarize the eigenvector and obtain a robust approximation to the optimum solution.

One novel aspect of our approach is that we drop the mapping constraints during the optimization step, and use them only afterwards to binarize the eigenvector. The problem becomes one of finding the main cluster of the assignments graph and can be solved easily using the well known eigenvector technique. We show that this method is very robust, because the main cluster in the assignments graph is statistically formed by the correct assignments.

A key insight in the understanding of the statistics of M is that a pair of model features is very likely to agree (in terms of pairwise relationship between features) with the correct corresponding pair of data features. The same pair is very unlikely to agree with an incorrect pair of data features. Thus, correct assignments are expected to establish links between them, while incorrect assignments are not expected to form such links, and when they do, it happens in a random, unstructured way. This suggests that the correct assignments will form a highly connected cluster with a high association score, while the wrong assignments will be weakly connected to other assignments and not form strong clusters. The larger the value in the eigenvector $x^*(a)$, the stronger the association of a with the main cluster. Since this cluster is statistically formed by correct assignments, it is natural to interpret $x^*(a)$ as the confidence that a is a correct assignment.

2.1.2 Algorithm

We present a greedy algorithm for finding the solution to the correspondence problem. As discussed earlier, we interpret the eigenvector value corresponding to a particular assignment $a = (i, i')$ as the *confidence* that a is a correct assignment. We start by first accepting as correct the assignment a^* for which the eigenvector value $x^*(a)$ is maximum, because it is the one we are most confident of being correct. Next we have to reject all other assignments that are in conflict with a^*, as dictated by the constraints on the correspondence mapping. In our experiments these are assignments of the form $(i, *)$ or $(*, i')$ (one feature $i \in P$ can match at most one feature $i' \in Q$ and vice-versa). Note that here one could use different constraints to find the assignments that are in conflict with a^*. We accept the next correct assignment as the one with the second highest chance of being correct that has not been rejected and thus it is not in conflict with a^*. We repeat this procedure of accepting new

assignments of next highest confidence that are not in conflict with the ones accepted already, until all assignments are either rejected or accepted. This algorithm will split the set of candidate assignments in two: the set of correct assignments C^* and rejected assignments R, having the following property: every assignment from R will be in conflict with some assignments from C^* of higher confidence. Thus, no element from R can be included in C^* without having to remove from C^* an element of higher confidence.

The overall algorithm can be summarized as follows:

1. Build the symmetric positive n x n matrix M as described in section 2.
2. Let x^* be the principal eigenvector of M. Initialize the solution vector x with the n x 1 zero vector. Initialize L with the set of all candidate assignments.
3. Find $a^* = argmax_{a \in L}(x^*(a))$. If $x^*(a) = 0$ stop and return the solution x. Otherwise set $x(a^*) = 1$ and remove a^* from L.
4. Remove from L all potential assignments in conflict with $a^* = (i, i')$. These are assignments of the form (i, k) and (q, i') for one-to-one correspondence constraints (they will be of the form (i, k) for one-to-many constraints).
5. If L is empty return the solution x. Otherwise go back to step 3.

We note that the outliers are found at steps 3 and 4. They belong to weak assignments incompatible with assignments of higher confidence, or to those that have a zero corresponding eigenvector value (step 3). Different kinds of constraints on the correspondence mapping can be used to remove the assignments conflicting with higher confidence assignments (step 4). Our approach takes advantage of the fact that these constraints are usually easy to check. The algorithm provides a simple way to enforce the constraints as a post-optimization step, without the need of embedding them in the general form of $Ax = b$, required for the more expensive quadratic optimization approach. In practice our algorithm was several orders of magnitude faster than the linear programming approximation [4] to the quadratic problem , even for medium size data-sets (matching 15-20 points). In turn, the linear optimization approximation is less computationally expensive than the optimal quadratic programming approach [45], especially as the size of M increases. See [38] for the experimental analysis of the spectral matching algorithm and its performance comparison to [4].

2.2. First Method for Learning the Pairwise Constraints for Graph Matching (work in progress)

In Section 2.1 we presented our novel spectral matching algorithm, which is an optimization technique for the graph matching task, also known as the quadratic assignment problem (QAP). This problem is frequently encountered in computer vision. We briefly reiterate it here: the task is formulated as an optimization problem, with the goal of finding the assignments that maximize a quadratic score, given the constraints that one feature from one image can match only one other feature from the other image, and vice-versa:

$$x^* = argmax(x^T M x) \tag{3}$$

Here x^* must be a binary vector such that $x^*_{ia} = 1$ if feature i from one image is matched to feature a from the other image, and $x^*_{ia} = 0$ otherwise. As stated before, each feature from one image can match only one feature from the other, and vice-versa. This problem is NP hard, so most research on this topic focused mainly on developing efficient algorithms for finding approximate solutions, such as the graduated assignment (GA) [28], the spectral matching [38] or linear approximations [4] algorithms . However, as in graphical models, it is not only important to find the optimal solution, but it is also very important to have the right function to optimize. In this case the matrix M contains the second order potentials, such that $M(ia, jb)$ measures how well the pair of features (i, j) from one image agrees in terms of geometry and/or appearance with their matched counterparts (a, b) from the other image. Using the right function $M(ia, jb)$ is crucial for obtaining correct correspondences. Most work on this problem uses pairwise scores $M(ia, jb)$ that are designed manually. In the graphical models literature where the learning issue is addressed abundantly, but on learning the pairwise constraints for Graph Matching there is only one paper published [8], to the best of our knowledge. This is mainly because learning for graph matching is a harder problem than learning for graphical models, because the matching scores used in QAP are not normalized probability distributions.

The pairwise matching constraints $M_{ia;jb}(\mathbf{w})$ are functions of some parameter vector $\mathbf{w}$, which will ultimately influence the matching performance. The parameter vector depends on the actual implementation and it is irrelevant for our discussion

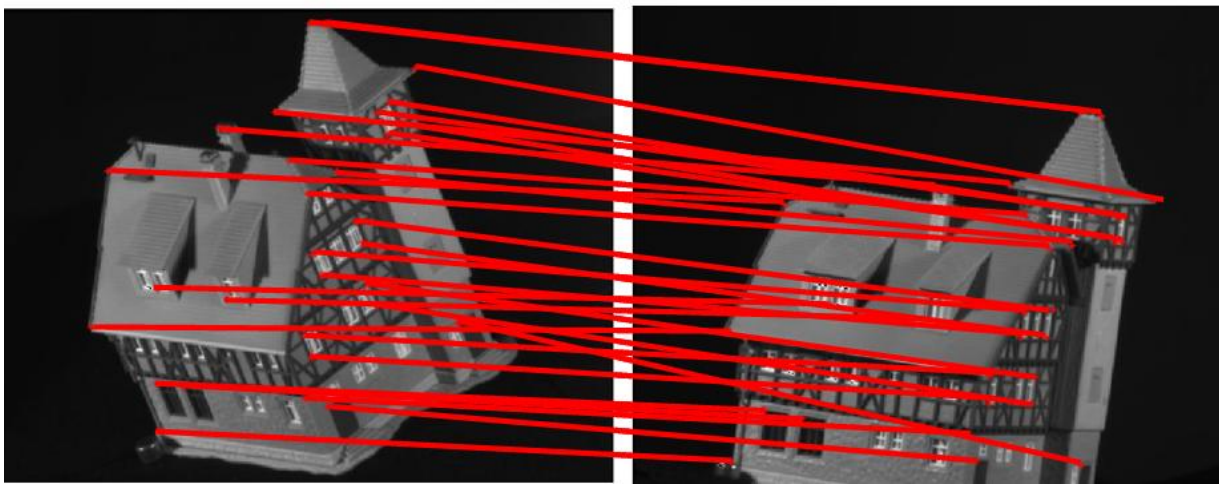

Figure 3. After learning the pairwise constraints for spectral matching, all the features in the first image are correctly matched to the features from the last image (House sequence)

here, since our optimization method is general and can handle any function $M_{ia;jb}(\mathbf{w})$ (unlike [8]). The performance of the matching algorithm is the function depending on $\mathbf{w}$ (which cannot be written in closed form) that we want to optimize during learning (it is similar to the one in [8]):

$$f(\mathbf{w}) = \sum_{i=1}^{m} \frac{n_c^{(i)}(\mathbf{w})}{n_t^{(i)}} \tag{4}$$

Here $n_c^{(i)}$ is the number of correct matches for image pair i, $n_t^{(i)}$ is the total number of ground truth possible matches (for the same image pair i) and $\mathbf{w}$ is the vector of parameters that define the pairwise scores. Then, the optimization problem is formulated as:

$$\mathbf{w}^* = argmax(f(\mathbf{w})) \tag{5}$$

To solve it we use our optimization algorithm for non-negative functions presented in Section 3.

2.3. Second Method for Learning the Pairwise Constraints for Graph Matching (work in progress)

We have shown how we can use our general optimization method for optimizing the true score function (relative number of correct matches). Next we present a different learning method that is specifically designed for our spectral matching algorithm. While this second approach is based on a local optimization scheme, it is in practice very efficient. In this case the cost function that we minimize is the sum of squared errors between the approximate principal eigenvector $\mathbf{v}$ (used by the spectral matching algorithm) and the ground truth normalized indicator vector $\mathbf{x_t}$, summed over all training image pairs.

$$J(\mathbf{w}) = \sum_{i=1}^{m} (\mathbf{v}^{(i)}(\mathbf{w}) - \mathbf{x_t}^{(i)})^2 \tag{6}$$

Notice that this error function is the soft version of the previous one and minimizing it is the same as maximizing the sum of correlations between the eigenvectors and the ground truth indicator solutions. We choose to minimize $J(\mathbf{w})$ by gradient descent:

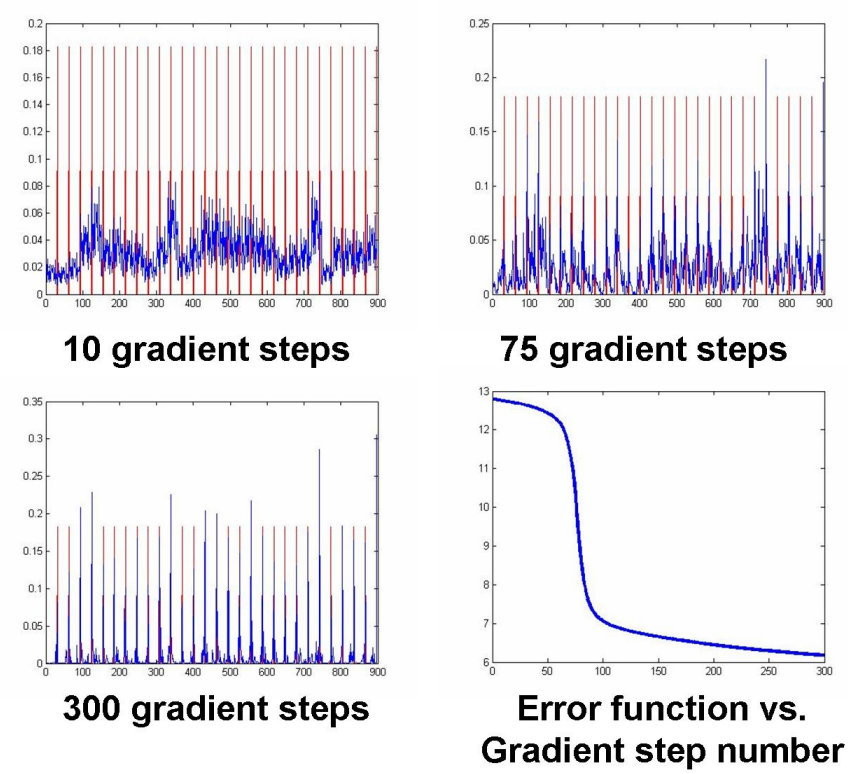

Figure 4. First three plots: the eigenvector (blue) vs the ground truth indicator vector during learning. After 300 gradient steps the eigenvector is very close to the ground truth (0.88 correlation). Last plot: the error monotonically decreases with each gradient step

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \sum_{i=1}^{m} (\mathbf{v}^{(i)}(\mathbf{w}) - \mathbf{x_t}^{(i)}) \nabla \mathbf{v}^{(i)}(\mathbf{w}) \tag{7}$$

The usual way of taking the partial derivatives of an eigenvector $\mathbf{v}$ is [15]:

$$\mathbf{v}' = (\lambda I - M)^{\dagger} (\lambda' I - M') \mathbf{v} \tag{8}$$

where

$$\lambda' = \frac{\mathbf{v}^T M' \mathbf{v}}{\mathbf{v}^T \mathbf{v}} \tag{9}$$

Since M is in our case a very large matrix (it can sometimes be up to 5000 by 5000) computing the required pseudo-inverse can be very tricky and expensive in practice. Below we present a new, much more efficient method for obtaining the derivatives of $\mathbf{v}$, which takes full advantage of the fact that v is the principal eigenvector of M (as opposed to any eigenvector).

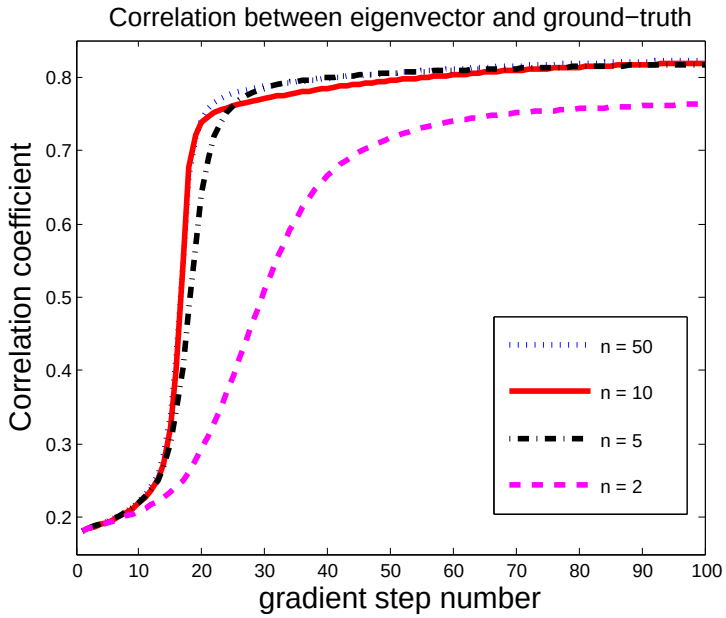

Figure 5. Experiments on our second method for learning for graph matching on the House sequence. The plots show the correlation coefficient between the eigenvector (used by spectral matching) and the ground truth solution vector for different number of recursive iterations n used to compute the approximative derivative of the eigenvector (the plots represent are averages over all our experiments). Notice that for a very wide range of n (even for n as small as 5) the learning method behaves in the same way, returning the same result. From a qualitative point of view even small n are equally good. However, from a computational point of view, the time increases linearly with n

In our implementation, we obtain $\mathbf{v}$ by using the power method for a fixed number n of iterations (maximum 100, but in most cases $10-20$ are enough). So $\mathbf{v}$ is in fact not the true eigenvector, but an approximation of it: $\mathbf{v} = \frac{M^n\mathbf{1}}{\sqrt{(M^n\mathbf{1})^T(M^n\mathbf{1})}}$. We use this formula to take partial derivatives of $\mathbf{v}$:

$$\mathbf{v}' = \frac{(M^n\mathbf{1})'(\|M^n\mathbf{1}\|) - M^n\mathbf{1}/\|M^n\mathbf{1}\|((M^n\mathbf{1})^T(M^n\mathbf{1})')}{\|M^n\mathbf{1}\|^2} \tag{10}$$

We notice that in order to obtain the derivative of $\mathbf{v}$, we first need to compute the derivative of $M^n\mathbf{1}$, which can be easily obtained recursively:

$$(M^n\mathbf{1})' = M'(M^{n-1}\mathbf{1}) + M(M^{n-1}\mathbf{1})' \tag{11}$$

Notice that, in this formulation, the approximation $\mathbf{v}$ to the true eigenvector is differentiable whenever the elements of M are differentiable.

Table 1. Matching performance on the hotel and house datasets. In the first three columns the same 5 training images from the House dataset were used. For the fourth column 106 training images from the House sequence were used. SC stands for Shape Context

Datasest	Ours (I) No SC (5)	Ours (II) No SC(5)	[8] SC (5)	[8] SC (106)
House	99.8%	99.8%	$< 84\%$	$\approx 95\%$
Hotel	94.8%	94.8%	$< 87\%$	$< 90\%$

In Figure 4 we show how the approximate eigenvector $\mathbf{v}$ indeed approaches the ground truth indicator vector (for a specific image pair) as the parameters are learned using gradient descent. Even though the method is local, in practice it is very fast and effective.

To get a better feeling of how much more efficient our method is than [15], we performed some quantitative experiments to measure empirically the relative computational cost between them. To implement [15] in Matlab one can use the function $pinv$, which computes the pseudo-inverse of a matrix. Also, one can use a more efficient alternative, the Matlab matrix division operator, but that one is more efficient only if the matrix $(\lambda I - M)$ is non-singular. Unfortunately, in our experiments on matching, the matrix was most of the time singular, so the implementation of [15] became very inefficient. In turn, our method works with singular matrices in the same fashion as with non-singular ones, and it also has the option of tuning the number of iterations n for increasing efficiency. If our method for $n = 10$ takes 1 unit of time to compute a gradient step then the one in [15] takes over 60 times longer if the matrix $(\lambda I - M)$ is non-singular (which was a rare case) and over 1500 times longer when the matrix was singular (which happened much more often). Both methods perform equally well qualitatively. In turn, our method is linear in n and as Figure 5 shows, qualitatively insensitive to n (it works equally well with n as low as 5). These results are averaged over all gradient steps, in all our experiments on 900 by 900 matrices.

Experiments We perform experiments on two tasks that are the same as the ones in [8]. We used exactly the same image sequences both for training and testing [32, 31], and the same features, which were manually selected by [8]. For solving the quadratic assignment problem we used the spectral matching algorithm [38], instead of the the graduated assignment [28], because it is faster. The goal of these experiments were not to directly compare the two learning algorithms, but rather to show that our algorithm is suitable for this problem also. The algorithm in [8] is specifically designed for a certain class of score functions, but both our algorithms can work with any pairwise score $M(ia, jb)$. The algorithm in [8] optimizes a convex upper bound to the cost function, while in our case we attempt to optimize directly the score/error functions.

The type of pairwise scores that we want to learn is:

$$M_{ia;jb} = \mathbf{exp}(w_0 + w_1 \frac{|d_{ij} - d_{ab}|}{|d_{ij} + d_{ab}|} + w_2 |\alpha_{ij} - \alpha_{ab}|) \tag{12}$$

Here d_{ij} and d_{ab} are the distances between features (i, j) and (a, b) respectively, while α_{ij} and α_{ab} are the angles between the X axis and the vectors $\vec{ij}$ and $\vec{ab}$, respectively. As in [8] we first obtain a Delaunay triangulation and allow non-zero pairwise scores $M_{ia;jb}$ if and only if both (i, j) and (a, b) are connected in their corresponding triangulation. The pair-wise scores we work with are different than the ones in [8] because we wanted to put more emphasis on the second order scores. The authors of [8] make the point that after learning there is no real added benefit for using the second order scores, and that linear assignment using appearance only terms (based on Shape Context) suffices. We make the counter argument by showing that in fact the second order terms are much stronger once distance and angle information is used (which they did not use). Our performance is significantly better even when we do not use any appearance terms (Figure 1). With only 5 training images used, we obtain almost 100% accuracy, more than 15% better than what they obtain using the same exact training and testing pairs of images. Our belief that the second order terms are much more powerful than the local appearance terms is a common theme throughout this thesis, as can also be seen (to a larger extent) in Section 4.

3. Smoothing-based Optimization (work in progress)

Some of the problems that we want to solve in this thesis, such as learning for graph matching, require the optimization of complex, non-linear, possibly non-differentiable functions. In fact, there are many such problems in computer vision. Here we propose a novel, efficient method for such tasks. Probably the two most popular algorithms used in such cases are Markov Chain Monte Carlo (MCMC) and Simulated Annealing (and their variants). While these algorithms have global optimality

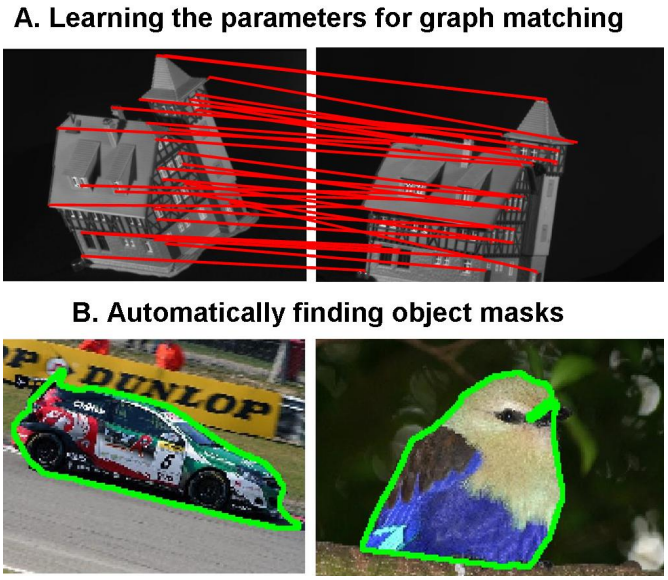

Figure 6. Many different vision tasks invlove the optimization of complex functions: A. Learning the parameters for graph matching (Quadratic Assignments Problem) B. Automatically extracting object masks, given an input bounding box

properties, in practice they lack efficiency as they require a large number of samples. Variants of MCMC are commonly used in various vision applications such as segmentation [66], object recognition [67] and human body pose estimation [64]. Other difficult optimization problems such as learning graph matching [8] are approached by optimizing an upper bound of the original cost function.

We propose an efficient method for such optimization problems. One of our main ideas is that searching for the global maximum through the scale space of a function [42] is equivalent to looking for the optimum of the original function, but with the added benefit that we have to avoid fewer local optima. Our method works with any non-negative, possibly non-smooth function, and requires only the ability of evaluating the function at any specific point. In order to better explain our algorithm we first discuss the inspirations that stand behind it.

3.1. First Motivation: Smoothing for Optimization

There are two main ideas that inspired the design of our algorithm. Even though at a first glance they seem unrelated, their connection becomes obvious once we explicitly present our algorithm.

Functions with many local optima have been always a problem in optimization. Most optimization algorithms are local and prone to get stuck in local optima. Compared to the area of convex optimization, there are relatively much fewer choices of algorithms that attempt to find the global optimum, or even an important optimum, of highly nonlinear and non-differentiable functions. Algorithms such as Graduated Non-convexity [5] address the non-convexity problem by modifying the original function and adding to it a large convex component such that the sum will also be convex. Starting from an initial global optimum, and tracking it as the influence of the convex component is slowly reduced, the procedure hopes to finally converge to the original global optimum. Our idea of smoothing is similar: the more we smooth a function (the larger the variance of the Gaussian kernel) the less local optima the function will have. Instead of corrupting the original function by adding to it a foreign convex function such as it is the case with Graduated Non-convexity [5], blurring uses the function's own values to obtain a similar and most probably a better effect (Figure 7). There is only one caveat with the smoothing approach. Since the Gaussian kernel has infinite support, for smoothing the function at a single point one would have to visit the entire space, and would thus find the global optimum by exhaustive search! So even though the idea sounds interesting, it is in fact impossible to apply in its pure form. Fortunately, as we will show later, in practice things are not even nearly as impractical as they might seem. But before we focus on the practical aspects of our algorithm, let us first convince ourselves that we have the theory to support it.

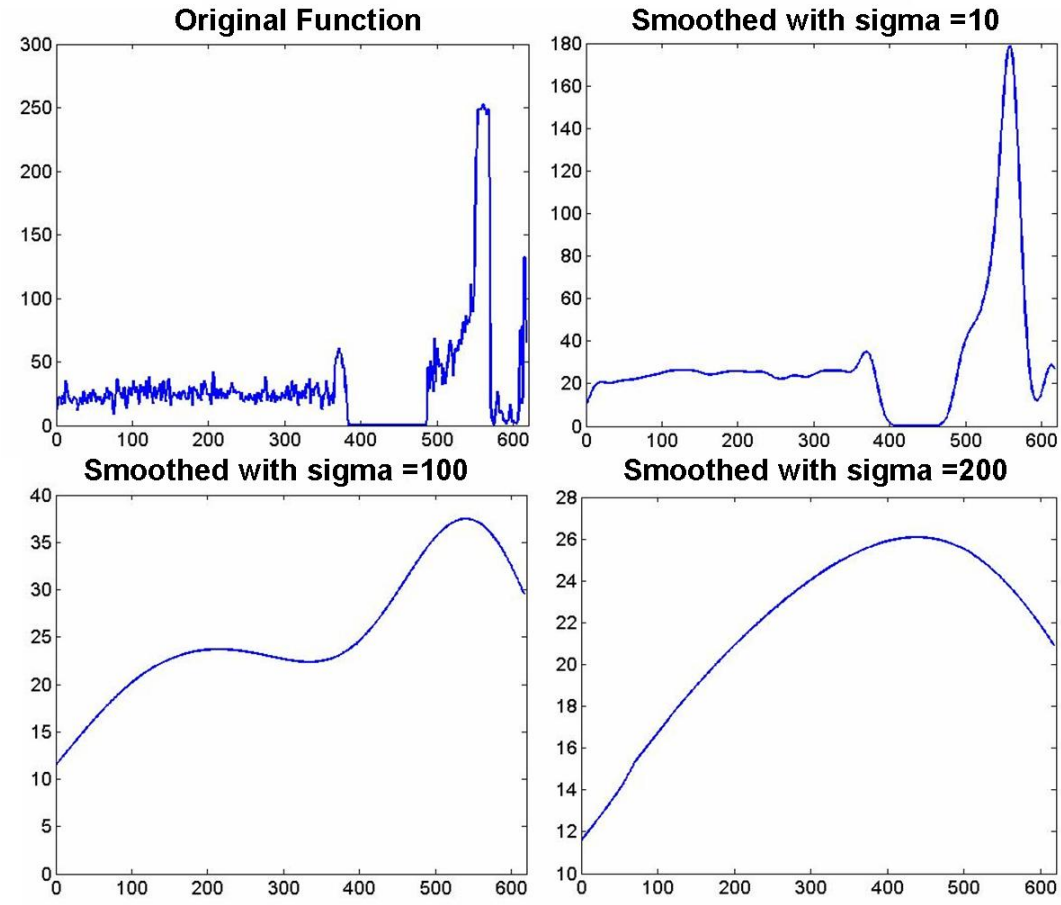

Figure 7. At higher levels of smoothing less and less local optima survive. Even for a relatively small variance (=10), most of the noisy local optima disappear while the significant ones survive

The results from scale space theory [42] show that for most functions local optima disappear very fast as we increase the variance of the Gaussian blurring. Also, the local optima that survive at higher levels of smoothing can be usually traced back to significant local optima in the original function. Moreover, any nonnegative function with compact support will end up with a single global maximum for a large enough variance of smoothing [43].

In Figure 7 the original function is extremely wiggly and any gradient based technique would immediately get stuck in a local maximum. However, as soon as we blur the function with a relatively small sigma, most noisy local optima vanish. Finally, for a large enough sigma there is only one global optimum which could be traced back to the original global optimum. The unique global maximum of a blurred function cannot always be traced back to the original global optimum, nevertheless, for most functions, it will be traced back to a *significant* local optimum, which constitutes an important progress compared to local optimization techniques. So, if we could somehow have access to the values of our smoothed function in the neighborhood of our current position, we would know where to move next to approach a more important optimum.

3.2. Second Motivation: Updating our Knowledge

Our second motivation, which is seemingly unrelated to the smoothing idea, is to represent our knowledge of where the optimum of our complex function is with a multidimensional Gaussian. At any point in time, we want to evaluate the complex function at points where this Gaussian (which represents our current knowledge) has high probability mass and use those evaluations for updating our current Gaussian in a way that will get us closer to the optimum we are looking for. In Figure 8 we present this idea by running our algorithm on a one dimensional function. The function is sampled in a region where the Gaussian has high probability. Based on those evaluation the variance can increase or decrease. At the final iteration we are indeed very close to the true optimum and our search (sampling) space is minimal (very small variance).

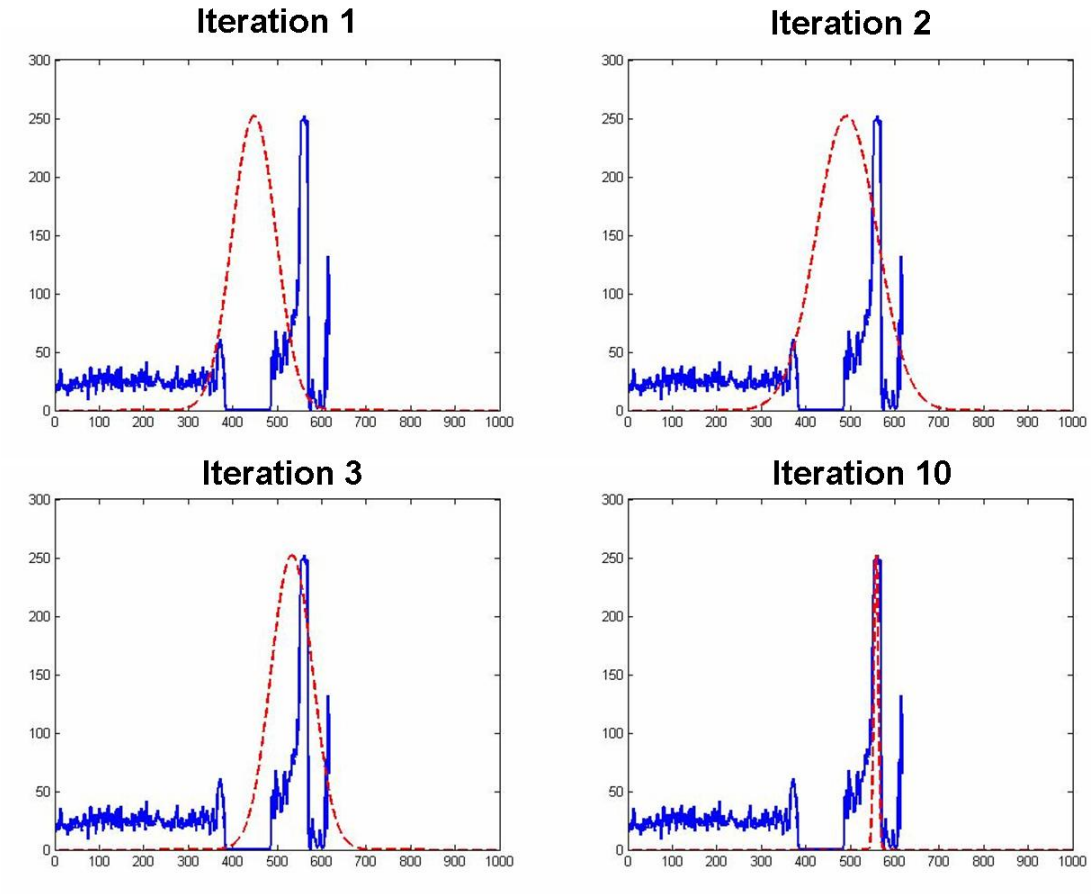

Figure 8. Refining our knowledge (red dashed line) about the optimum of the function we want to optimize (blue line). At each iteration the mean of the Gaussian represents our current guess, while its variance the uncertainty. By the tenth iteration we are very close to the true optimum and also very certain about where it is (very small variance)

This idea is related to the Cross Entropy Method for optimization [57]. Even though technically very different, both ideas are based on sampling from a distribution that is sequentially refined until it converges around the optimum. Other more distantly related work includes importance sampling algorithms for Bayesian Networks [11, 61], where a function, that is relatively inexpensive to draw samples from, is sampled in order to estimate marginals (expressed as integrals that are hard to compute exactly). The samples obtained are used for continuously refining the sampling function in order to obtain better and better estimates of these marginals.

3.3. Algorithm

The two motivations described above are seemingly unrelated, but the connection between them becomes clear once we look in detail at our algorithm. Before describing the algorithm, we present the following theorem on which it is based:

Theorem 1: Let $f : \mathbb{R}^n \rightarrow \mathbb{R}+$ be a non-negative multi-dimensional function. Let its scale space function (its smoothed version) be defined as $F(\mu, \sigma^2 I) = \int g(x; \mu, \sigma^2 I) f(x) dx$, where g is a multidimensional Gaussian (of dimension n) with mean μ and covariance matrix $\sigma^2 I$. Given the pair $(\mu^{(t)}, \sigma^{(t)})$ at time step t, we define $(\mu^{(t+1)}, \sigma^{(t+1)})$, at the next time step $t+1$, by the following update rules (i refers to dimension indices, and $g^{(t)}(x) = g(x; \mu^{(t)}, \sigma^{(t)})$):

1. $\mu^{(t+1)} = \frac{\int x g^{(t)}(x) f(x) dx}{\int g^{(t)}(x) f(x) dx}$
2. $\sigma^{(t+1)} = \sqrt{\frac{1}{n} \frac{\int (\sum_{i=1}^{n} (x_i - \mu_i)^2) g^{(t)}(x) f(x) dx}{\int g^{(t)}(x) f(x) dx}}$

Then, the following conclusions hold:

a). $F(\mu^{(t+1)}, \sigma^{(t)}) \geq F(\mu^{(t)}, \sigma^{(t)})$

b). $F(\mu^{(t)}, \sigma^{(t+1)}) \geq F(\mu^{(t)}, \sigma^{(t)})$

Proof: see Sections 7.1 and 7.2.

This theorem basically states that the update steps 1 and 2 represent growth transformations [2, 35] for the function F. They provide a specific way of updating the Gaussian which represents our knowledge about the optimum at any specific time step. This gives us the connection to our second motivation. Also, the updating steps are in the direction of the gradient of the scale space function F, and thus provide us with a way of traveling not only through the original search space but also through scale. This gives us the link to the first motivation based on smoothing.

The smaller σ the more F approaches the original function f. It is clear that the global optimum of F is the same as the global optimum of f. So optimizing f is practically equivalent to optimizing F. The difference is, as we discussed in Section 3.1, that in F it is much easier to avoid the local optima. The theorem above is the basis of our method. Probably its most interesting feature (as we found in our experiments in Section 3.4) is its ability of updating automatically σ, which grows if we need to escape from valleys and shrinks if we reach the top of an important mountain on the function's surface (we know from scale space theory that σ will indeed shrink once we are close to such optima). As mentioned before, the main issue of this idea is how to compute the update steps 1 and 2.

Since they cannot be computed exactly (because we can only evaluate f at a given point and do not want to search the whole space) we will resort to methods commonly used for estimating integrals. One well-known possibility is the Monte Carlo Integration method [48], another one is by Gaussian quadrature [20], which could be more efficient in practice in spaces of lower dimensions, because it requires less function evaluations.

Our algorithm is an implementation of the above theorem. Below we present the version of the algorithm that uses Monte Carlo Integration sampling, but, as we mentioned above, Gaussian quadratures could also be used and are often more efficient in practice:

1. Start with initial values of $\mu^{(0)}$ and $\sigma^{(0)}$, set t = 0
2. Draw samples $s_1, s_2, ..., s_m$ from the normal distribution $N(\mu^{(t)}, (\sigma^{(t)})^2 I)$
3. Set: $\mu^{(t+1)} = \frac{\sum_{k=1}^{m} s_k f(s_k)}{\sum_{k=1}^{m} f(s_k)}$
4. Set: $\sigma^{(t+1)} = \sqrt{\frac{1}{n} \frac{\sum_{k=1}^{m} (\sum_{i=1}^{n} (s_i^{(k)} - \mu_i^{(t)})^2) f(s_k)}{\sum_{k=1}^{m} f(s_k)}}$
5. if $\sigma^{(t+1)} < \epsilon$ stop.
6. t = t+1. Go back to step 2

Notice that we apply the update steps for μ and σ at the same time, even though our theorem gives theoretical guarantees only if we apply them sequentially. Even if that is of theoretical concern, we found that in practice this does not hurt the performance, but on the contrary, it actually makes the algorithm more efficient.

Our algorithm is also related to the Mean Shift algorithm [14], [13], since both algorithms can adapt the mean and the kernel size. However, the difference between the two algorithms is substantial. Mean Shift does not work with f directly, but with data samples (drawn from some unknown f, that cannot be evaluated at a given point) and uses a kernel k to weigh the samples. Therefore, Mean Shift cannot be used for optimization of arbitrary functions. In our case, we cannot draw samples from f, because the functions we want to optimize are very complex, so instead we draw samples from our Gaussian kernel g and use instead the specific values of f to weigh these samples. It seems like the two algorithms are complementary to each other.

3.4. Experiments on Synthetic Data

In the first set of experiments we compare the performance of our algorithm to two well established methods commonly used in complex optimization problems: Markov Chain Monte Carlo (MCMC) and Simulated Annealing (SA). While MCMC is not specifically designed for optimization, it has been successfully used for this purpose in the vision literature. SA on the other hand has guaranteed optimality properties in a statistical sense. Given enough samples, both MCMC and SA are

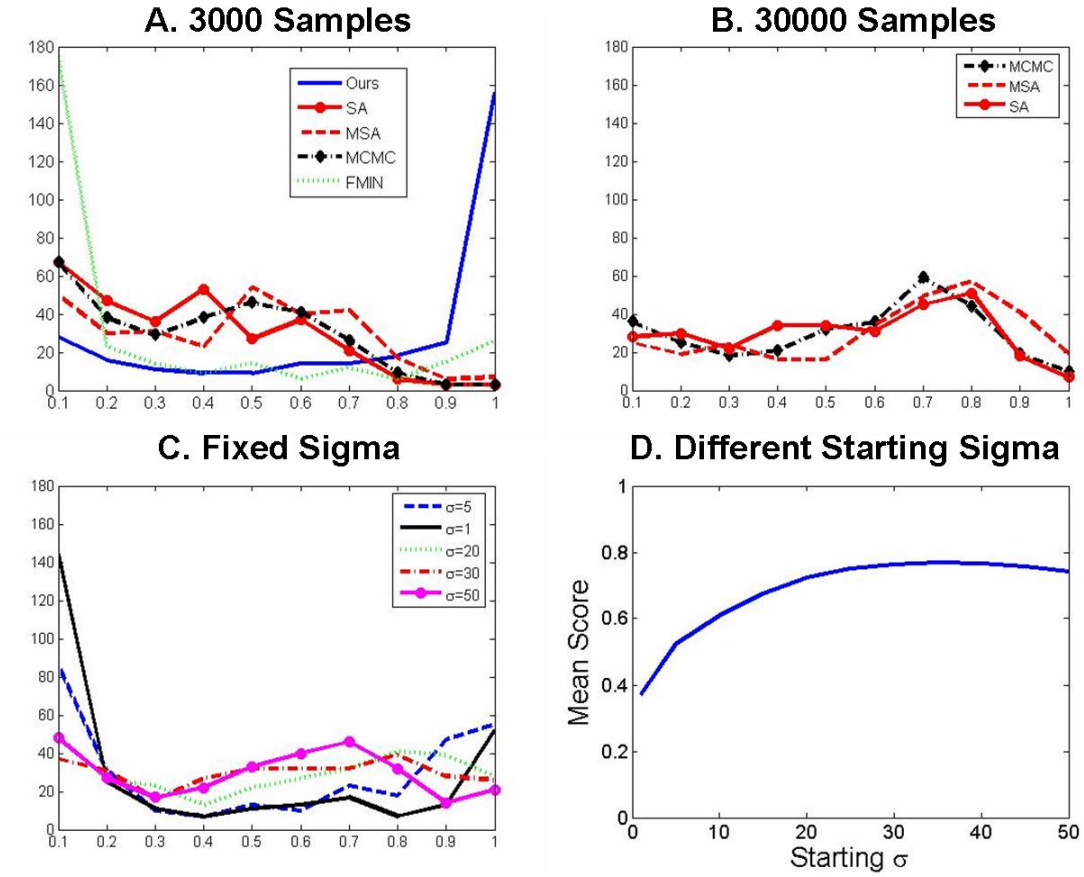

Figure 9. Except for D, all are plots showing histograms of the scores obtained over 300 experiments A. all algorithms can run for a maximum of 3000 function evaluations (samples). B. the algorithms were run for a maximum of 30000 function evaluations (samples). C. Our algorithm, without the ability of changing its initial covariance matrix, for different initial $\sigma^{(0)}$. D. Our algorithm, with different starting $\sigma^{(0)}$ (shown on the X axis in degrees), being able to adapt it. The mean scores obtained over 300 experiments is shown in plot D.

guaranteed to find the global maximum, but often the number of samples required is very large, thus neither method is particularly efficient.

For this experiment we used synthetic data. Given a rectangle of known dimensions, known location (in 3D) of its center, we rotate it in 3D by $\theta_t = (\theta_x, \theta_y, \theta_z)$ and obtain its projection on the XY plane as a binary mask I_{θ_t}. The algorithms are provided only with this mask, their task being of finding the θ^* which maximizes the overlap between the mask given I_{θ_t} and I_{θ^*}. More precisely, the score that needs to be maximized is:

$$f(\theta^*) = \left(\frac{N(I_{\theta_t} \cap I_{\theta^*})}{N(I_{\theta_t} \cup I_{\theta^*})}\right)^{10} \tag{13}$$

Here $N(I_{\theta_t} \cap I_{\theta^*})$ is the area of the intersection of the two masks, and $N(I_{\theta_t} \cup I_{\theta^*})$ is the area of their union. We raised the score function to the 10^{th} power because it is too flat otherwise.

This score function is periodical with infinitely many local and, of course, global maxima. The global maxima have obviously the known value of 1. The resolution of the image mask is such that an exhaustive search of the angles space in the intervals $[0, 90]$ would require around 10^{10} samples (the function is sensitive to changes in angles as little as 0.02 degrees).

In Figure 9 , plot A, we compare our method against MCMC, standard Simulated Annealing (SA), Metropolis-SA (MSA), and Nelder-Mead method as the *fminsearch* (FMIN) function from the Matlab optimization toolbox (for *fminsearch* we used as the cost function $1 - f(\theta)$ since it is a minimizing procedure, but the results showed here were only in terms of $f(\theta)$). All algorithms except *fminsearch* are limited to a maximum of 3000 function evaluations (samples). The plot shows the histogram of the maximum scores obtained over 300 experiments. Each algorithm ran on the same problems, with the same

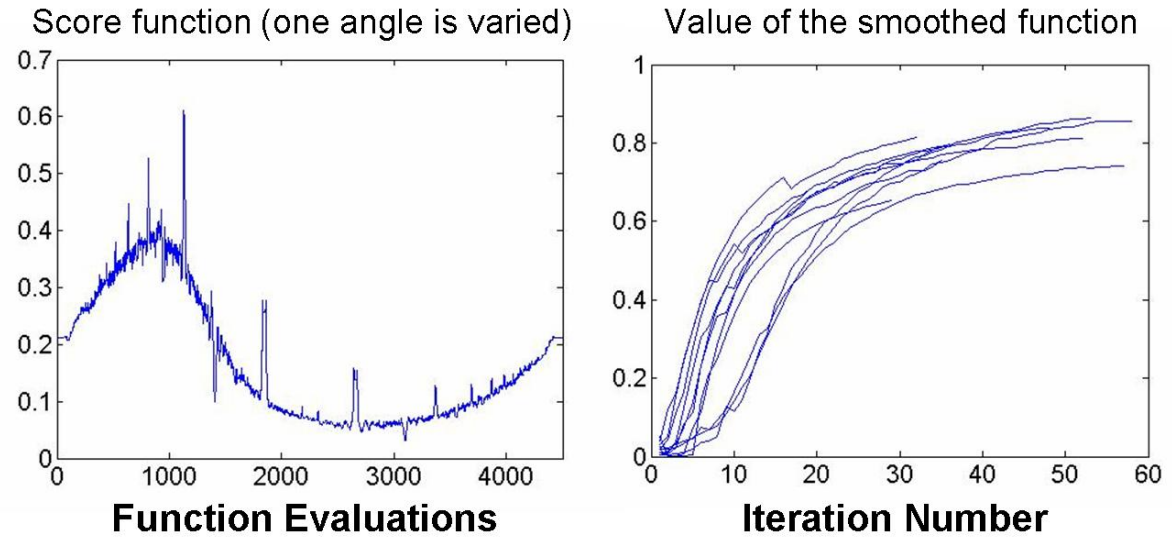

Figure 10. Left: the score function evaluated every 0.02 degrees of θ_z in $[0, 90]$. The other angles were kept constant. Notice how wiggly the function is due to the fact that the mask is discrete in practice. Right: the value of the smoothed function for each iteration of our algorithm. Notice that it is mainly monotonic which agrees with the theory. Sometimes it fails, but this happens only because the updating steps are approximations to the ones in the theorem. The plots belong to the first 10 random experiments

starting points and ground truth θ_t. For each experiment, both the starting point and the ground truth were chosen randomly in the degree space $[0, 360]$ (in each dimension of θ). For MCMC, SA and MSA we chose the variance of the proposal distribution that gave the best performance. The worst performer was *fminsearch*, as expected, since it is a local method and the score function has a lot of local optima (see Figure 10). Our algorithm outperformed all the others (Figure 9. Even when we allowed MCMC, SA and MSA to run for 10 times more samples, their performance was still inferior to ours (plot B). Of course, for a sufficiently large number of samples MCMC, SA and MSA will always find the right solution, but the point of this experiment was to consider the efficiency of the different algorithms.

In the next experiment (plot C) we wanted to emphasize that one of the main strengths of our algorithm is its capacity to change the covariance of its sampling distribution. On the one hand we see that if we keep this covariance fixed its performance degrades considerably, for a wide range of σ(the starting covariance matrix was diagonal, with diagonal elements equal to σ) (Figure 9, plot C). On the other hand, if we allow this covariance to change, the starting value of σ is not very relevant (Figure 9, plot D). Except when the starting σ is very small (< 5 degrees), the mean score obtained over the same 300 experiments does not vary much. From this we can draw the conclusion that our algorithm is most often able to adapt its covariance correctly during the search, regardless of its starting value.

4. Object Recognition without Grouping

4.1. Introduction

In this Section we describe an algorithm for object category recognition (not using grouping cues) [39], which is based on the observation that for a wide variety of common object categories, it is the shape that matters more than their local appearance. For example, it is the shape, not the color or texture, that enables a plane to fly, an animal to run or a human hand to manipulate objects. Many categories are defined by their function and it is typically the case that function dictates an object's shape rather than its low level surface appearance.

Although shape alone was previously used for category recognition [52, 24] we demonstrate for the first time (to the

best of our knowledge) that simple geometric relationships with no local appearance features can be used successfully in a semi-supervised setting.

The importance of shape for object recognition was previously emphasized by Belongie *et al*. [3] and later by Berg *et al*. [4]. In [3] the shape context descriptor is introduced, which is a local, unary (one per feature) descriptor that loosely captures global shape transformations, but, by nature, is also sensitive to background clutter. Later, Berg et al. used second-order geometric relationships between features, but the main focus was still on the local geometric information (using the geometric blur descriptor). In this thesis we emphasize the importance of pairwise geometric relationships by dropping the local information and using a more extensive set of parameters (see Section 11) for representing the pairwise geometric relationships ([4] uses only the pairwise distance and a single angle). Also, unlike [4], we learn these parameters in order to better capture the space of pairwise deformations (Section 4.3).

The idea of using geometric constraints with shape features (e.g., edges and surfaces) for object recognition dates back to 1980's . For instance, Grimson and Lozano Perez [19] propose matching using an interpretation tree that enforces a global geometric transformation and requires a combinatorial search space. Our work focuses solely on pairwise geometric relationships that gives us flexibility in matching deformable objects. We address the combinatorial problem by employing a fast approximate algorithm.

There are two popular trends in object recognition. The first trend is to formulate it as a matching problem, and use either the nearest neighbor approach [4] or the SVM classifier [29]. The second trend is to formulate it as an inference problem and use the machinery of graphical models [22, 23, 10, 17]. In our work we use ideas from matching using second-order geometric relationships [4, 38], but, unlike current approaches in matching, we build a category model that is a compact representation of the training set (similar to approaches using Conditional Random Fields).

We integrate several training images into a single abstract shape model that captures both common information shared by several training images as well as useful information that is unique to each training image. We also automatically discover and remove the irrelevant clutter from training images, keeping only the features that are indeed useful for recognition. This gives us a more compact representation, which reduces the computational and memory cost, and improves generalization.

The use of second-order geometric relationships enables our algorithm to successfully overcome problems often encountered by previous methods from the literature:

1. During training our algorithm is translation invariant, robust to clutter, and does not require aligned training images. This is in contrast with previous work such as [4, 51, 49, 25].

2. We efficiently learn models consisting of hundreds of fully interconnected parts (capturing both short and long range dependencies). Most previous work handles models only up to 30 sparsely inter-connected parts, such as the star shaped [23, 16], k-fan, [17] or hierarchical models [21, 6]. There has been work [10] handling hundreds of parts, but their model is not translation invariant and each object part is connected only to its k-nearest neighbors.

3. We select features based on how well they work together as a team rather than individually (as it is the case in [17, 51]). This gives us a larger pool of very useful features, which are discriminative together, not necessarily on an individual basis.

We formulate the problem as follows: given a set of negative images (not containing the object) and weakly-labeled positive images (containing an object of a given category somewhere in the image), the task is to learn a category shape model (Section 4.3) that can be used both for the *localization* and *recognition* of objects from the same category in novel images (Section 11). This problem is challenging because we do not have any prior knowledge about the object's location in the training images. Also these images can contain a substantial amount of clutter that is irrelevant to the category we want to model. All we know at training time is that the object is somewhere in the positive training images and nowhere in the negative ones.

4.2. The Category Shape Model

The category shape model is represented as a graph of interconnected parts (nodes) whose geometric interactions are modeled using pairwise potentials inspired from Conditional Random Fields (CRF) [37, 36] (see Section 4.2.1). The nodes in this graph are fully interconnected (they form a clique) with a single exception: there is no link between two parts that have not occurred together in the training images. These model parts have a very simple representation: they consist of sparse, abstract points together with their associated normals. Of course we could add local information in addition to their normals, but our objective is to assess the power of the geometric relationships between these simple features. We represent

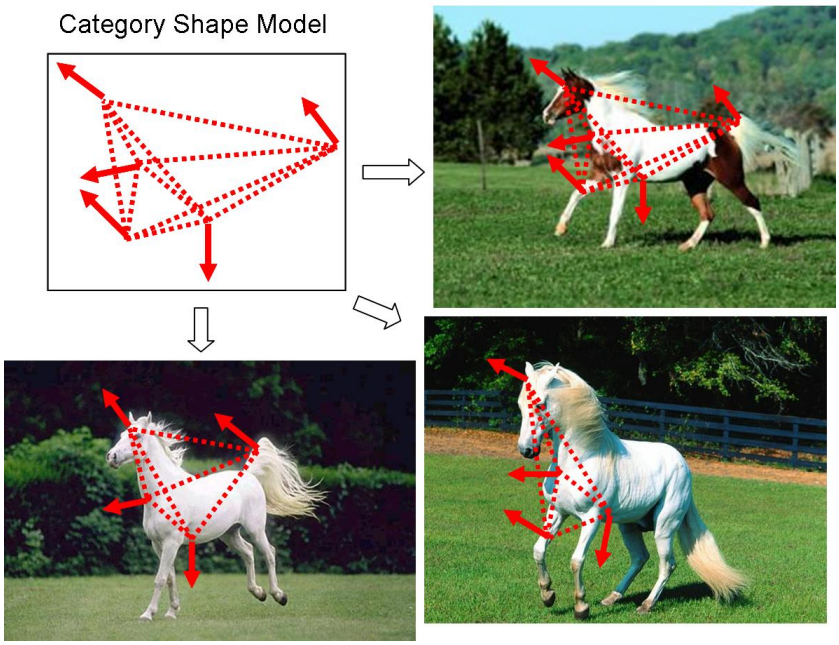

Figure 11. The model is a graph whose edges are abstract pairwise geometric relationships. It integrates generic configurations common to most objects from a category as well as more specific configurations that capture different poses and aspects

the pairwise relationships by an over-complete set of parameters (see Section 4.2.1). The parts as well as their geometric relationships are learned (see Section 4.3) from actual boundary fragments extracted from training images as described in [39].

Our model is a graph whose edges are abstract pairwise geometric relationships. It is a compact representation of a category shape, achieved by sharing generic geometric configurations common to most objects from a category and also by integrating specific configurations that capture different aspects or poses (Figure 11). In Section 4.3 we discuss and also exemplify (Figure 14) the ability of our learning stage to reuse common configurations and also integrate new ones.

In order to explain in detail the pairwise geometric relationships we start with the *localization* problem, that is matching the object parts to the image features.

4.2.1 Object Localization

We define the object localization problem as finding which feature in the image best matches each model part. We formulate it as a quadratic assignment problem (QAP), due to our use of second-order relationships. The matching score E is written as:

$$E_x = \sum_{ia;jb} x_{ia} x_{ib} G_{ia;jb} \tag{14}$$

Here x is an indicator vector with an entry for each pair (i, a) such that $x_{ia} = 1$ if model part i is matched to image feature a and 0 otherwise. With a slight abuse of notation we consider ia to be a unique index for the pair (i, a). We also enforce the mapping constraints that one model part can match only one model feature and vice versa: $\sum_i x_{ia} = 1$ and $\sum_a x_{ia} = 1$.

The *pairwise potential* $G_{ia;jb}$ (terminology borrowed from graphical models) reflects how well the parts i and j preserve their geometric relationship when being matched to features a, b in the image. Similar to previous approaches taken in the context of CRFs [36] we model these potentials using logistic classifiers :

$$G_{ia;jb} = \frac{1}{1 + exp(-\mathbf{w}^T \mathbf{g}_{ij}(a, b))} \tag{15}$$

Here $\mathbf{g}_{ij}(a, b)$ is a vector describing the geometric deformations between the parts (i, j) and their matched features (a, b). We now explain in greater detail the type of features used and their pairwise relationships. As mentioned already, each object

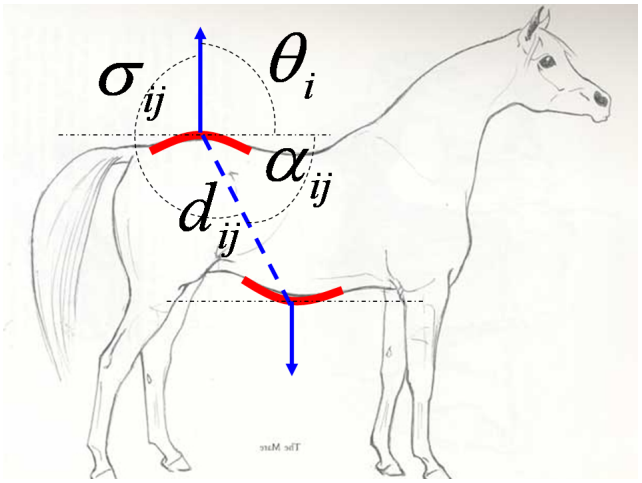

Figure 12. Parameters that capture the pair-wise geometric relationships between object parts

part can be seen as an abstract point and its associated normal (with no absolute location). For a pair of model parts (i, j) we capture their translation invariant relationship in the vector $\mathbf{e}_{ij} = \{\theta_i, \theta_j, \sigma_{ij}, \sigma_{ji}, \alpha_{ij}, \beta_{ij}, d_{ij}\}$, where d_{ij} represents the distance between them, β_{ij} is the angle between their normals and the rest are angles described in Figure 12.

The same type of information is extracted from input images, each image feature corresponding to a point sampled from some boundary fragment extracted from that image (see Section **??**). We consider a similar pairwise relationship $\mathbf{e}_{ab}$ for the pair (a, b) of image features that were matched to (i, j). Then we express the pairwise geometric deformation vector as $\mathbf{g}_{ij}(a, b) = [1, \epsilon_1^2, ..., \epsilon_7^2]$, where $\epsilon = \mathbf{e}_{ij} - \mathbf{e}_{ab}$. Notice that the geometric parameters $\mathbf{e}_{ij}$ form an overcomplete set of values, some highly dependent on each other. Considering all of them becomes very useful for geometric matching and recognition because it makes $G_{ia;jb}$ more robust to changes in the individual elements of $\mathbf{g}_{ij}(a, b)$.

In order to localize the object in the image, we find the assignment $\mathbf{x}^*$ that maximizes the matching score E (written in matrix notation by setting $\mathbf{G}(ia; jb) = G_{ia;jb}$):

$$\mathbf{x}^* = argmax(\mathbf{x}^T \mathbf{G} \mathbf{x}) \tag{16}$$

For one-to-one constraints (each model part can match only one image feature and vice-versa) this combinatorial optimization problem is known as the quadratic assignment problem (QAP). For many-to-one constraints it is also known in the graphical models literature as MAP inference for pairwise Markov networks. In general, both problems are intractable. We enforce the one-to-one constraints and use the spectral matching algorithm presented earlier [38].

4.2.2 Discriminative Object Recognition

In the previous section we have presented how we localize the object by efficiently solving a quadratic assignment problem. However, this does not solve the recognition problem, since the matching algorithm will return an assignment even if the input image does not contain the object. In order to decide whether the object is present at the specific location $\mathbf{x}^*$ given by our localization step, we need to model the posterior $P(C|\mathbf{x}^*, D)$ (where the class $C = 1$ if the object is present at location $\mathbf{x}^*$ and $C = 0$ otherwise). Modeling the true posterior would require modeling the likelihood of the data D given the background category (basically, the rest of the world), which is infeasible in practice. Instead, we take a discriminative approach and try to model this posterior directly as described below.

We consider that $P(C|\mathbf{x}^*, D)$ should be a function of several factors. First, it should depend on the quality of the match (localization) as given by the pairwise potentials $G_{ia;jb}$ for the optimal solution $\mathbf{x}^*$. Second, it should depend only on those model parts that indeed belong to the category of interest and are discriminative against the negative class. It is not obvious which are those parts, since we learn the model in a semi-supervised fashion . For this reason we introduce the *relevance* parameter r_i for each part i (in Section 4.3 we explain how it is learned), which has high value if part i is discriminative against the background, and low value otherwise. We approximate the posterior with the following logistic classifier:

$$S(\mathbf{G}_o, \mathbf{r}) = \frac{1}{1 + exp(-q_0 - q_1 \sigma(\mathbf{r})^T \mathbf{G_o} \sigma(\mathbf{r}))} \tag{17}$$

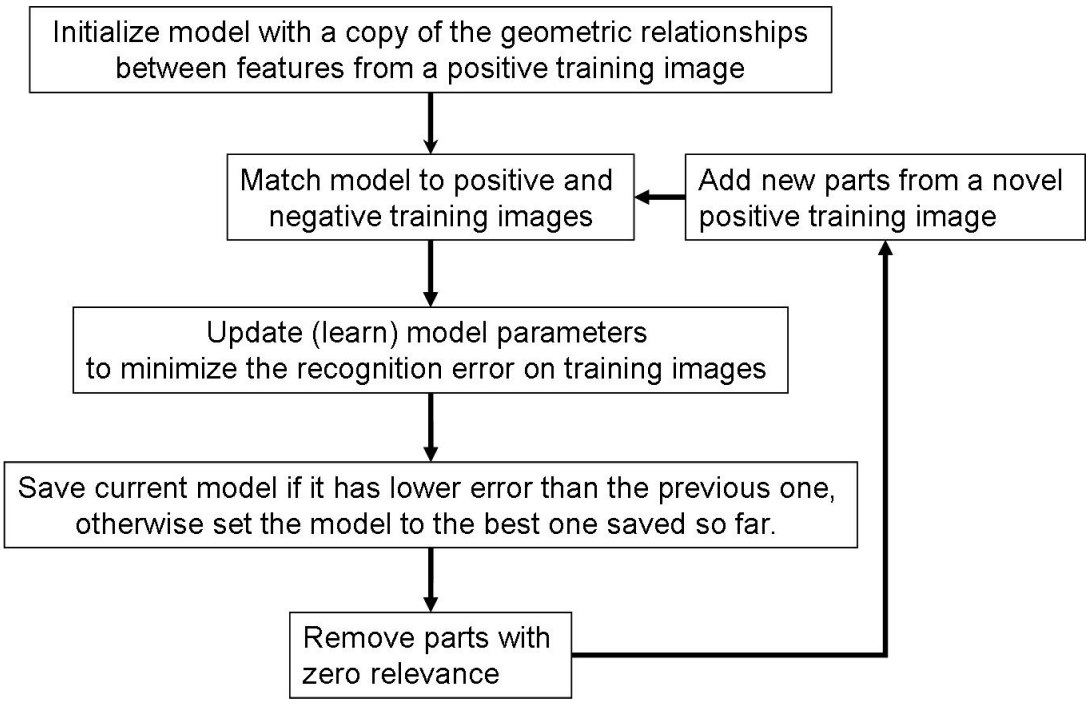

Figure 13. Learning algorithm overview

The matrix $\mathbf{G_o}(i,j) = G_{ia^*;jb^*}$ contains all the pairwise potentials for the optimal localization $\mathbf{x}^*$. In the equation above, each pairwise potential G_{ij} is weighted by the product $\sigma(r_i)\sigma(r_j)$, where:

$$\sigma(r_i) = \frac{1}{1 + exp(-r_i)} \tag{18}$$

The primary reason for passing the relevance parameters through a sigmoid function is the following: letting the relevances be unconstrained real-valued parameters would not help us conclusively establish which parts indeed belong to the category and which ones do not. What we really want is a binary relevance variable that is 1 if the model part belongs to the category model and 0 otherwise. Having a binary variable would allow us to consider only those parts that truly belong to the object category and discard the irrelevant clutter. Our intuition is that if we squash the unconstrained relevances r_i we effectively turn them into soft binary variables, and during training we force them to be either *relevant* ($\sigma(r_i) \approx 1$) or *irrelevant* ($\sigma(r_i) \approx 0$). This is exactly what happens in practice. The squashed relevances of most parts either go to 1 or 0, thus making it possible to remove the irrelevant ones ($\sigma(r_i) \approx 0$) without affecting the approximate posterior $S(\mathbf{G_o}, \mathbf{r})$. An additional benefit of squashing the relevance parameters is that it basically damps the effect of very large or very small negative values of r_i, reducing overfitting without the need for a regularization term.

The higher the relevances r_i and r_j, the more $\mathbf{G_o}(i,j)$ contributes to the posterior. It is important to note that the relevance of one part is considered with respect to its pairwise relationships with all other parts together with their relevances. Therefore, parts are evaluated based on how well they work together as a team, rather than individually. Also, it is important to understand that we interpret the logistic classifier $S(\mathbf{G_o}, \mathbf{r})$ not as the true posterior, which is impractical to compute, but rather as a distance function specifically tuned for classification.

4.3. Learning

The model parameters to be learned consist of: the pairwise geometric relationships $\mathbf{e}_{ij}$ between all pairs of parts, the sensitivity to deformations $\mathbf{w}$ (which defines the pairwise potentials), the relevance parameters $\mathbf{r}$ and q_0, q_1 (which define the classification function S). The learning steps are described below (Figure 13):

4.3.1 Initialization

We first initialize the pairwise geometric parameters ($\mathbf{e}_{ij}$) for each pair of model parts by simply copying them from a positive training image. Thus, our initial model will have as many parts as the first training image used and the same pairwise relationships. We initialize the rest of the parameters to a set of default values. For each part i we set the default value of its r_i to 0 ($\sigma(r_i) = 0.5$). The default parameters of the pairwise potentials ($\mathbf{w}$) are learned independently as described in Section 4.4.

4.3.2 Updating the Parameters

Starting from the previous values, we update the parameters by minimizing the familiar sum-of-squares error function (typically used for training neural networks) using sequential gradient descent. The objective function is differentiable with respect to r, q_0 and q_1 since they do not affect the optimum x^* (for the other parameters we differentiate assuming fixed x*):

$$J = \sum_{n=1}^{N} b_n (S(\mathbf{G_o}^{(n)}, r) - t^{(n)})^2 \tag{19}$$

Here $t^{(n)}$ represents the ground truth for the n^{th} image (1 if the object is present in the image, 0 otherwise). The weights b_n are fixed to m_N/m_P if $t^{(n)} = 1$ and 0 otherwise, where m_N and m_P are the number of negative and positive images, respectively. These weights balance the relative contributions to the error function between positive and negative examples. The matrix $\mathbf{G_o}^{(n)}$ contains the pairwise potentials for the optimal localization for the n^{th} image.

We update the parameters using sequential gradient descent, looping over all training images for a fixed number of iterations, in practice always leading to convergence. The learning update for any given model parameter λ for the n^{th} example has the general form of:

$$\lambda \leftarrow \lambda - \rho b_n (S(\mathbf{G_o}^{(n)}, \mathbf{r}) - t^{(n)}) \frac{\partial S(\mathbf{G_o}^{(n)}, \mathbf{r})}{\partial \lambda} \tag{20}$$

Using this general rule we can easily write the update rules for all of the model parameters. The pairwise potentials ($\mathbf{G_o}$) do not depend on the parameters $\mathbf{r}, q_0, q_1$. It follows that the optimal labeling $\mathbf{x}^*$ of the localization problem remains constant if we update only $\mathbf{r}, q_0, q_1$. In practice we update only $\mathbf{r}, q_0, q_1$ and the pairwise distances d_{ij}, while assuming that $\mathbf{x}^*$ does not change, thus avoiding the computationally expensive step of matching after each gradient descent update.

4.3.3 Removing Irrelevant Parts

As mentioned earlier, in general the relevance values $\sigma(r_i)$ for each part i tend to converge towards either 1 or 0, with very few parts staying in between. This is due to the fact that the derivative of J with respect to the free relevance parameters r_i is zero only when the output $S(\mathbf{G_o}^{(n)}, \mathbf{r})$ is either 0 or 1, or the relevance $\sigma(r_i)$ is either 0 or 1, the latter being much easier to achieve. This is the key factor that allows us to discard irrelevant parts without significantly affecting the output $S(\mathbf{G_o}^{(n)}, \mathbf{r})$. Therefore, all parts with $\sigma(r_i) \approx 0$ are discarded. In our experiments we have observed that the relevant features found were most of the time belonging to the true object of interest (Figure 15).

4.3.4 Adding New Parts

We proceed by merging the current model with a newly selected training image (randomly selected from the ones on which the recognition output was not close enough to 1): we first localize the current model in the new image, thus finding the subset of features in the image that shares similar pairwise geometric relationships with the current model. Next, we add to the model new parts corresponding to all the image features that did not match the current model parts. As before, we initialize all the corresponding parameters involving newly added parts, by copying the geometric relationships between the corresponding features and using default values for the rest. At this stage, different view-points or shapes of our category can be merged together (Figure 14). The geometric configurations that different aspects share are already in the model (that is why we first perform matching) and only the novel configurations are added (from the parts that did not match). After adding new parts we return to the stage of updating the parameters (see Figure 13). We continue this loop until we are satisfied with the error rate.

Figure 14. The model integrates geometric configurations belonging to different aspects (view-points) within the same category. Training images (left) and the boundary fragments containing the relevant parts learned from different view-points and integrated in the same model (right). Note that the algorithm automatically determined the features that belonged to the object rather than the background

The approach of adding training images one by one is related to incremental semi-supervised learning methods [53]. In our case, we later discard the information that is not useful for recognition (the parts of zero relevance). By removing and adding parts we allow our model to grow or shrink dynamically, as needed for the recognition task.

4.4. Learning the Default Pair-wise Geometric Potentials

We learn a default set of parameters $\mathbf{w}$ for the pairwise potentials independently of the object parts (i, j) and the object class C. Learned independently, the pairwise potentials are logistic classifiers designed to model the posterior that a given pair of assignments is correct given the geometric deformation g, regardless of the object class C or the specific parts (i, j). We learn the default $\mathbf{w}$, from a set of manually selected correct correspondences and randomly selected set of incorrect ones, using the iteratively re-weighted least-squares algorithm. The correspondences are selected from different databases used in the literature: CALTECH-5 (faces, motorbikes, airplanes, motorcycles, cars, leaves, background), INRIA-horses and GRAZ-02 (person, bikes, cars, background). Then we used the same set of default $\mathbf{w}$ for all our recognition experiments.

Data points $\mathbf{g}_{ij}(x_i, x_j)$ are collected for both the positive (pair of correct correspondences) and the negative class (at least one assignment is wrong), where image feature x_i from one image is matched to the image feature i from the other image. For randomly-selected pairs of images containing the same object category, we manually selected approximately 8000 correct correspondences per database (whenever the poses were similar enough so that finding exact correspondences between the contours of the two images was possible). We selected 16000 wrong correspondences per database.

In Table 2 we show how the geometry based pair-wise classifier can generalize across different object categories (note that this is the pairwise potential classifier on pairs of assignments and not a classifier of object categories). In this set of experiments we trained the classifier on pairs of candidate assignments from one database and tested it on pairs of assignments from all three databases. We repeated this ten times for different random splits of the training and testing sets and averaged the results. The interesting fact is that the performance of the classifier is roughly the same (within 1%) for the same test database (indexed by rows), regardless of which database was used for training (indexed by columns). This strongly suggests that the same classifier was basically learned each time, which further implies that the space of geometric second-order deformations is more or less the same for a large variety of solid objects. It follows that the pairwise geometry can be used with confidence on a wide variety of objects even using a single set of default parameters. In our recognition experiments we actually used the default parameters learned only from the Caltech-5 database. The results in Table 2 seem to confirm the idea that accidental alignments are rare events regardless of the object class.

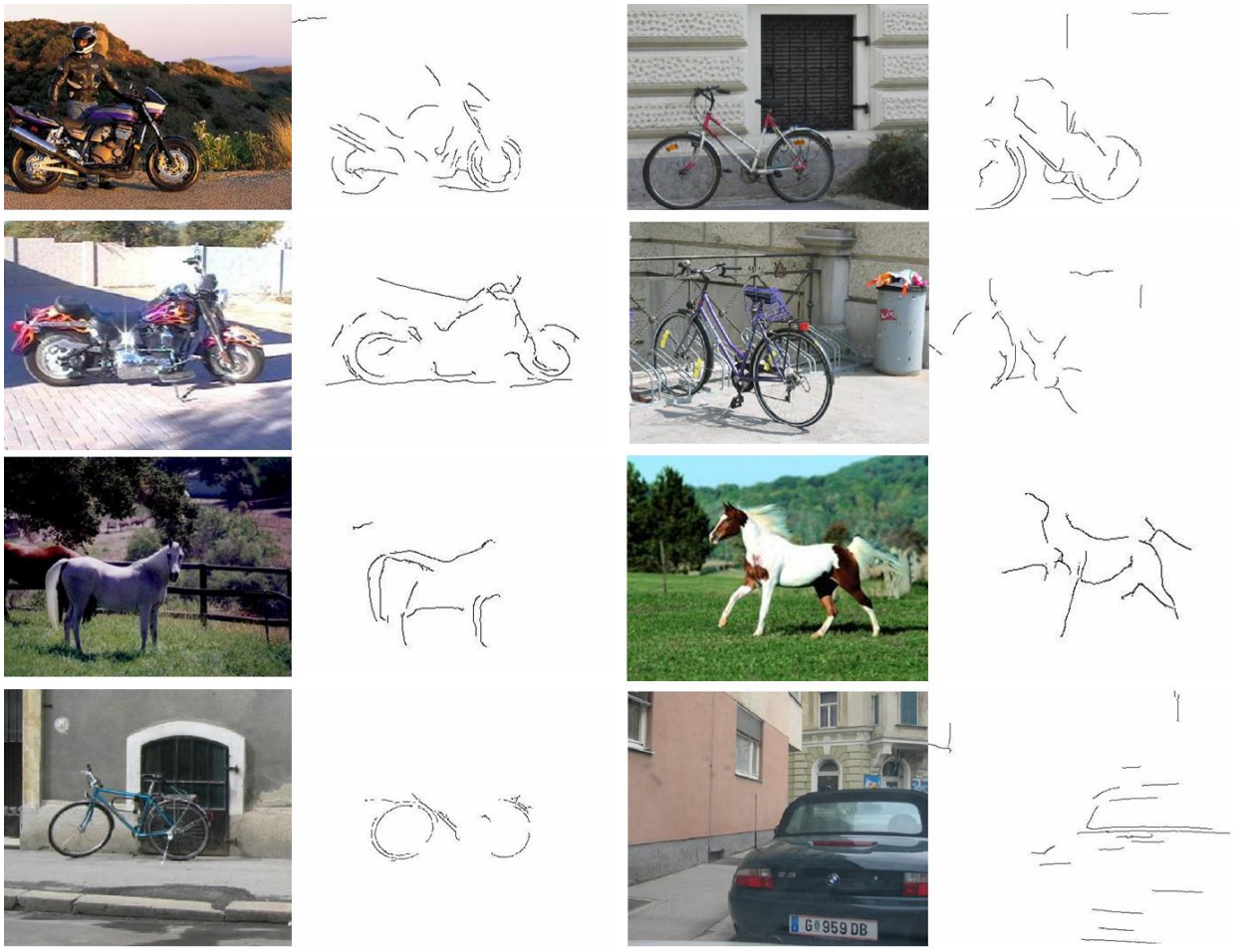

Figure 15. Training images (Left) and the contours on which the relevant features were found during training (Right)

Table 2. The classification rates (at equal error rate) of the geometry based pair-wise classifier trained and tested on different databases: row indicates the test database and the column indicates the training database. Results are averaged over 10 runs.

Database	Caltech-5 (train)	INRIA (train)	GRAZ-02 (train)
Caltech-5 (test)	97.42	97.65	97.23
INRIA (test)	94.66	95.33	94.31
GRAZ-02 (test)	92.93	93.73	93.24

Learning the Pair-wise Potential (proposed work) In the previous section we learned the pair-wise parameters independently of the final recognition task, which is dependent on the specific category model and the performance of our spectral matching method. We intend to use the optimization algorithm presented in Section 3 for learning the pair-wise potentials in a way that will directly improve the recognition performance. The function that we want to maximize (similar to the recognition error function presented previously) quantifies the recognition performance over the training set for a specific object category C and it depends on the parameters $\mathbf{w}$ that define the pair-wise potentials.

$$J_C(\mathbf{w}) = \sum_{n=1}^{N} b_n(1 - |S(\mathbf{w})^{(n)} - t^{(n)}|) \tag{21}$$

Here $J_C(\mathbf{w})$ increases as the overall recognition error decreases, and as before, $S(\mathbf{w})^{(n)}$ is the output of the recognition algorithm on the n^{th} image.

4.5. Experiments

We compare the performance of our method with the one by Winn *et al*. [68] on the Pascal challenge training dataset[1] (587 images) (Table 4). We think this is an interesting experiment because our method focuses only on geometry, ignoring the local appearance, while, in contrast, [68] focuses on local texture information, while ignoring the geometry. We followed the same experimental setup, splitting the dataset randomly in two equal training and testing sets. In the first set of experiments we used the bounding box provided (also used by [68]). We outperform the texture based classifier [68] by more than 10%, which confirms our intuition that shape is a stronger cue than local appearance for these types of object categories. Surprisingly, bikes and motorcycles were not confused as much as we expected, given that they have similar shapes. In the second set of experiments we did not use the bounding boxes[2] (in both training and testing) in order to demonstrate that our algorithm can learn in a weakly supervised fashion. The performance dropped by approximately 5%, which is significant, but relatively low considering that in this experiment the objects of interest sometimes occupy less than 25% of the training and testing images. An even more serious problem is that some positive images for one class contained objects from the other classes (e.g. there are people present in some of the motorcycle and car positive images), which we did not take into account (a motorbike positive image containing a person and classified as person was considered an error).

Table 3. Confusion Matrix for Pascal Dataset (using the bounding boxes)

Category	Bikes	Cars	Motorbikes	People
Bikes	80.7%	0%	7%	12.3%
Cars	5.7%	88.6%	5.7%	0%
Motorbikes	4.7%	0%	95.3%	0%
People	7.1%	0%	0%	92.9%

Table 4. Average multiclass recognition rates on the Pascal Dataset

Algorithm	Ours (bbox)	Ours (no bbox)	Winn [68] (bbox)
Pascal Dataset	89.4%	84.8%	76.9%

As mentioned earlier the models we learn are compact representations of the relevant features present in the positive training set. The algorithm is able to discover relevant parts that, in our experiments, belong in a large majority to the true object of interest, despite the background clutter present sometimes in large amounts in the training images (Figure 15). An interesting and useful feature of our method is that it is able to integrate different view-points, aspects or shapes within the same category (Figure 14). This happens automatically, when new parts are added from positive images on which the recognition output was not high enough.

The computational cost of classifying a single image does not depend on the number of training images: the model is a compact representation of the relevant features in the training images, usually containing between 40 to 100 parts. It is important to note that the size of the model is not fixed manually, it comes out automatically from the learning stage.

5. Object Recognition with Grouping (proposed work)

In Section 4 ([39]) we presented an object category recognition algorithm that did not use any grouping cues to help the matching and recognition of specific categories. The matching part of the algorithm considered all pairs of features (pieces of contours) from an image, as if all pairs were equally likely to belong to the same object. Grouping is a way of constraining the matching/recognition search space by considering only pairs of features that are likely to come from the same object. We believe that grouping is essential in improving the recognition rate because it uses general, category independent information to prune the search space and guide the recognition process on the right path. In Figure 17 we show two examples that intuitively explain this idea. The images in the left column contain edges extracted from a scene. We notice that without grouping the objects are not easily distinguished (e.g. the bus, or the horse). However, after using color

[1] http://www.pascal-network.org/challenges/VOC/voc2005/index.html. We also ignored the gray level UIUC car images

[2] For very few images in which the object was too small we selected a bounding box 4 to 5 times larger in area than the original one

Figure 16. The grouping stage enforces pairwise constraints that could significantly prune our matching/recognition search space. The contours in white (for second and third image) are the ones likely to be on the same object as the red circle, given the geometric or color cues

information for perceptual grouping we are able to retain only the edges that are likely to belong to the same object as the edge pointed out by the red circle (right column). Perceptual grouping could also bring a second benefit to the recognition process, because, without it, matching could be very expensive especially when the image contains a lot of background features/clutter. As Grimson [30] has shown, the complexity of the matching process when the search is constrained is reduced from an exponential to a low order polynomial. Therefore, it is important to be able to establish *a priori* which pairs of features are likely to be part of the same object, and discard all the other pairs. To summarize, perceptual grouping does not only improve the recognition performance but it also reduces the computational complexity.

5.1. Pairwise Grouping of Features

Grouping is the task of establishing which features in the image are likely to belong to the same object, based on cues that do not include the knowledge about the specific object or object category. In his pioneering book [47] Marr argued, based on medical human studies, that a vision system should be able to recover the 3D shape of objects without knowledge about the objects' class or about the scene. In support of this idea come simple facts from everyday life. Humans are able to see a car parked in a room, even though that would be totally unexpected based on prior experience. Also, humans are able to perceive the shape of an abstract sculpture from a single image, even if they have never seen it before and have no clue about what it might represent. It seems clear that for humans both knowledge about the object class and the scene is not necessary for shape perception. But if one is able to perceive the 3D shape of the scene with respect to its own reference frame, then in most cases it would be relatively easy to figure out which features in the scene should belong together. In fact perceptual grouping most probably helps humans in the process of 3D shape recovery and not the other way around. We consider grouping as a process that happens entirely before the object recognition stage, and uses cues such as color, texture, shape, and perceptual principles that apply to most objects in general, regardless of their category or specific identity.

Unlike prior work in grouping [46], [59], [1], [34], [54], [50], [44], we do not make a hard decision about which features belong together. And that is for an important reason: it is sometimes impossible to divide features into their correct groups without the knowledge of the specific category (or the desired level of detail): for example, is the wheel of a car a separate object or is it part of the whole car? We believe that both situations can be true at the same time, depending on what we are looking for. If we are looking for whole cars, than the wheel is definitely a part of it. If we are looking just for wheels then (at least conceptually) it is not. While perceptual grouping alone should most of the time separate correctly most objects (the ones that are clearly at different depths, such as a flying plane from a close car), it sometimes does not have access to enough information to make the correct hard decisions. We immediately see why it is important to keep most the grouping information around and transmit it to the higher recognition processes (without making hard decisions, except for pruning the cases when the pairwise grouping relationship is extremely weak). Instead of being interested on forming exact feature groups based on perceptual information alone, we rather focus on the quality of pair-wise grouping relationships and how to integrate them into our recognition step. Since we use pair-wise relationships at both the grouping and the recognition levels, the two could be naturally integrated.

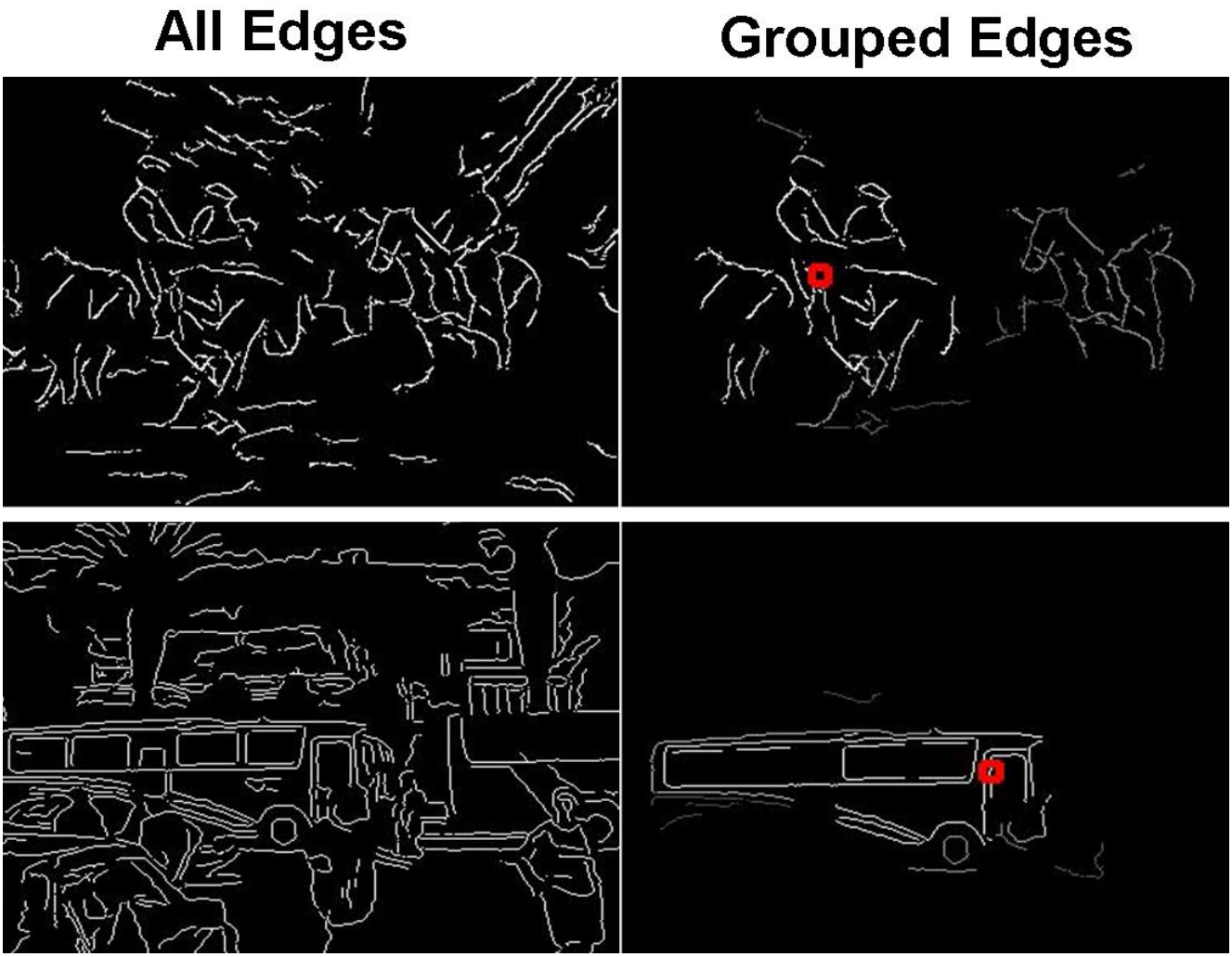

Figure 17. If grouping is not used it is very hard to distinguish the separate objects (left column). After grouping (right column) it is perceptually easier to distinguish them (the bus and the horse)

Our pair-wise grouping relationships are soft weights that should reflect the likelihood that the two features belong together (prior to recognition, that is before using any category specific knowledge). We focus on two types of perceptual pair-wise grouping:

1. Geometric based, Gestalt-like cues (between line segments or contours), such as proximity, good continuation, parallelism/perpendicularity (see Section 5.2 and Figure 18 for more details)

2. Color based: objects tend to have unique and relatively homogenous color distributions. We propose a novel and powerful way of using color histograms for inferring pair-wise grouping relationships (Section 5.3)

5.2. Pairwise Grouping using the Geometry of Line Segments

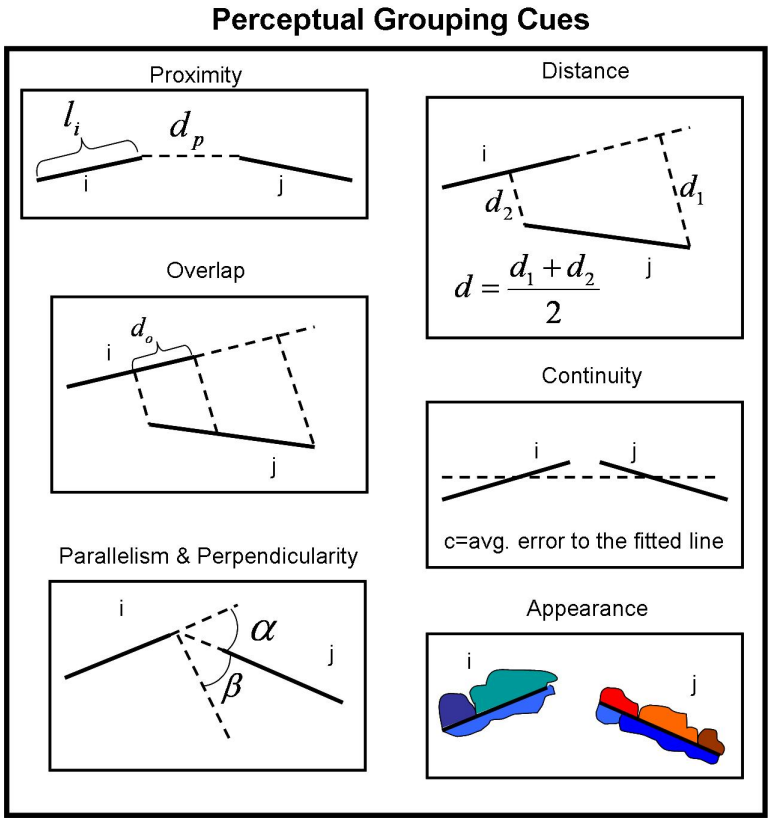

Figure 18. Geometric perceptual cues used for grouping pairs of line features

Table 5. Perceptual cues used to describe the relationship between pairs of lines. Based on these cues we estimate the likelihood that pairs of lines belong to the same object or not

Cue	Description
Proximity	$\frac{d_p}{l_i+l_j}$
Distance	$\frac{d_i+d_j}{l_i+l_j}$
Overlap	$\frac{d_{oi}+d_{oj}}{l_i+l_j}$
Continuity	c
Parallelism	α
Perpendicularity	β
$Color_1$	difference in mean colors
$Color_2$	difference in color histograms
$Color_3$	difference in color entropies

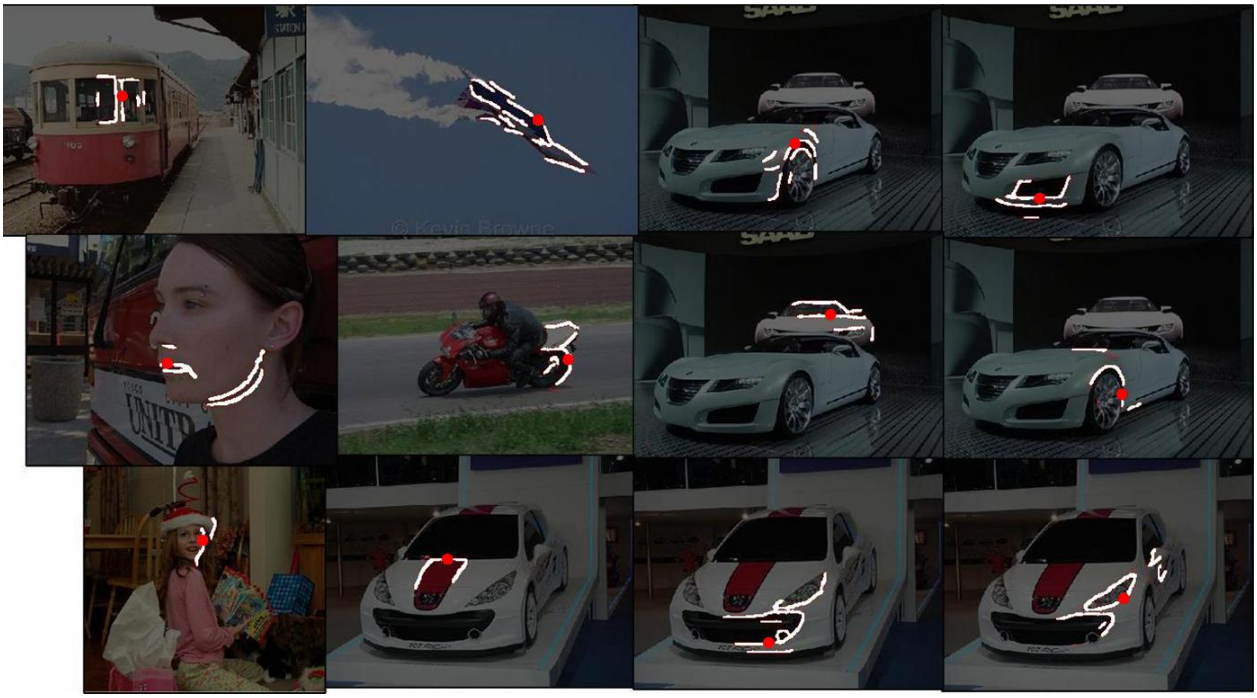

Figure 19. Pairwise grouping constraints based on geometry. The contours in white are the ones that establish a pairwise grouping relationship with the contour pointed out by the red circle

In the task of feature grouping it is important to use as many cues as possible. Geometric perceptual cues are particularly important because of their connection to important studies in human vision (mainly from the Gestalt school). Our main features for object recognition are pieces of contours extracted from the image. In the grouping stage we approximate these contours by fitted line segments. The geometric grouping cues we propose to use consist of specific relationships between pairs of such line segments (i, j): proximity, distance, overlap, continuity, parallelism and perpendicularity as shown in Figure 18. We also use some local appearance cues (which are different than the global color histograms used in the next section), which are computed over the super-pixels adjacent to the pair of lines, such as: difference between the mean colors of the super-pixels belonging to each line, as well as the differences in color histogram and color entropy. All these cues, both geometric and appearance based, form a *relation* vector $\mathbf{r}(\mathbf{i}, \mathbf{j})$ for any pair of lines (i, j) whose elements are described in Table 5 (each row of the table corresponds to an element of $\mathbf{r}(\mathbf{i}, \mathbf{j})$). We have already performed some preliminary experiments with a basic implementation of this idea. We have manually collected about 300 positive pairs of lines (lines that belong to the same object) and 1000 negative ones (pairs of lines which do not belong to the same object), and learned a binary classifier on the corresponding relation vectors $\mathbf{r}$, using the logistic regression version of Adaboost [12] with weak learners based on decision trees [26]. We intend to use the soft output of this classifier (the likelihood that a pair of lines belongs to the same object) later in the object recognition part of our proposed work.

In Figure 19 we present some results. The contours shown in red belong to line segments that were classified as being part of the same object as the line segment pointed by the white circle. We notice that in general only lines that are relatively close to the white circle are positively classified. This is due to the fact that in general geometric perceptual grouping is a local process and is not able to link directly pairs of faraway lines. Such pairs could be ultimately connected indirectly thorough intermediate lines. Of course, these are only preliminary results, and more thorough experimentation is needed. More specifically, we would probably need to:

1. Consider the output of the classifier as a pairwise weighing function and use it to connect pairs of lines through other intermediate lines

2. Train the classifier on larger training sets.

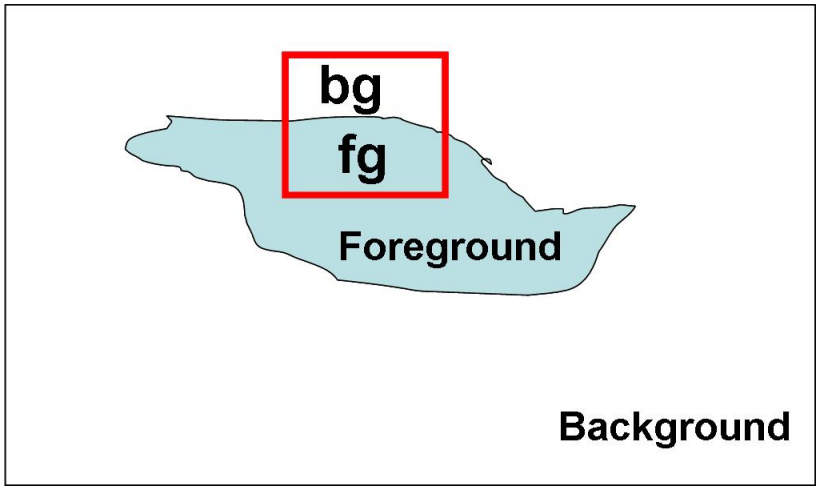

Figure 20. Automatically discovering the foreground mask

3. Instead of training the classifier on independent pairs of lines, tune its output such that the final goal (soft segmentation of line segments) is optimized

5.3. Pairwise Grouping using Global Color Statistics

Color histograms are simple but powerful global statistics about objects' appearances that have been successfully used in some recognition applications. We present a novel algorithm for automatically discovering soft object masks (based on their color distributions), without knowledge about their locations or shapes. This method becomes very useful for pairwise grouping of features, because two features that belong (in a soft way) to the same (soft) mask are more likely to belong to the same object then pairs of features that belong to different (soft) masks.

In order to emphasize the power of color histograms and understand better how we approach the pairwise grouping based on color, we first present a toy foreground/background segmentation application (Figures 21, 6) that is inspired by GrabCuts [55] and Lazy Snapping [41]. This application is based on our optimization algorithm presented in 3. For now we assume that we know the ground truth bounding box and then automatically obtain the object mask (which will help, as explained later, with how pairs of features connect based on color). Later on we will also show that, since it is not crucial to know the exact size, shape and location of the bounding box, one can obtain fairly accurate soft masks of the objects in the scene, without knowledge of their ground truth bounding boxes.

The user is asked to provide the bounding box of an object, and the algorithm has to return a polygon which should be as close as possible to the true object boundary. This is just another instance of the foreground-background segmentation problem. In computer vision most segmentation algorithms approach this task from bottom up. The problem is usually formulated as a Markov Random Field [40] with unary and pairwise terms that use information only from a low, local level, and do not integrate a global view of the object, which would be needed for a better segmentation. Here we present a simple algorithm for obtaining object masks, that is based on the global statistics of the foreground vs. the background. The main idea is that a good segmentation is the one that finds the best separation (in terms of certain global statistics) between the foreground and the background. In this case we use color likelihoods derived from color histograms (of the initial foreground and background, defined by the bounding box given) as their global statistics (we have successfully applied a very similar idea to object tracking [56]) . Starting from the bounding box provided by the user, the algorithm has to find the polygon that best separates the color likelihood histogram computed over the interior of the polygon (foreground) from the corresponding histogram computed over its exterior (background). The function to optimize looks very simple but it is non-differentiable, highly non-linear and has high dimensionality, so the task could be very difficult:

$$f(\mathbf{x}, I) = 1 - h_f(\mathbf{x}, I)^T h_b(\mathbf{x}, I) \tag{22}$$

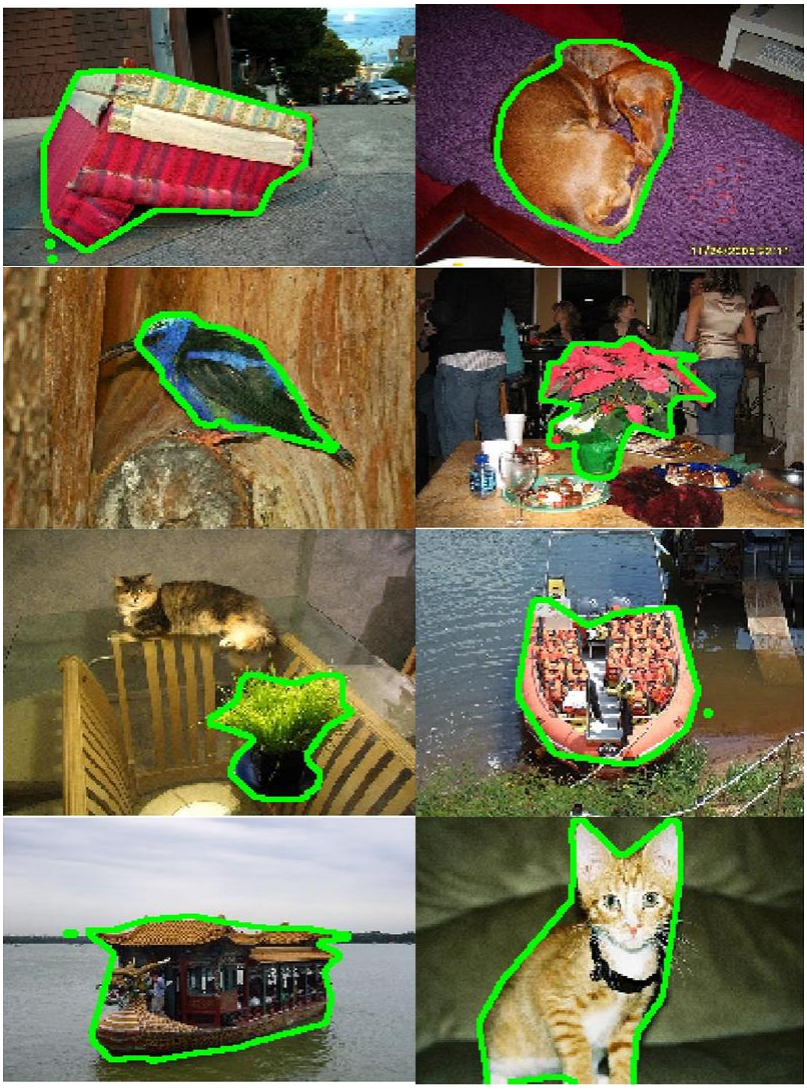

Figure 21. The masks of objects are automatically found, given their ground truth bounding boxes

Here x are the vertices of the polygon, $h_f(\mathbf{x}, I)$ and $h_b(\mathbf{x}, I)$ are the foreground and background normalized color likelihood histograms, given the image I . The likelihood of color c is computed as $l(c) = \frac{N_f(c)}{N(c)}$, where $N_f(c)$ is the number of pixels of color c inside the initial bounding box and $N(c)$ is the total number of pixels of color c in the image.

Our segmentation algorithm is very simple and can be briefly described as follows (it is a toy implementation used only to prove the power of our optimization method):

1. initialize x with the bounding box provided by the user
2. find $x^* = argmax f(x)$, by using the algorithm from Section 3.3 without changing the number of polygon vertices
3. if x^* does not improve significantly over the previous solution stop.
4. add new vertices at the midpoints of the edges of x^*
5. go back to step 2

As long as the color distribution of the foreground is different from the background, this algorithm works well, being very robust to local variations in color or texture, unlike MRF based algorithms whose unary and pairwise terms are more sensitive to such local changes (see Figures 21, 6).

Unsupervised discovery of the foreground mask As the previous application showed, color histograms alone are often powerful enough to segment the foreground vs the background, without using any local intensity information or edges. In the previous application we used the ground truth bounding box of the object (provided by the user), but in most recognition applications we do not have access to such bounding boxes. We want to be able to use the power of color histograms without knowing the mask (or bounding box) of the foreground. But this seems almost impossible. How can we compute the color histogram of the foreground if we have no clue about the foreground size and rough shape?

During our experiments with a given bounding box we found that the algorithm was surprisingly robust to the exact size and location of the bounding box. More precisely in most cases one could find a fairly good foreground mask based on color distributions starting from a bounding box centered on the object, but of completely wrong shape, size and location. These apparently surprising experimental findings can actually be easily explained theoretically. In Figure 20 the foreground is shown in blue and the bounding box in red. The bounding box's center is on the foreground, but its size and location are obviously wrong. We make the following assumptions, which are very reasonable in practice:

1. The area of the foreground is smaller than that of the background: this is true for most objects in images
2. The majority of pixels inside the bounding box belong to the true foreground: this is also often true in practice since the center of the bounding box is considered to be on the foreground object by (our own) definition. (This definition is sound in the context of our thesis since we are considering pairs of features, so two features that belong to the same object are considered in the foreground for some bounding box centers).
3. The color distributions of the true background and foreground inside the bounding box are the same as the ones outside it: this assumption is reasonable in practice, but harder to meet than the first two.

Even though the three assumptions above are not necessarily true all the time in practice, most of the time they do not need to be *perfectly true* for the following result to hold (they represent only loose sufficient conditions): let bg and fg be the true foreground and background distributions, and FG and BG the ones computed using the (wrong) bounding box satisfying the assumptions above. Then one can easily prove that for any color c such that $fg(c) > bg(c)$ we must also have $FG(c) > BG(c)$. This result enables us to use color histograms as if we knew the true object mask, by using any bounding box satisfying the assumptions above.

In Figure 22 we present some results using this idea. We compute the objects' masks over four different bounding boxes of increasing sizes (the sizes are fixed, the same for all examples) and return as the final result the average bounding box (last column). In Figure 23 we show that the mask obtained is robust to the location of the bounding boxes, so long as the bounding boxes' centers are on the object of interest.

As we mentioned already, the idea of automatically discovering foreground masks at given locations can be easily used for pairwise grouping of features. To prove our point, we present a simple approach for using such masks for color grouping. Given two features (i, j) one can compute the associated soft masks m_i and m_j by centering the bounding boxes at the features locations (x_i, y_i) and (x_j, y_j) and follow the procedure explained previously, using bounding boxes of those four fixed different sizes. The values $m_i(y_j, x_j)$ and $m_j(y_i, x_i)$ can then be used by a classifier to establish the likelihood that the two features belong to the same object. In Figure 24 we present some preliminary results of this idea. The red circles represent the location of some contour (feature) i. In white we show all those contours (features) j that were automatically classified as likely to belong to the object of i. Here the classifier was simply thresholding the average $(m_i(y_j, x_j) + m_j(y_i, x_i))/2$ at 0.5 (everything above 0.5 was considered positive and vice-versa. The images show weighted contours for the positive examples, and no contours for the negative). It is important to note that the shape and extent of the foreground is not known, and that all internal parameters are fixed (such as the four fixed bounding box sizes).

In Figure 25 we present some *failure* examples of the color pairwise constraints. The algorithm does fail in the sense that it connects contours from the house to contours from the car, but it also connects car contours among themselves, so it should still improve the recognition performance (because most clutter is removed). The main reason for this lower quality results is that the house and the car have similar colors (relative to the histogram bin-ing). This issue could probably be solved to a certain extent by improving the color histogram-ing. We also want to point out that these *failures* are the exception and not the rule. In fact the vast majority of our results are of similar quality with those in Figure 24. However, as future work, we need to quantitatively measure the grouping performance.

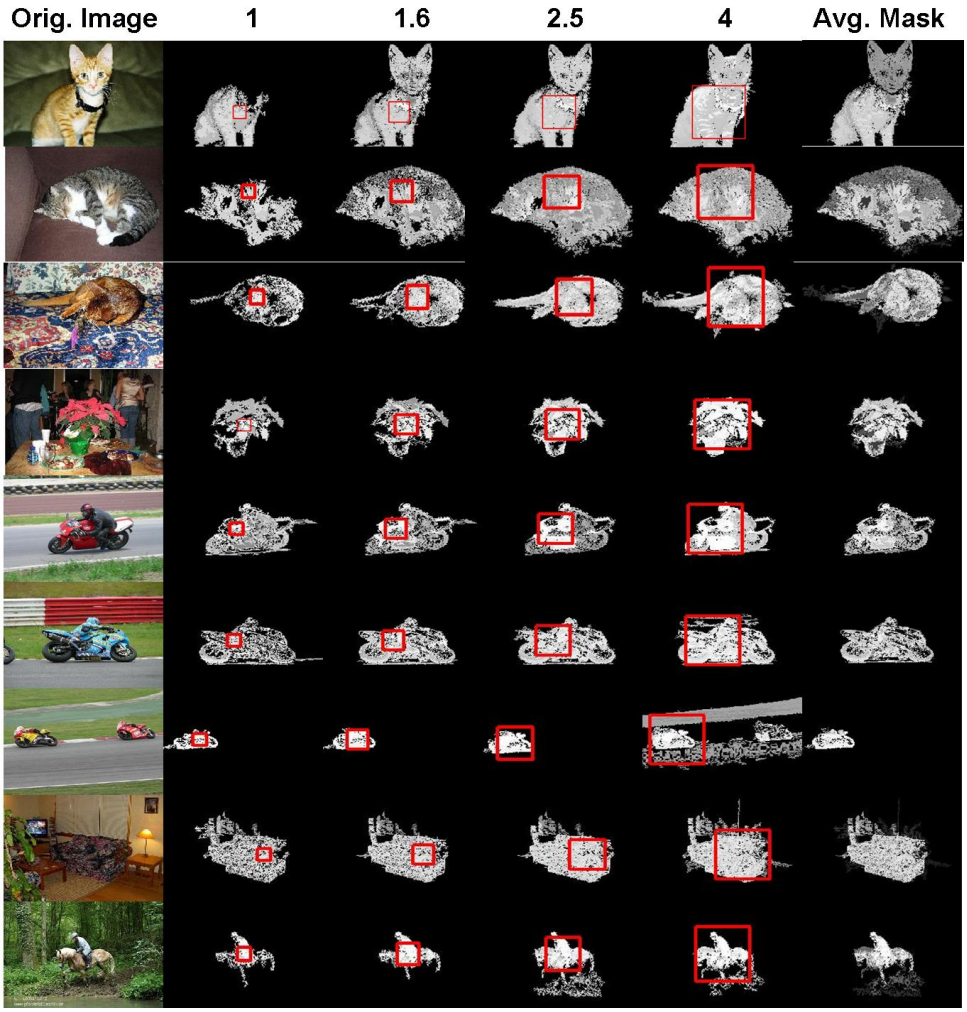

Figure 22. Object masks obtained from completely wrong bounding boxes, centered on the object of interest. Averaging over several fixed scales improves the result. The masks shown are computed from color likelihood histograms based on the bounding boxes (completely wrong) shown, which are centered on the object of interest. The interior of the boxes is considered to be the foreground, while their exterior is the background. The posterior color likelihood image is obtained, threshold-ed at 0.5 and the largest connected component touching the interior of the bonding box is retained. We notice the even when the boxes have a wrong location and size (their center and size do not correspond to the true object center and size) the masks obtained are close to the ground truth. Different bounding boxes sizes (which are fixed for every image, starting at 8 pixels and growing at a rate of 1.6) are tried and the average mask is shown in the rightmost column.

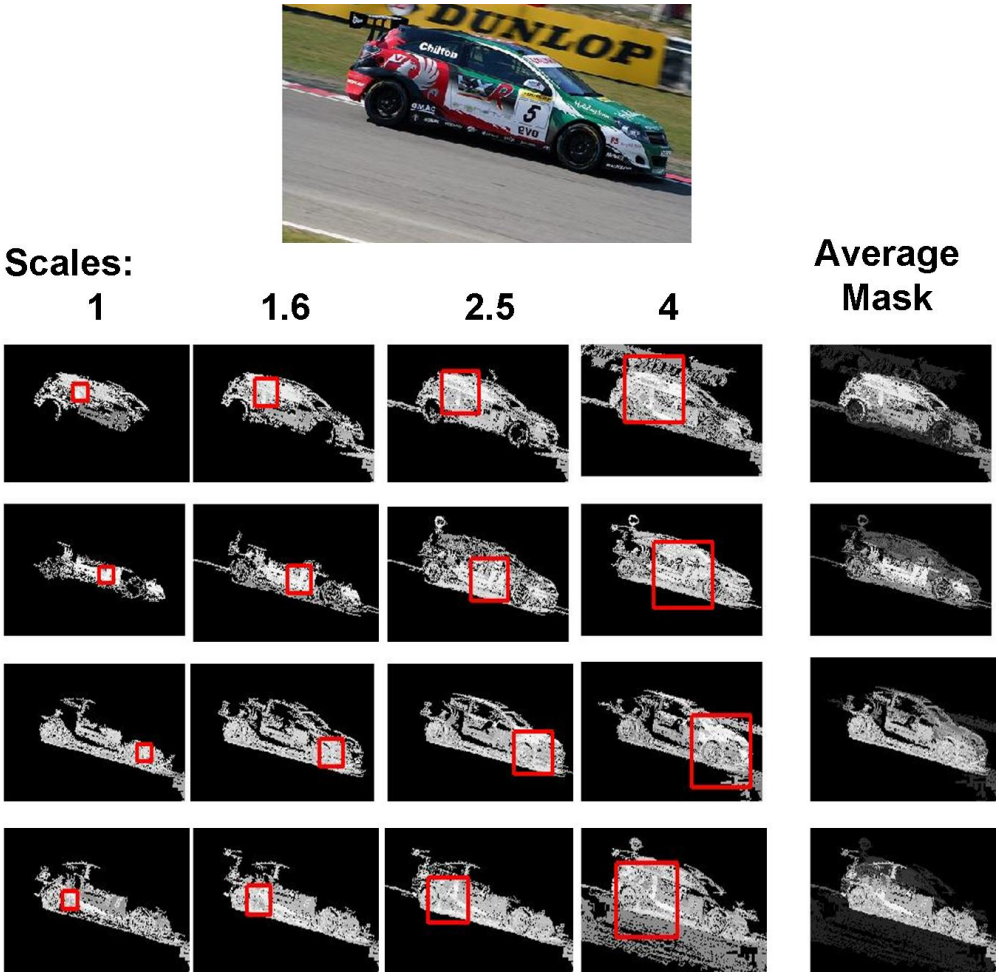

Figure 23. As in the previous Figure, finding reasonable object masks is robust to the location of the bounding box, even for objects with a complex color distribution.

5.4. Combining Matching with Grouping

We propose to combine the grouping constraints with the category specific constraints, by augmenting the initial pairwise scores $M(ia; jb)$ as follows:

$$M(ia; jb) = M_0(ia; jb)P(a, b) \tag{23}$$

Here $M_0(ia; jb)$ is the score previously used in Section 4, that measures how well the geometry (and possibly the local appearances) of model features (i, j) agrees with the geometry of image features (a, b). Using the grouping cues we include the perceptual grouping score $P(a, b)$ that uses *a priori* pairwise grouping information to quantify how likely the image features (a, b) are to be part of the same object:

$$P(a, b) = \mathbf{exp}(w_0 + w_1 g(a, b) + w_2 c(a, b)) \tag{24}$$

Here $P(a, b)$ includes both geometric $(g(a, b))$ and color based $(c(a, b))$ cues obtained as suggested previously. Both $g(a, b)$ and $p(a, b)$ are the output of the classifiers trained on their corresponding cues. In Figure 16 we present the potential

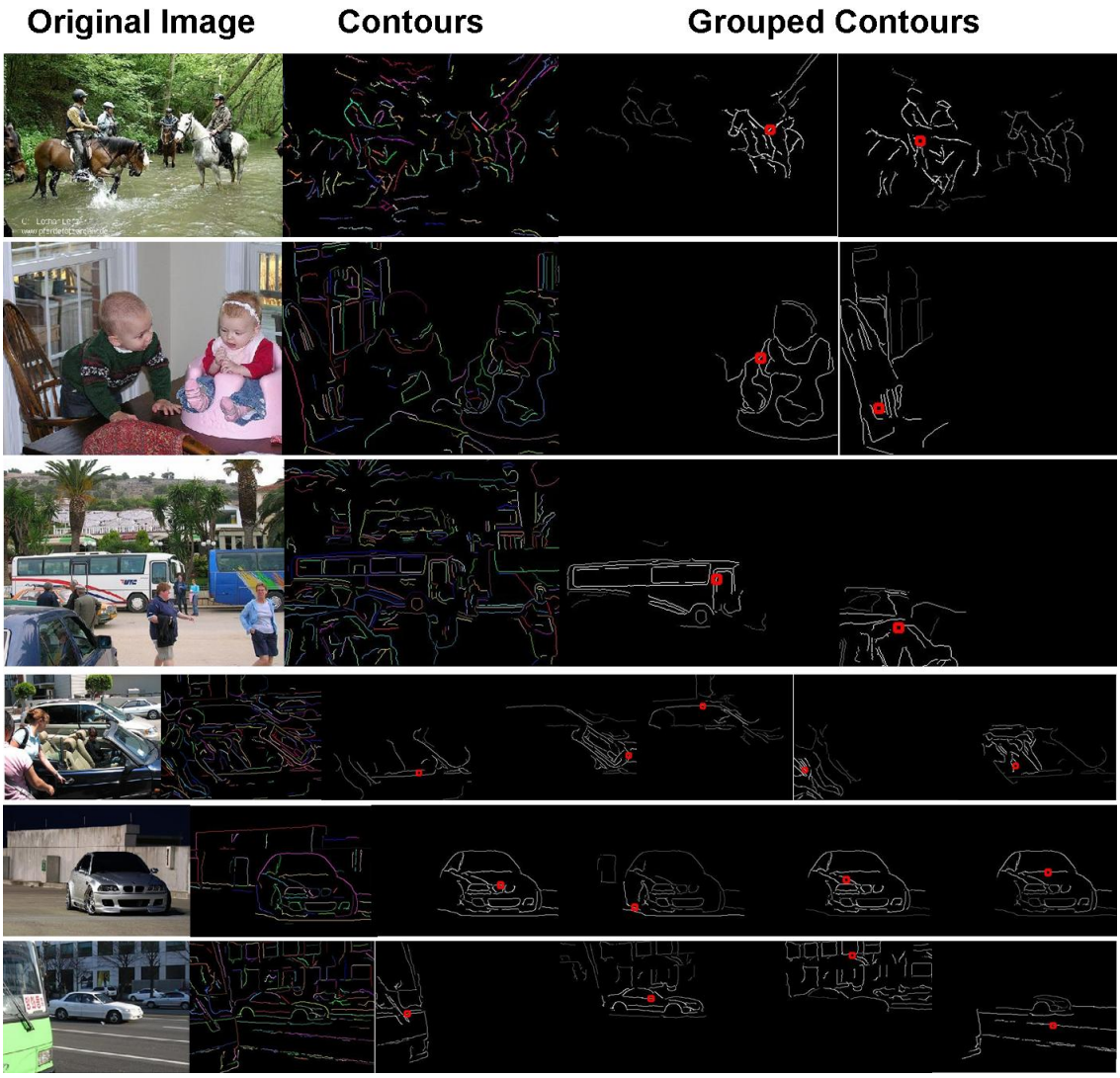

Figure 24. Pairwise grouping relationships (constraints) based on color distribution. The contours shown in white are the ones establishing a pairwise grouping relationship based on color with the contour pointed out by the red circle. Notice some difficult cases from very cluttered scenes. The second column (next to the original image) shows all the contours extracted, while the next images show the contours that form a positive grouping relationship with the contour shown by the red circles.

advantage of using grouping. On the left we show all the pieces of contours (in white) that are likely to be part of the same object as the contour indicated by the red circle, as considered by the previous work 4. Since no grouping information was used, all contours are considered. In the middle we show the contours likely to be grouped with circle if we use the current implementation of the geometrically driven grouping method. On the right, we show the same type of results if we are using the color histogram based grouping. As already discussed, it is clear that grouping could significantly improve the performance of our recognition algorithm, since most pairs that should not be considered can be automatically discarded as shown in Figure 16. At this point we do not know precisely what is the way of optimally combining pairwise grouping with

Figure 25. Examples when color grouping does not work so well. Upper left corner: original image. Upper middle: all the contours extracted. The rest: the results are shown in the same style as in Figure 24. The results are of worse quality than the ones from Figure 24. We can see that parts of the house are weakly connected to contours from the car. This happened mainly because the house and the car have similar colors, and the differences were lost during histogram bin-ing.

feature matching, but it should be clear that even a straight forward, natural solution, such as the one we presented in this Section, should improve the recognition performance.

6. Conclusions

Our completed work (Spectral Matching and Object Category Recognition without Grouping) clearly demonstrates that there is a lot of potential in using pairwise constraints for object recognition. By improving our matching algorithm, through both learning (Learning for Spectral Matching) and integrating grouping cues (both Geometric and Color based) we hope to significantly improve the recognition performance. Below we reiterate the main parts of this thesis in terms of completed and proposed work:

The most significant completed work includes:

1. Spectral Matching, Section 2: an efficient algorithm for feature matching using second order terms. It is basically a solver for the Quadratic Assignment Problem applied to computer vision.

2. Object Recognition without Grouping, Section 4: a novel approach for semi-supervised learning of object categories using higher order interactions between simple features. The same framework can easily incorporate more complex features

The proposed work (or work in progress) can be summarized as follows (in the order of its importance):

1. Object Recognition with Grouping, Section 5: a novel, unifying approach for combining object specific information (matching model parts to image features) and image based (model independent) grouping cues (by May 2009)

2. An efficient algorithm for grouping (in a soft way) image features based on color distributions and geometric relationships, Section 5 (by December 2008)

3. A new general optimization method for complex functions and its applications to grouping and object recognition, Section 3 (by May 2008)

4. An efficient method for learning graph matching, specifically designed for the spectral matching algorithm, Section 2 (by May 2008)

References

[1] G. M. J. I. A. Etemadi, J.-P. Schmidt and J. Kittler. Low-level grouping of straight line segments. In *BMVC*, 1991. 25

[2] L. Baum and G. Sell. Growth transformations for functions on manifolds. 14

[3] S. Belongie, J. Malik, and J. Puzicha. Shape context: A new descriptor for shape matching and object recognition. In *NIPS*, 2000. 17

[4] A. Berg, T. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *CVPR*, 2005. 3, 4, 5, 6, 17

[5] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, 1987. 11

[6] G. Bouchard and B. Triggs. Hierarchical part-based visual object categorization. In *CVPR*, 2005. 17

[7] R. Brooks. *Symbolic Reasoning mong 3-D Models and 2-D Images*. PhD thesis, Stanford University Computer Science Dept., 1981. 2

[8] T. Caetano, L. Cheng, Q. Le, and A. J. Smola. Learning graph matching. In *ICCV*, 2007. 6, 7, 10, 11

[9] M. Carcassoni and E. R. Hancock. Alignment using spectral clusters. In *BMVC*, 2002. 4

[10] G. Carneiro and D. Lowe. Sparse flexible models of local features. In *ECCV*, 2006. 17

[11] J. Cheng and M. J. Druzdzel. Ais-bn: An adaptive importance sampling algorithm for evidential reasoning in large bayesian networks. 13

[12] M. Collins, R. Schapire, and Y. Singer. Logistic regression, adaboost and bregman distances. In *Machine Learning*, 2002. 28

[13] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 24(5):603–619, 2002. 14

[14] D. Comaniciu, V. Ramesh, and P. Meer. The variable bandwidth mean shift and data-driven scale selection. In *ICCV*, 2001. 14

[15] T. Cour, J. Shi, and N. Gogin. Learning spectral graph segmentation. 2005. 8, 10

[16] D. Crandall, P. Felzenszwalb, and D. Huttenlocher. Spatial priors for part-based recognition using statistical models. In *CVPR*, 2005. 17

[17] D. Crandall and D. Huttenlocher. Weakly supervised learning of part-based spatial models for visual object recognition. In *ECCV*, 2006. 17

[18] G. Csurka, C. Bray, C. Dance, and L. Fan. Visual categorization with bags of keypoints. In *ECCV Workshop on Statistical Learning in Computer Vision*, 2004. 1

[19] T. L. P. E. Grimson. Recognition and localization of overlapping parts from sparse data. Technical report, MIT, 1984. 17

[20] W. P. et al. *Numerical Recipes in C*. Cambridge University Press, 1999. 14

[21] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 2004. In press. 17

[22] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, volume II, pages 264–270, 2003. 17

[23] R. Fergus, P. Perona, and A. Zisserman. A sparse object category model for efficient learning and exhaustive recognition. In *CVPR*, 2005. 17

[24] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. Technical report, INRIA, 2006. 16

[25] V. Ferrari, T. Tuytelaars, and L. V. Gool. Object detection by contour segment networks. In *ECCV*, 2006. 17

[26] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. In *Annals of Statistics*, 2000. 28

[27] C. Goad. Special purpose automatic programming for 3D model-based vision. In *Proc. DARPA Image Understanding Workshop*, pages 94–104, June 1983. 2

[28] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. In *PAMI*, 1996. 6, 10

[29] K. Grauman and T. Darrell. Pyramid match kernels: Discriminative classification with sets of image features. In *ICCV*, 2005. 17

[30] W. E. L. Grimson. *Object Recognition by Computer: The Role of Geometric Constraints*. MIT Press, 1990. 2, 25

[31] C. hotel dataset. http://vasc.ri.cmu.edu/idb/html/motion/hotel/index.html. 10

[32] C. house dataset. http://vasc.ri.cmu.edu/idb/html/motion/house/index.html. 10

[33] J. E. Hummel. Where view-based theories break down: The role of structure in shape perception and object recognition. In *In Cognitive Dynamics: Conceptual Change in Humans and Machines. Erlbaum*, 2000. 1

[34] D. Jacobs. Robust and efficient detection of salient convex groups. In *PAMI*, 1996. 25

[35] D. Kanevsky. Extended baum transformations for general functions. In *Acoustics, Speech, and Signal Processing*, 2004. 14
[36] S. Kumar. *Models for Learning Spatial Interactions in Natural Images for Context-Based Classification*. PhD thesis, The Robotics Institute, Carnegie Mellon University, 2005. 17, 18
[37] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*. Morgan Kaufmann Publishers Inc., 2001. 17
[38] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *ICCV*, 2005. 2, 6, 10, 17, 19
[39] M. Leordeanu, M. Hebert, and R. Shukthankar. Beyond local appearance: Category recognition from pairwise interactions of simple features. In *CVPR*, 2007. 16, 18, 24
[40] S. Z. Li. *Markov Random Field Modeling in Computer Vision*. Springer, 1995. 2, 29
[41] Y. Li, J. Sun, and C. T. H. Shum. Lazy snapping. 2004. 29
[42] T. Lindeberg. Scale-space behaviour of local extrema and blobs. In *Journal of Mathematical Imaging and Vision*, 1992. 11, 12
[43] M. Loog, J. J. Duistermaat, and L. M. J. Florack. On the behavior of spatial critical points under gaussian blurring. In *International Conference on Scale-Space and Morphology in Computer Vision*, 2001. 12
[44] D. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, Boston, 1985. 2, 25
[45] J. Maciel and J. Costeira. A global solution to sparse correspondence problems. In *PAMI*, 2003. 4, 5, 6
[46] S. Mahamud, L. Williams, K. Thornber, and K. Xu. Segmentation of multiple salient closed contours from real images. In *PAMI*, 2003. 4, 25
[47] D. Marr. *Vision*. Freeman, San Francisco, 1982. 2, 25
[48] N. Metropolis and S. Ulam. The monte carlo method. In *Journal of the American Statistical Association*, 1949. 14
[49] K. Mikolajczyk, B. Leibe, and B. Schiele. Multiple object class detection with a generative model. In *CVPR*, 2006. 17
[50] R. Mohan and R. Nevatia. Using perceptual organization to extract 3-d structures. In *PAMI*, 1994. 25
[51] A. Opelt, A. Pinz, M. Fussenegger, and P. Auer. Generic object recognition with boosting. *PAMI*, March 2006. 17
[52] A. Opelt, A. Pinz, and A. Zisserman. A boundary-fragment-model for object detection. In *ECCV*, 2006. 16
[53] C. Rosenberg. *Semi-Supervised Training of Models for Appearance-Based Statistical Object Detection Methods*. PhD thesis, Carnegie Mellon University, 2004. 22
[54] G. Roth and M. Levine. Geometric primitive extraction using a genetic algorithm. In *PAMI*, 1994. 25
[55] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. 29
[56] Y. L. RT Collins and M. Leordeanu. Online selection of discriminative tracking features. In *PAMI*, 2005. 29
[57] R. Rubinstein. The cross-entropy method for combinatorial and continuous optimization. In *Methodology and Computing in Applied Probability*, 1999. 13
[58] S. Sarkar and K. Boyer. Quantitative measures of change based on feature organization: Eigenvalues and eigenvectors. In *Computer Vision and Image Understanding*, 1998. 4
[59] S. Sarkar and P. Soundararajan. Supervised learning of large perceptual organization: Graph spectral partitioning and learning automata. In *PAMI*, May 2000. 25
[60] G. Scott and H. C. Longuet-Higgins. Feature grouping by relocalisation of eigenvectors of the proximity matrix. In *BMVC*, 1990. 4
[61] R. D. Shachter and M. A. Peot. Simulation approaches to general probabilistic inference on belief networks. 13
[62] L. Shapiro and J. Brady. Feature-based correspondence - an eigenvector approach. In *Image and Vision Computing*, 1992. 4
[63] J. Shi and J. Malik. Normalized cuts and image segmentation. In *PAMI*, 2000. 4
[64] C. Sminchisescu and B. Triggs. Fast mixing hyperdynamic sampling. 2004. 11
[65] A. Treisman. Features and objects in visual processing. In *Scientific American*, November 1986. 2
[66] Tu and S. C. Zhu. Image segmentation by data driven markov chain monte carlo. In *PAMI*, 2002. 11
[67] Tu and S. C. Zhu. Image parsing: unifying segmentation, detection and recognition. In *IJCV*, 2005. 11
[68] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *ICCV*, 2005. 24

7. Appendix

7.1. Proof of Theorem 1, Conclusion a

Let the inverse covariance matrix be $\Lambda = \Sigma^{-1}$. Then we have

$$g^{(t)}(x) = e^{-\frac{1}{2}(x-\mu^{(t)})^T \Lambda (x-\mu^{(t)})}$$

To simplifying notations we dropped the normalizing constant since it does not depend on μ. We will use the following inequality which holds for any u and v:

$$e^{u+v} \geq (1+u)e^v$$

Let us define: $\delta = \mu^{(t+1)} - \mu^{(t)}$, then:

$$g^{(t+1)}(x) = e^{-\frac{1}{2}(x-\mu^{(t+1)})^T\Lambda(x-\mu^{(t+1)})}$$
$$= e^{-\frac{1}{2}(x-\mu^{(t)}-\delta)^T\Lambda(x-\mu^{(t+1)}-\delta)}$$

Now using our inequality we have:

$$g^{(t+1)}(x) \geq (1 + (x-\mu^{(t)})^T\Lambda\delta - \frac{1}{2}\delta^T\Lambda\delta)g^{(t)}(x)$$

Since f is non-negative the inequality carries over to F:

$$F(\mu^{(t+1)}) \geq \int (1 + (x-\mu^{(t)})^T\Lambda\delta - \frac{1}{2}\delta^T\Lambda\delta)g^{(t)}(x)f(x)dx$$

Remembering that $\delta = \frac{\int (x-\mu^{(t)})g^{(t)}(x)f(x)dx}{\int g^{(t)}(x)f(x)dx}$ we have:

$$\int (x-\mu^{(t)})^T\Lambda\delta g^{(t)}(x)f(x)dx =$$
$$\frac{(\int (x-\mu^{(t)})g^{(t)}(x)f(x)dx)^T\Lambda(\int (x-\mu^{(t)})g^{(t)}(x)f(x)dx)}{\int g^{(t)}(x)f(x)dx} =$$
$$\delta^T\Lambda\delta \int g^{(t)}(x)f(x)$$

Substituting this into the initial inequality in F we obtain:

$$F(\mu^{(t+1)}) \geq \int (1 + \frac{1}{2}\delta^T\Lambda\delta)g^{(t)}(x)f(x)$$

This concludes the proof:

$$F(\mu^{(t+1)}) \geq \int g^{(t)}(x)f(x) = F(\mu^{(t)})$$

7.2. Proof of Theorem 1, Conclusion b

It can be easily shown that the partial derivative $\frac{\partial F(\mu,\sigma)}{\partial\sigma} = 0$ when σ is a fixed point of the update step 2 of the theorem, and thus satisfies the equation $\sigma = \sqrt{\frac{1}{n}\frac{\int (\sum i=1^n (x_i-\mu_i)^2)g(x;\sigma)f(x)dx}{\int g(x;\sigma)f(x)dx}}$. Also, it is straightforward to check that the update step 2 of the theorem is taken in the direction of the gradient. Therefore, conclusion b will be satisfied if the partial derivative mentioned above is never 0 in the interval between $\sigma^{(t)}$ and $\sigma^{(t+1)}$. (Without loss of generality we can assume that $\sigma^{(t+1)} > \sigma^{(t)}$).

We give here the sketch of a proof by *reduction ad absurdum*. Let us assume that there exists $\sigma^* \in (\sigma^{(t)}, \sigma^{(t+1)})$ such that

$$\sigma^* = S(\sigma^*) = \sqrt{\frac{1}{n}\frac{\int (\sum_{i=1}^n (x_i-\mu_i)^2)g(x;\sigma^*)f(x)dx}{\int g(x;\sigma^*)f(x)dx}} \tag{25}$$

To simplify notations, we omit μ, which remains constant during this step. Here $S(\sigma)$ is basically the update function; it tells us which is the next sigma given σ. From the assumption made it is clear that $S(\sigma^*) = \sigma^*$ and $S(\sigma^{(t)}) = \sigma^{(t+1)}$, while $\sigma^* \in (\sigma^{(t)}, \sigma^{(t+1)})$. It follows that:

$$\frac{S(\sigma^*) - S(\sigma^{(t)})}{\sigma^* - \sigma^{(t)}} = \frac{\sigma^* - \sigma^{(t+1)}}{\sigma^* - \sigma^{(t)}} < 0 \tag{26}$$

From the intermediate value theorem it follows that the derivative of S with respect to σ has to be negative somewhere inside $(\sigma^{(t)}, \sigma^{(t+1)})$. That means there exists a point σ_- where:

$$\int (\sum_{i=1}^{n}(x_i - \mu_i)^2)^2 g(x;\sigma_-)f(x)dx \int g(x;\sigma_-)f(x)dx -$$
$$(\int (\sum_{i=1}^{n}(x_i - \mu_i)^2) g(x;\sigma_-)f(x)dx)^2 < 0$$

But this is impossible by Cauchy-Schwarz inequality, which gives us the contradiction that concludes the proof.