5-16-2011

Holistic Shape-Based Object Recognition Using Bottom-Up Image Structures

Praveen Srinivasan
University of Pennsylvania, psrin@seas.upenn.edu

HOLISTIC SHAPE-BASED OBJECT RECOGNITION USING BOTTOM-UP IMAGE STRUCTURES

Praveen Srinivasan

A DISSERTATION

in

Computer and Information Science

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

2011

Supervisor of Dissertation	Graduate Group Chairperson
Jianbo Shi	Jianbo Shi
Associate Professor	Associate Professor
Computer and Information Science	Computer and Information Science

Dissertation Committee

Kostas Daniilidis	Professor of Computer and Information Science
Camillo J. Taylor	Associate Professor of Computer and Information Science
Pedro Felzenszwalb	Associate Professor of Computer Science, Univ. of Chicago
Ben Taskar	Assistant Professor of Computer and Information Science

HOLISTIC SHAPE-BASED OBJECT RECOGNITION
USING BOTTOM-UP IMAGE STRUCTURES

Acknowledgements

No thesis is ever completed in isolation. While many people have played an important role in my thesis, there are some who deserve special recognition. My parents, Sampurna and Cidambi Srinivasan, without whose care I never would have come this far, deserve first billing. From the time I took my first step to receiving my first diploma, they have been there for me, unwavering in their love and support. My sister Priyanka, Aunt Uma, Uncle Jayaraman, and my cousins Arun and Vivek round out my family supporters.

Of course, without a good advisor research can be aimless and mediocre. Thus I also thank deeply Jianbo Shi, my advisor. His blunt feedback, good and bad, has critically shaped both my work and my work ethic, pushing me to never settle for easy answers, to never blindly trod the well-worn path. Also worth mentioning are Ramesh Gupta and Drago Anguelov, who provided me with some of my first experiences in academic research.

An ideal academic environment is one with free exchange of ideas, criticisms and encouragement. Only with a diverse and committed set of fellow students can such an ideal be fulfilled. My colleagues Timothee Cour, Qihui Zhu, Weiyu Zhang, Katerina Fragkiadaki, Roy Anati, Gang Song, Ben Sapp, Alex Toshev, Elena Bernardis, Sandy Patterson, Abhinav Gupta, Yang Wu, Liming Wang, and Spring Berman to name just a few, have all contributed. Late night discussions, last-minute deadline pushes and helpful favors from them have also made my life much easier. Qihui deserves special mention for having done important groundwork in many-to-many matching

and extraction of image contours, and Weiyu has been a tireless worker in pushing the efforts in this thesis still further.

My committee members, Kostas Daniilidis, Ben Taskar, C.J. Taylor and Pedro Felzenszwalb were very helpful in shaping my thesis work and therefore I am indebted to them as well.

ABSTRACT

HOLISTIC SHAPE-BASED OBJECT RECOGNITION USING BOTTOM-UP IMAGE STRUCTURES

Praveen Srinivasan

Jianbo Shi, Associate Professor of Computer and Information Science

Object recognition performance that rivals human ability is one of the primary goals of computer vision research. While recognition may take many forms, key tasks include detection, estimating object pose, and segmenting the object from the background. This thesis explores the use of holistic shape matching for recognition using bottom-up image structures such as image segments and contours for all of these tasks. Holistic shape matching utilizes global information about object shape for matching, rather than local image features which often contain too little information to match reliably to the object model. By examining different tasks related to object recognition, we demonstrate the value of holistic shape matching in a broad range of problems, including perceptual grouping, human pose estimation, and object recognition.

First, we introduce a method for perceptual grouping of contours in an image into larger groups that uses holistic shape matching to estimate the motion of image contours to a second, related image and group them according to similarties in motion. Holistic shape matching provides scoring for different motion hypotheses, and the final grouping is achieved using a min-cut graph cut to infer the cluster assignment for each contour.

Secondly, we describe a method for human pose estimation using image segments that incrementally merges segments into hypotheses for increasingly larger regions of the human body. These hypotheses are verified by matching against a set of shape exemplars using a shape matching method that is articulation-invariant and incorporates holistic shape information.

Lastly, we present a two-step method for automatically learning an object detector

for an object category from positive and negative images annotated with bounding boxes. In the first step, the object shape is learned from bottom-up image contours extracted in the positive images by searching for "lucky" contours that can explain large portions of the shape of positive examples. Given the learned shape, the second step trains a discriminative object detector that matches the shape against contours, emphasizing shape features that provide good detection performance. We compare against baselines and previous work that do not use holistic evaluation of shape features to demonstrate its value.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Object Recognition

A long-held goal of computer vision has been recognition of a wide variety of objects in complex, cluttered images, an everyday task humans perform with ease. Although there are many different tasks that comprise the recognition problem, three important tasks (Figure 1.1) include detection, alignment and segmentation:

- Detection: indicating the presence or absence of an object at a particular location in the image.
- Alignment: determining the pose of an object by corresponding it to a shape model.
- Segmentation: in order to understand how to manipulate or interact with an object, we must be able to determine the boundaries of the object.

There are many different applications that can benefit from accurate object recognition, including robot navigation, web image search, video analysis and medical image understanding. However, despite substantial progress, the problem remains unsolved. One promising area of focus concerns the use of object shape to recognize objects. Shape is a critical cue for recognition, as it is sufficiently invariant to

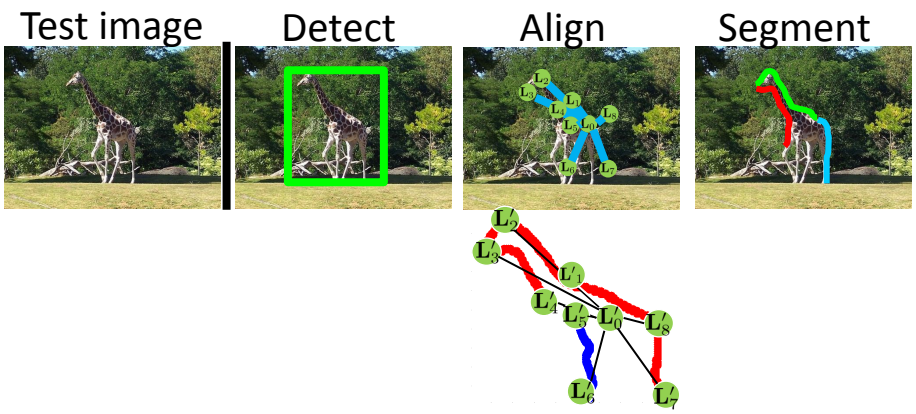


Figure 1.1: Three goals of recognition: detection, alignment and segmenting object boundaries.

represent commonalities of different instances of a particular object category, while preserving enough detail about objects in order to differentiate them from each other or the background. It also varies systematically with 3D viewpoint, enabling estimation of the object pose from shape, and segmentation is exactly determining the object shape boundary. While there are many different approaches to using object shape for recognition, there are two difficulties faced by nearly all approaches: object pose variation and the presence of background clutter.

- Background clutter: nearby regions or edges in the image may belong to objects in the background. Shape feature descriptors computed on the boundary of the object can therefore be corrupted by these background objects.

- Object pose variation: many objects, such as articulated objects (e.g. humans), or deformable objects vary in appearance according to pose. As a result, the relative positions of important shape features may vary substantially.

Given these challenges, we can characterize the recognition problem into four categories based on their presence or absence:

No pose variation, no background clutter: Object shape features appear clearly in this scenario, and discriminatively trained object detectors can identify clear foreground shape features to detect objects.

Pose variation, no background clutter: When only pose variation is present, bottom-up image segmentation can yield clear foreground segments with large portions of the object boundary shape. These segments can be used to infer the articulated pose of the object and achieve articulation invariant recognition.

No pose variation, background clutter: If background clutter is present without pose variation, relative spatial relationships between foreground shape features and background clutter areas remain consistent. Therefore, discriminatively trained detectors can emphasize areas that consistently contain foreground object shape features while simultaneously ignoring areas that consistently contain background clutter.

Pose variation, background clutter: This case is in a separate category from the rest, as the spatial relationships of shape features and clutter vary along with the object pose. Therefore, bottom-up image segmentation fragments unpredictably, while the relative positions of shape features and background clutter vary with object pose. As a result, none of the previous assumptions necessary for applying the aforementioned strategies are satisfied.

In complex, real-world images, the last case is the norm and not the exception. Focusing on this setting of recognition, we can analyze different approaches to shape-based object recognition in terms of how they address these two problems.

1.2 Shape for recognition

There are three important areas that must be addressed in using shape for object recognition: representation of the shape model, shape features used for matching, and the method of matching the shape features with the image. All of these choices

come with different trade-offs among computational efficiency, tractability of good approximate or exact inference, and learnability of good cost functions for recognition, as well as addressing the recognition challenges outlined above.

1.2.1 Model Representation

On the model side, there are many different methods for representing object shape. Broadly speaking, representations in the early history of computer vision research tended towards greater abstraction, representing objects in terms of high-level concepts such as geometric shapes (2-D primitives such as lines and curves, and 3-D primitives such as rectangular prisms and conic sections) or other semantic categories (e.g., an airport is composed of a runway and a terminal building). Unfortunately, the semantic gap between these abstract representations and the image pixels has proven to be difficult to bridge directly. As a result, vision research in recent years has focused on much simpler template representations of objects. Templates do not capture the same general properties of object shape, but are much easier to compare against image features than abstract representations. However, because they are not as general template models are not as compact as abstract representations (e.g., for viewpoint invariant recognition, Basri et al. [4] used multiple 2-D templates). We discuss examples of the two types of model representations:

Abstract Representations

Generalized cylinders: Consisting of a set of cross-sectional shapes and a path that joins their centers of gravity, the generalized cylinder is the shape formed by resulting volume formed from the cross-sectional shapes. Proposed by Binford ([9]), generalized cylinders can represent a wide variety of different object shapes.

Geons: Biederman ([8]) proposed geons, a set of simple 3-D geometric shapes that can be combined together to form a wide variety of 3-D object shapes, similar to the children's puzzle, tangrams, in 2-D. There are three important characteristics that

all geons possess - view-invariance: each geon can be identified when seen from any viewpoint; stability: under occlusion or deformation, geons are still recognizable; and discriminability: two different geons can be visually distinguished from one another, regardless of viewpoint or occlusion.

AND-OR Graphs: To address some of the issues of previous abstract models, AND-OR graphs [13] were proposed as an object model that contains multiple levels of abstraction, from the image pixels all the way up to abstract parts of objects. Specifically, AND-OR graphs are a type of graph with two types of nodes: AND nodes and OR nodes. AND nodes represent inclusion of all children nodes as part of the object, while at each OR node, one child is selected to be part of the object. As a result, AND-OR graphs can compactly represent substantial object variation in a single model. For example, a chair may be composed of a back, base and bottom; each of these can have multiple different appearances depending on the specific type of chair, and each may decompose further into object subparts.

Template Representations

Deformable 2D templates: A simple form of object representation, the 2D template may consist of a set of contours or weights on histogram bins (a histogram computed on either the outline of the shape or gradients of image examples of the object) to represent the shape of an object from a single view, in a single pose. By combining a collection of 2D templates of an object imaged from different viewpoints, 3D object recognition can be achieved as demonstrated by Basri [4].

Articulated templates: For articulated objects, a simple deformable template is insufficient to capture all of the possible deformations of the object. Instead, a model that explicitly takes into account the articulated nature of the object can help. The most popular example is the pictorial structures model of Fischler & Elschlager [27], popularized by Felzenszwalb and Huttenlocher [24], which consists of a set of rigid object parts that are related by pairwise geometric potentials that penalize

non-articulated deformations. An efficient inference method for pictorial structures models was proposed in [24] and was used to estimate the pose of the human body (locating the limbs) and faces (locating the eyes, nose mouth).

In this thesis, we focus on template representations of objects, using bottom-up image structures as a mid-level representation between the object model and the image. Future work would include introducing increasingly abstract model representations using perceptual structures such as image segments, contours and junctions as a mid-level representation between the abstract representations and the image.

1.2.2 Features

In order to use object shape for recognition, we require a method for computing statistics about the shape that can be compared with the image in order to find shape matches in the image. These image features allow us to compare the template models of objects against bottom-up image structures. Various shape features have been proposed to address this problem:

Shape context: Belongie et al. ([5]) introduced the shape context descriptor for matching of rigid templates in 2-D images. The shape context is a log-polar spatial histogram over the location of edges in an image. Smaller bins near the center of the descriptor capture local, precise shape details, while larger bins further away capture more general statistics about the overall object shape. Multiple shape contexts can be computed at different locations in the image and model, and then these shape contexts can be matched by a variety of methods, including the Hungarian method. However, clutter may corrupt the descriptor in complex scenes, resulting in poor match scores despite the actual object of interest being present.

Inner-distance shape context: Related to the shape context, the inner-distance shape context (IDSC) proposed by Ling et al. ([41]) is again a histogram over the locations of object boundary points, but is computed in a way as to be invariant to articulation using shortest paths between two points on the boundary of the shape

through the interior. The length of this path and the local orientation of the object shape at each of the boundary points are used to compute an articulation-invariant descriptor (described in further detail in Chapter 2). The IDSC was used for shape matching of silhouette images for shape retrieval in [41], and was also used for human pose estimation in [62].

HOG feature: The Histogram of Oriented Gradients, or HOG feature ([18]), is a histogram over gradient orientations in a particular region of the image. These gradients can capture local shape features of an object, for example the shape of a person's head, or the appearance of a body limb. Similar to the shape context, the descriptor can be corrupted by background clutter. For this reason, the descriptor support is typically very small relative to the overall object size, limiting the scale of shape features that can be represented. The descriptors are often scored using a set of linear weights learned discriminatively, e.g. from a support vector machine (SVM), that emphasize the important local features of object shape for good detection performance. The HOG feature was applied to pedestrian detection by [18], where weights on the individual features were learned via linear SVM from positive and negative instances of pedestrians in a dataset of outdoor street scenes.

1.2.3 Matching

Given shape features in the image, these features must be matched against the object shape model in order to achieve recognition. This typically involves at least alignment of the model to the image, and may also include segmentation of the object. Many different methods for shape matching have been developed, implementing a variety of different cost functions with corresponding trade-offs in computational complexity and detection accuracy.

Template Matching Using Local Features

Chamfer matching/Distance transform: Given a template representation of an object consisting of a set of points that represent the object boundary, chamfer matching using the distance transform can be used to evaluate the matching of the template at a particular location in the image. A placement (represented by a translation) of a 2D object template in an image can be scored by computing the distance of each point in the template to the closest edge in the image, under the specific placement. These distances can be summed to provide a score for placing an object at a particular location in the image. The distance transform of [23] can be used to efficiently compute this score at all possible placements of the template in the image. While chamfer matching is a very fast method for computing a matching score at many locations in an image, clutter in the image can cause many false matches since the more image edges that are present, the more likely a point in the template will have an image edge nearby. Active shape models ([14]) have a similar cost function for matching, but allow for deformation of the model shape via a linear basis for the shape learned from training example shapes.

Local features + pairwise geometric constraints: Another simple method for shape matching is using local image features, computed by the similarity of a set of model weights with a set of image features extracted at a particular location in the image, measured by correlation. For example, the image features might be HOG features, while the weights may have been learnt through a discriminative procedure, as in [18]. Multiple object parts can be detected in the image using this method and their scores can be combined via voting for the object center using the known spatial relationships of the parts relative to the object center, as in [22]. Arbitrary pair-wise relationships can also be incorporated, as did Coughlan and Ferreira ([15]), using loopy belief propagation for inference.

Abstract Representation Matching

Interpretation tree: The interpretation tree, introduced by Gatson and Lozano-Perez in [29], is a formalism for structuring the search space of correspondences between an image and a set of object models. Each edge in the tree represents an additional correspondence of image points to the object model, and typically edges emanating from the ith level of the tree represent possible correspondences of the ith point to different points on the various object models. A leaf in the tree represents an interpretation, or assignment of image points to object models. Not all leaves represent valid interpretations; for example, in rigid object recognition from range images, there must exist plausible 3-D transformations that can align the object models with the corresponded image points. The interpretation tree is very general, and is capable of matching any image against virtually any object type of object model. Unfortunately the number of possible interpretations is in general exponential in the number of image points, making interpretation tree search (exploration of all possible leaves) impractical for complex scenes with many image points/object models.

Holistic Matching

Many-to-many matching: Bottom-up image segmentation can yield important image structures that are useful for object recognition. However, these image structures may fragment in unpredictable ways, resulting in no possible one-to-one correspondence between image structures and object parts. Several researchers have explored the concept of many-to-many matching as a way of dealing with this fragmentation problem. Demirci et al. [19] formulated the many-to-many matching problem between two graphs by first finding an embedding of nodes of each graph using a low-distortion graph embedding technique, followed by solving an Earth Mover's Distance (EMD; [55]) problem where the flows between nodes were interpreted as the many-to-many matching. They applied their method to match shock

graphs of silhouette images for shape matching.

Zhu et al. [73] developed an alternative approach for many-to-many matching based on linear programming. Given two sets of contours, the goal was to find a subset of contours in each that had similar shape. Shape similarity was measured by comparing shape contexts computed over the selected subsets of contours, and a computationally efficient approximation to this combinatorial problem was formulated as a linear program. The many-to-many matching was used to detect object parts in the image, which were then combined via a voting scheme to provide object detection scores. The approach was evaluated on the ETHZ Shape Classes dataset from [25], and showed good detection performance.

Many-to-one matching: Discussed further in Chapter 4, this thesis introduces a method for many-to-one matching of image segments to an object model, specifically for human pose estimation. For the human body, different shape exemplars were specified for different regions of the body. Because the human body is compositional in nature, proposals for a particular body region were created by combining proposals from subregions. For example, to form a proposal for the lower body, a single segment could be taken, two proposals for legs could be combined, or a proposal for three-fourths of the lower body and a lower leg (the remaining one-fourth) could be combined. Because these proposals consist of image segments, a region of the body could be formed by combining one or more image segments together. Evaluation of the proposals was achieved by shape matching of these proposals to shape exemplars via the inner-distance shape context to achieve articulation invariant matching.

1.3 Layout and Contributions

From the above discussion, it is clear there are many different possible approaches to object recognition, using one or more of the previous ideas. In this thesis, we focus on holistic evaluation of object shape via many-to-one and many-to-many

matching of bottom-up image structures such as contours and segments to an object shape model. In particular, holistic evaluation of bottom-up image structures is well suited to addressing the challenges of pose variation and background clutter. We demonstrate the application of these ideas to a variety of different problems in recognition, as well as superior performance due to the use of holistic evaluation of shape:

- Chapter 2 reviews several different methods for holistic evaluation of object shape, including the works of Zhu et al. [73], many-to-many matching, and Ling et al. [41], articulation-invariant shape evaluation.

- Chapter 3 discusses perceptual grouping. We introduce a method for grouping of image contours into larger clusters using many-to-many matching to establish accurate correspondences, thereby estimating the motion of contours and grouping contours with similar motions into the same group. Final group assignment is achieved using a min-cut graph cut that derives its unary scores from the many-to-many matching score of a contour in one image to another under a specific motion hypothesis.

- Chapter 4 studies human pose estimation. In this chapter a method for estimating human pose using bottom-up parsing of image segments is described. The overall cost function for evaluating human pose hypotheses is non-additive, allowing for evaluation of holistic shape properties of groups of image segments that cannot be observed from the individual segments. Proposals for regions of the human body are evaluated by matching against shape exemplars using an articulation-invariant shape matching method; as a result, few exemplars are needed to be able to match a wide range of articulated human shapes.

- Chapter 5 addresses generic object recognition. For a specific object category, object shape is automatically learned from the image contours of positive examples and a shape-based object detector is trained discriminatively using the

learned shape model and the contours of positive and negative images. The result is a fully automatic system for learning an object detector that can detect a specific object category in a new image using bottom-up image contours. A key characteristic of both the shape learning as well as the object detection is the use of many-to-one matching for holistic shape evaluation of detection hypotheses as well as candidate image contours for constructing the model shape.

- Chapter 6 concludes with a summary of the work in this thesis as well as a discussion on directions for future work.

Chapter 2

Background

2.1 Bottom-up Image Structures

One of the key themes of this thesis is the use of bottom-up image structures for grouping and recognition. Such structures naturally capture long-range grouping constraints between pixels or edges that can constrain and improve both tasks. There are two primary types of structures, image segments and image contours. An image segment consists of a subset of the pixels in the image, typically contiguous, and may often correspond to a portion of an object. For extracting segments, we use the method of Cour et al. ([16]), which uses the normalized cuts algorithm ([57]) to partition a graph over image pixels into multiple regions. The graph includes both short-range and long-range connections, where connection weights are determined using the intervening contour cue (the magnitude of the strongest edge that lies between two pixels). Multiple image segmentations can be produced by varying parameters such as the number of segments produced by the segmentation method, representing different hypotheses for groupings of image pixels.

An image contour is an ordered set of edge pixels in an image. Long contours often contain important information about the figure-ground boundary of an object or its interior shape. To extract contours, we use the untangling cycles method of Zhu

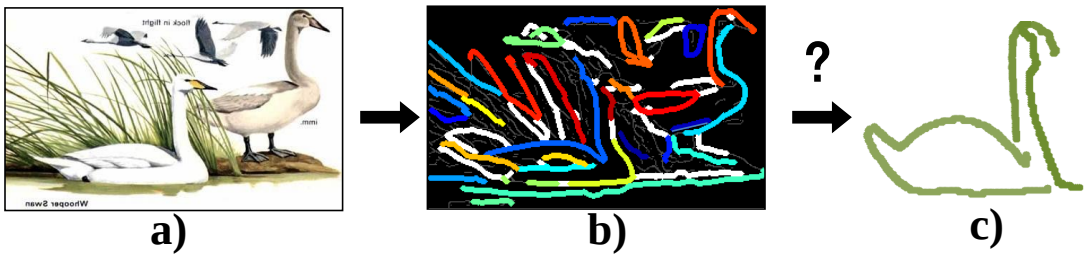


Figure 2.1: Recognizing an object by shape using image contours requires matching a subset of image contours against an object model. From an image (a) bottom-up contours are extracted (b), which then need to be matched against a shape model (c). Most of the image contours do not correspond to the object of interest, and those that do fragment unpredictably.

et al. ([72]). The approach first begins with edge detection in the image, keeping edges above a low threshold on the edge strength. A directed graph is formed over these edges, where the graph weights are determined by geometric consistency of neighboring edges (edges with similar orientations have stronger connections). Complex eigenvectors are computed from the weighted adjacency matrix for the graph, and for each of these eigenvectors, cycles are traced in the complex plane to find individual contours. The resulting contours may overlap, representing different grouping hypotheses at junctions in the image.

2.2 Matching Image Structures

Given a set of image structures, recognition typically requires matching these structures against an object model. For example, Figure 2.1 shows an image with extracted contours that we wish to match against an outline model of the object. There has been substantial work on this topic, but we focus here on two types of methods for matching, one-to-one matching and many-to-one matching, that have become popular for addressing the matching problem.

2.2.1 One-to-one Matching

One of the most common approaches to matching is one-to-one matching, where each model structure is matched to at most one image structure. There have been many approaches ([17, 40, 65, 31, 5]) which minimize similar cost functions that typically consist of unary terms indicating the direct compatibility of a match, and pairwise terms indicating the compatibility of pairs of matches. Because of the nature of these cost functions, only local relationships relating to a small region of the object or pairs of small regions are captured. Holistic characteristics of the object shape are not easily captured with one-to-one matching cost functions. Figure 2.1 illustrates an example of one-to-one matching being insufficient for matching image contours to an object shape model. Because the contours of different instances of a particular object category may fragment very differently in the image, there is no one-to-one correspondence of these contours to the object model, and keeping around all possible fragmentations of object contours is intractable.

2.2.2 Many-to-Many Matching

Because one-to-one matching is insufficiently flexible to handle the matching of bottom-up structures that fragment unpredictably, researchers have developed methods for many-to-many matching. A many-to-many matching maps subsets of a set A to subsets of a set B. The advantage of many-to-many matching is that groups of image structures can be holistically matched to the object model without regard to their specific fragmentation. The contours corresponding to the outline of the object in the image could be fragmented arbitrarily, yet many-to-many matching would be able to match them to the object shape model with the same cost. Demirci et al. [19] formulated the many-to-many matching problem between two graphs by first finding an embedding of nodes of each graph using a low-distortion graph embedding technique, followed by solving an Earth Mover's Distance (EMD) problem where the flows between nodes are interpreted as the many-to-many matching. They applied

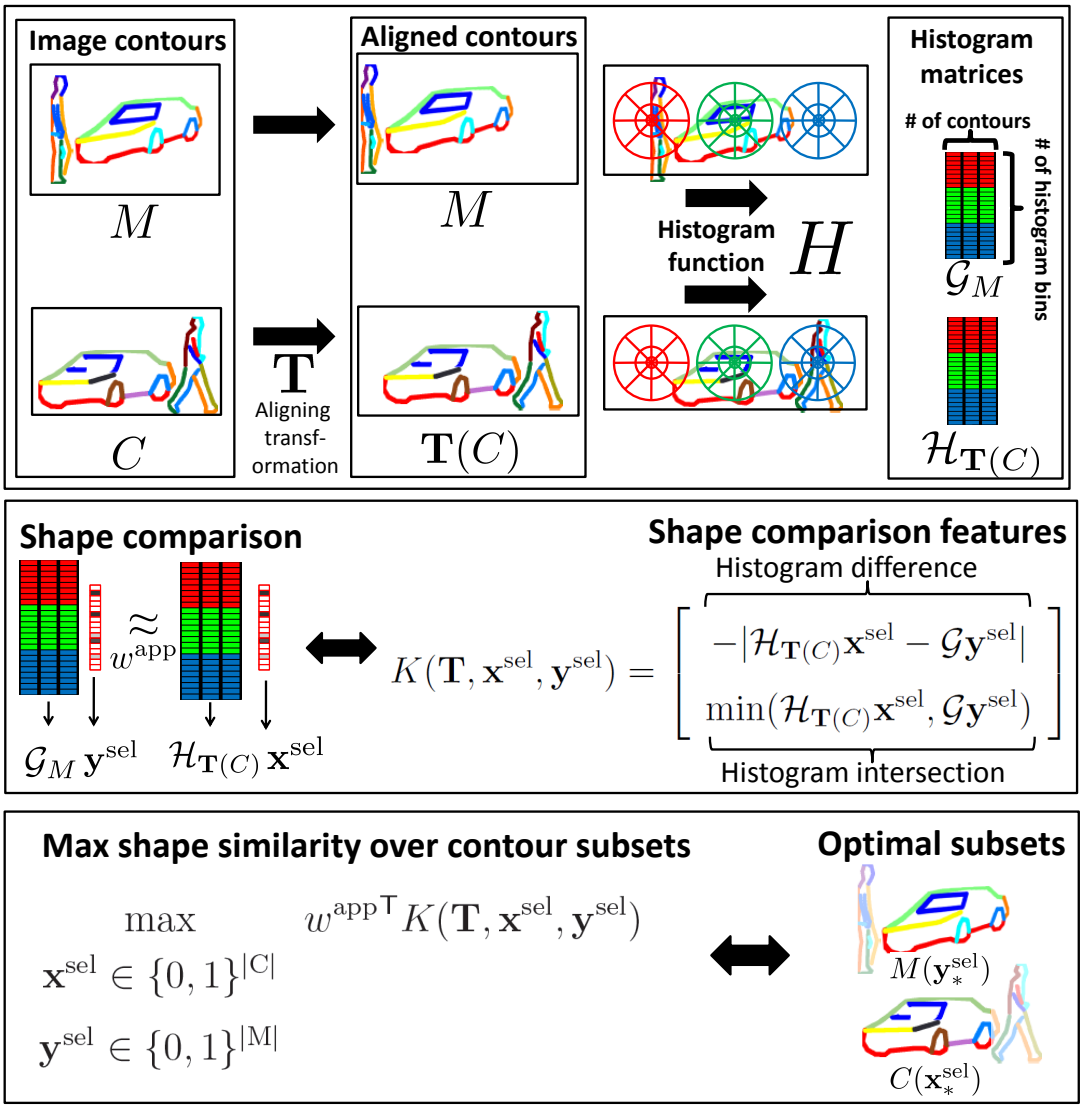


Figure 2.2: Overview of the many-to-many matching process. Top: two sets of contours M and C are provided as input. An aligning transformation $\mathbf{T}$ transforms contours C such that some object(s) align between the two sets of contours. A histogram function H operates on the contours M and transformed contours $T(C)$, producing a histogram for each contour, which appears as a column of matrices $\mathcal{G}_M$ and $\mathcal{H}_{\mathbf{T}(C)}$. Middle: our goal is to infer indicator vectors $\mathbf{x}^{\text{sel}}, \mathbf{y}^{\text{sel}}$ that specify a specific subset of contours in the two sets such that the two subsets have similar histograms (and hence shape). To compare histograms, we use histogram comparison features $K(\mathbf{T}, \mathbf{x}^{\text{sel}}, \mathbf{y}^{\text{sel}})$, a function of the transformation $\mathbf{T}$ and the contour subsets. Bottom: our goal is to maximize the similarity of the two histograms over the choices of subsets of contours to match the contours of the common aligned object (a car in this instance). The two quantities $\mathbf{x}_*^{\text{sel}}$ and $\mathbf{y}_*^{\text{sel}}$ together are the optimal solution (subsets of image contours) to the many-to-many matching problem.

their method to match shock graphs of silhouette images for shape matching.

Zhu et al. [73] developed an alternative approach for many-to-many matching based on linear programming. Given two sets of contours, the goal is to find a subset of contours in each that had similar shape. Shape similarity was measured by comparing shape contexts computed over the selected subsets of contours, and a computationally efficient approximation to this combinatorial problem was formulated as a linear program. The many-to-many matching was used to detect object parts in the image, which were then combined via a voting scheme to provide object detection scores. The approach was evaluated on the ETHZ Shape Classes dataset from [25], and showed good detection performance. An advantage of this approach over [19] is that it does not require explicit specification of a distance between contours in the two sets, which can be difficult to specify when one contour overlaps only partially with the other.

2.2.3 Many-to-Many Matching Formulation

Following Zhu et al. [73], we formulate a computational solution to the many-to-many matching problem for matching object shape. In the contour setting, we are given a set of model contours M and image contours C, and wish to find a subset of each such that the overall shapes of the two subsets is similar. To characterize the shape of the subsets, we can use any spatial histogram, such as one or more shape contexts [6], or a grid histogram. Figure 2.2 shows an example with several different shape contexts used together as a single histogram. During matching, we must find both the subsets of contours in the model and the image as well as an aligning transformation that aligns the image contours to the object shape model so that shape similarity can be measured accurately. These quantities can be defined as:

- $\mathbf{T} \in \mathbb{R}^2$: a transformation that describes the alignment of the image contours to the model contours.

- $\mathbf{x}^{\text{sel}} \in \{0,1\}^{|\text{C}|}$: an indicator vector that defines which image contours are **selected** for matching to the model. Contour C_i is selected if and only if $\mathbf{x}_i^{\text{sel}} == 1$.

- $\mathbf{y}^{\text{sel}} \in \{0,1\}^{|\text{M}|}$: an indicator vector that defines which model contours are **selected** for matching to the image. Contour M_i is selected if and only if $\mathbf{y}_i^{\text{sel}} == 1$.

Figure 2.2, top, shows a set of contours in two images as input to the matching; an aligning transformation $\mathbf{T}$ aligns the two sets of contours. We define a spatial histogram of dimension d_m over the edge points of image contours selected by $\mathbf{x}^{\text{sel}}$ and transformed by $\mathbf{T}$ as: $h_{\mathbf{T}(C),\mathbf{x}^{\text{sel}}}$. Any type of spatial histogram is allowed, including grid histograms (as used in [18]) or log-polar radial histograms (as in [6]). We encapsulate this property via a histogram function H, which maps a point in $\mathbb{R}^2$ to a vector in $\mathbb{R}^{d_m}$, or $H : \mathbb{R}^2 \rightarrow \mathbb{R}^{d_m}$. In general, any possible histogram is permitted as long as it satisfies the following property: given two sets R and S and the histogram function H, we require: $H(R \cup S) + H(R \cap S) == H(R) + H(S)$. Specifically in this case, a histogram over the points of several contours is equivalent to summing the histograms computed for each contour individually, first noted in [73], and also depicted in Figure 2.3. This means that histogram $h_{\mathbf{T}(C),\mathbf{x}^{\text{sel}}}$ can be represented as a linear function of $\mathbf{x}^{\text{sel}}$ as shown in Figure 2.3. We introduce the per-contour histogram matrix $\mathcal{H}_{\mathbf{T}(C)}$ (Figures 2.2 and 2.3) and write the histogram over selected-contours $h_{\mathbf{T}(C),\mathbf{x}^{\text{sel}}}$ as a linear function of $\mathbf{x}^{\text{sel}}$:

$$\mathcal{H}_{\mathbf{T}(C)} \in \mathbb{R}^{d_m \times |\text{C}|} \quad h_{\mathbf{T}(C),\mathbf{x}^{\text{sel}}} \Longleftrightarrow \mathcal{H}_{\mathbf{T}(C)}\mathbf{x}^{\text{sel}} \tag{2.1}$$

The k-th column of $\mathcal{H}_{\mathbf{T}(C)}$ is a histogram over the points in contour C_k. Similarly, we can also represent the model contour shape contexts with a matrix $\mathcal{G}_M$, where each column is a shape context for a model contour. To compare the histograms that result from selecting only a subset of the image and model contours, we measure two

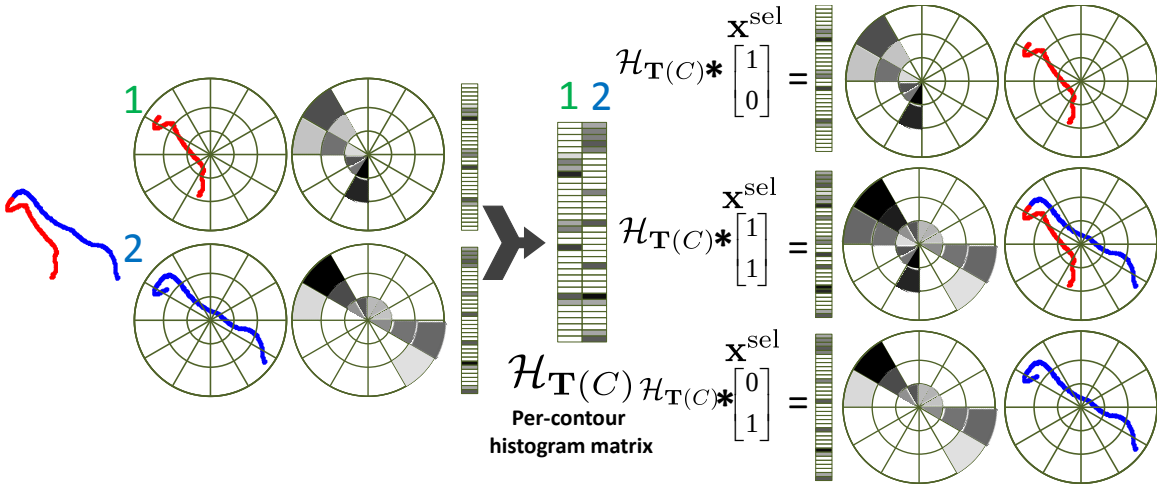


Figure 2.3: Encoding of contours for many-to-one matching. Left, two contours, are shown with associated histograms (any histogram, grid or shape context, is possible), which are combined to form matrix $\mathcal{H}_{\mathbf{T}(C)}$ (center). Right, different choices of selection vector $\mathbf{x}^{\text{sel}}$ lead to different histograms; $h_{\mathbf{T}(C),\mathbf{x}^{\text{sel}}}$ is thus a linear function of $\mathbf{x}^{\text{sel}}$.

types of features: bin-wise difference features $-|\mathcal{H}_{\mathbf{T}(C)}\mathbf{x}^{\text{sel}} - \mathcal{G}_M\mathbf{y}^{\text{sel}}|$ and intersection features $\min(\mathcal{H}_{\mathbf{T}(C)}\mathbf{x}^{\text{sel}}, \mathcal{G}_M\mathbf{y}^{\text{sel}})$.

$$K(\mathbf{T}, \mathbf{x}^{\text{sel}}, \mathbf{y}^{\text{sel}}) = \begin{bmatrix} -|\mathcal{H}_{\mathbf{T}(C)}\mathbf{x}^{\text{sel}} - \mathcal{G}_M\mathbf{y}^{\text{sel}}| \\ \min(\mathcal{H}_{\mathbf{T}(C)}\mathbf{x}^{\text{sel}}, \mathcal{G}_M\mathbf{y}^{\text{sel}}) \end{bmatrix} \tag{2.2}$$

Figure 2.2, middle, shows the comparison of the two histograms resulting from choosing a subset of contours in both the model and image, and the features used for histogram comparison. Given a weighting on these features $w^{\text{app}} \geq 0$, our goal is to solve the maximization problem:

$$\begin{aligned} \max \quad & w^{\text{app}\mathsf{T}} K(\mathbf{T}, \mathbf{x}^{\text{sel}}, \mathbf{y}^{\text{sel}}) \\ \mathbf{x}^{\text{sel}} \in \{0,1\}^{|\mathrm{C}|} \\ \mathbf{y}^{\text{sel}} \in \{0,1\}^{|\mathrm{M}|} \end{aligned} \tag{2.3}$$

An important question is how to perform the above maximization over $\mathbf{T}, \mathbf{x}^{\text{sel}}, \mathbf{y}^{\text{sel}}$. For fixed $\mathbf{T}$, the resulting optimization problem is an integer linear program; if we

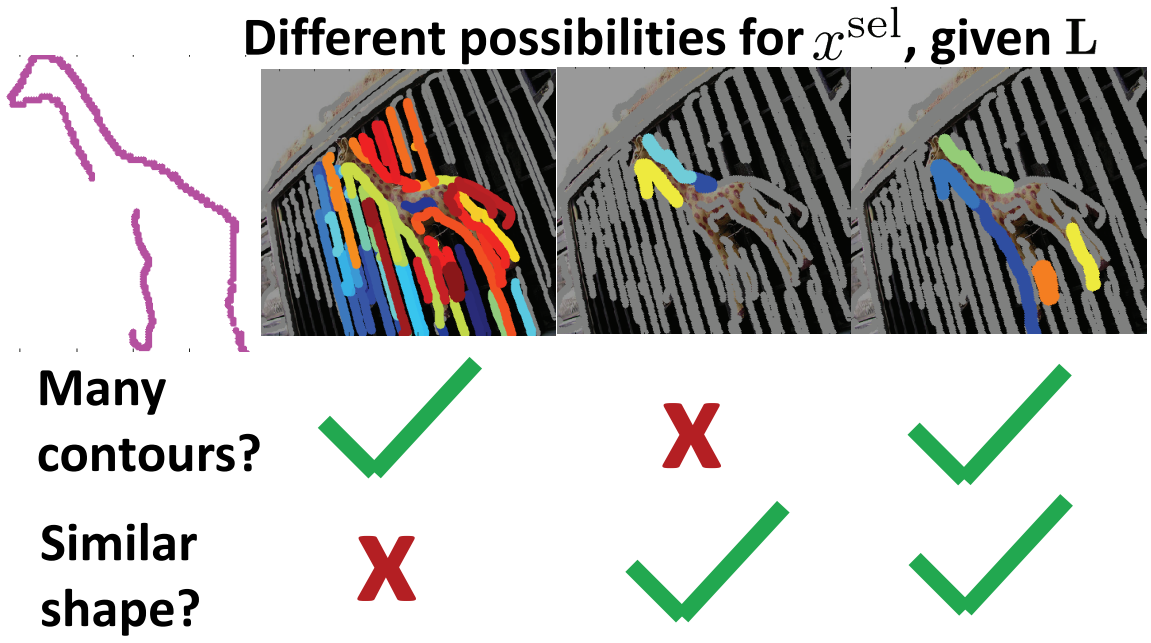


Figure 2.4: Matching score of many-to-one matching. For a given model shape (upper left; Giraffe) and placement $\mathbf{T}$, different selections $\mathbf{x}^{\text{sel}}$ of image contours are shown. Indicator $\mathbf{x}^{\text{sel}}$ encodes which image contours are many-to-one matched. Matching score prefers selections and placements that select many contours which have similar shape to the model.

can solve (or approximate) this integer linear program, we can directly search over different possible choices of $\mathbf{T}$, solving a separate optimization problem for each one. Instead of trying to solve the integer linear program exactly, we can relax $\mathbf{x}^{\text{sel}} \in [0,1]^{|\text{C}|}$ and $\mathbf{y}^{\text{sel}} \in [0,1]^{|\text{M}|}$, resulting in a linear program that can be solved efficiently via a linear program solver ([2]), as in Zhu et al. [73]. With the restriction of $w^{\text{app}} \geq 0$, this score function is concave and maximization is possible. We can write a linear program that maximizes a relaxation of our score function ${w^{\text{app}}}^{\mathsf{T}} K(\mathbf{T}, \mathbf{x}^{\text{sel}}, \mathbf{y}^{\text{sel}})$ using proxy variables m and o to represent the histogram difference $-|\mathcal{H}_{\mathbf{T}(C)}\mathbf{x}^{\text{sel}} - \mathcal{G}_M \mathbf{y}^{\text{sel}}|$ and intersection features $\min(\mathcal{H}_{\mathbf{T}(C)}\mathbf{x}^{\text{sel}}, \mathcal{G}_M \mathbf{y}^{\text{sel}})$ respectively:

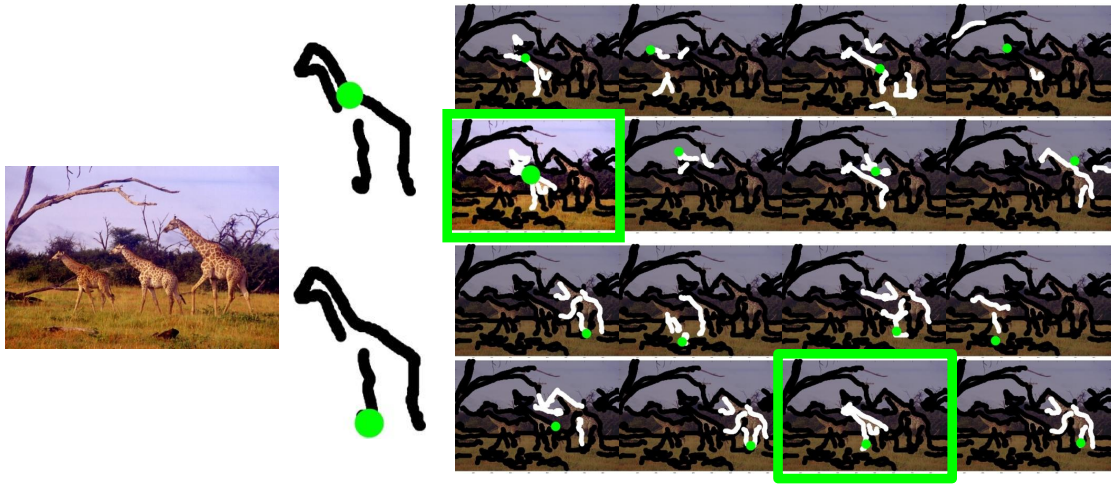


Figure 2.5: Examples of many-to-one matching. Left: input image; center: two different points on model to be matched in the image; right: different many-to-one matchings of model to image. A single shape context was used as the histogram, centered at the highlighted points on the model; the transformation $\mathbf{T}$ relating the image to the model was simply a translation of image contours derived from the relative locations of the model part point and the corresponding image point. Correct correspondences are highlighted in green; matched image contours are shown in white, and un-matched contours are black.

$$\begin{aligned} \max_{\mathbf{x}^{\text{sel}} \in [0,1]^{|C|}, \mathbf{y}^{\text{sel}} \in [0,1]^{|M|}} \quad & w^{\text{app}\mathsf{T}} \begin{bmatrix} m \\ o \end{bmatrix} \\ \text{s.t.} \quad & m \leq (\mathcal{G}_M \mathbf{y}^{\text{sel}} - \mathcal{H}_{\mathbf{T}(C)} \mathbf{x}^{\text{sel}}), (\mathcal{H}_{\mathbf{T}(C)} \mathbf{x}^{\text{sel}} - \mathcal{G}_M \mathbf{y}^{\text{sel}}) \\ & o \leq \mathcal{H}_{\mathbf{T}(C)} \mathbf{x}^{\text{sel}}, \mathcal{G}_M \mathbf{y}^{\text{sel}} \end{aligned} \tag{2.4}$$

Many-to-one Matching: An important special case of the many-to-many matching problem is the many-to-one matching problem. In this setting, instead of having multiple model contours, there is just one, which must always be matched (cannot be de-selected). The variables for model contour selection $\mathbf{y}^{\text{sel}}$ can be eliminated, and the term $\mathcal{G}\mathbf{y}^{\text{sel}}$ can be replaced with a single model histogram $h_{\mathcal{M}}$.

Figure 2.4 shows examples of different possible selections $\mathbf{x}^{\text{sel}}$ and how the many-to-one matching cost function behaves as a result, while Figure 2.5 shows examples of many-to-one matchings of different points on an object model to an input image.

2.2.4 Articulation-Invariant Matching

The previously described many-to-one matching handles only the case of largely rigid objects. However, many interesting object categories such as humans are articulated and therefore we require a different approach to matching bottom-up image structures to the object model in the case of articulation. The inner-distance shape context (IDSC) was proposed by Ling et al. ([41]) as a descriptor capable of addressing this issue. One way to use the IDSC is for matching bottom-up segments in an image against exemplar shapes of different regions of the body. Because the IDSC is largely invariant to articulation, image segments can still be matched to the exemplars even if the pose of the person in the image is different than that of the exemplars.

The IDSC is an extension of the original shape context proposed in [6]. In the original shape context formulation, given a contour of n points $x_1, ..., x_n$, a shape context was computed for point x_i by the histogram

$$\#(x_j, j \neq i : x_j - x_i \in bin(k)) \tag{2.5}$$

Ordinarily, the inclusion function $x_j - x_i \in bin(k)$ is based on the Euclidean distance $d = \| x_j - x_i \|_2$ and the angle $acos((x_j - x_i)/d)$. However, these measures are very sensitive to articulation. The IDSC replaces these with an *inner-distance* and an *inner-angle*.

The inner-distance between x_i and x_j is the shortest path between the two points traveling through the interior of the mask. This distance is less sensitive to articulation. The inner-angle between x_i and x_j is the angle between the contour tangent at the point x_i and tangent at x_i of the shortest path leading from x_i to x_j. Figure 2.6 shows the interior shortest path and contour tangent.

The inner-distances are normalized by the mean inner-distance between all pairs $\{(x_i, x_j)\}$, $i \neq j$ of points. This makes the IDSC scale invariant, since angles are also scale-invariant. The inner-angles and normalized log inner-distances are binned to form a histogram, the IDSC descriptor. For two shapes with points $x_1, ..., x_n$ and $y_1, .., y_n$, IDCSs are computed at all points on both contours. For every pair of points x_i, y_j, a matching score between the two associated IDCSs is found using the Chi-Square score ([6]). This forms an n-by-n cost matrix, which is used as input to a standard dynamic programming algorithm for string matching, allowing us to establish correspondence between the points on the two contours. The algorithm also permits occlusion of matches with a user-specified penalty. We try the alignment at several different, equally spaced starting points on the exemplar mask to handle the cyclic nature of the closed contours, and keep the best scoring alignment (and the score). The complexity of the IDSC computation and matching is dominated by the matching; with n contour points and s different starting points, the complexity is $O(sn^2)$.

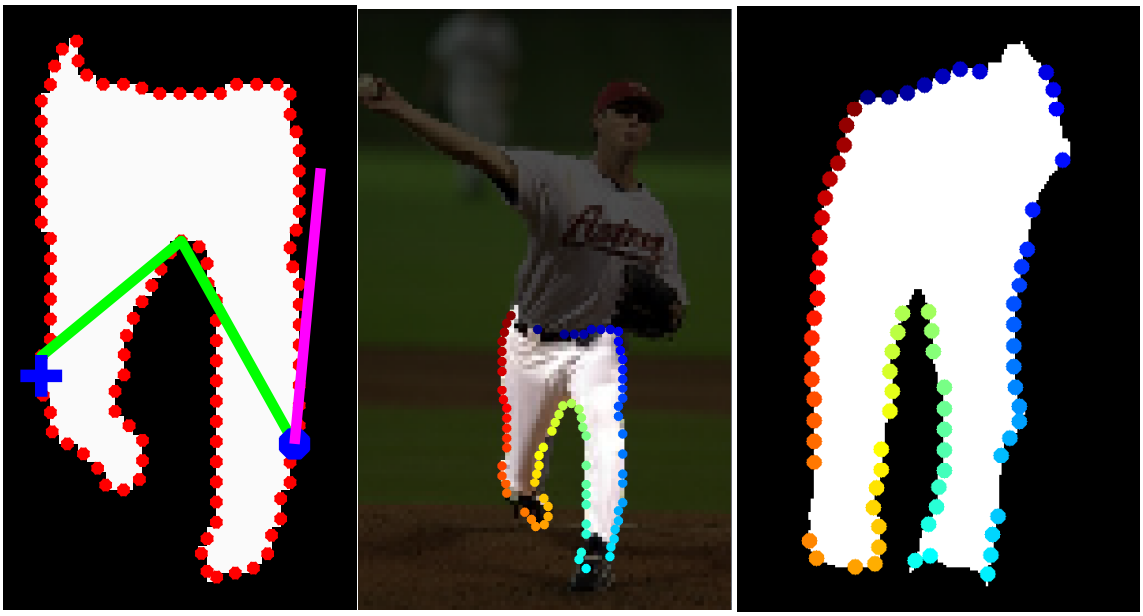


Figure 2.6: Inner-distance shape context computation. **Left:** We show: shortest interior path (green) from start (blue dot) to end (blue cross); boundary contour points (red); contour tangent at start (magenta). The length of interior path is the inner-distance; the angle between contour tangent and the start of the interior path is the inner-angle. **Center:** Lower body mask parse; colored points indicate correspondence established by IDSC matching with exemplar on **right**.

Chapter 3

Grouping with a Related Image

As mentioned in Chapter 1, bottom-up image structures are useful for addressing the pose variation and clutter challenges of object recognition. However, the process of bottom-up grouping is itself susceptible to corruption by these issues. For example, in segmentation by motion or stereo, correspondences between pairs of images must be established in order to estimate the motion of different parts of the image. Typically this is achieved using matching of local image patch features ([58, 67, 69]). As discussed previously, local features can be corrupted by background objects and object deformation and therefore be poorly matched between pairs of images. In this chapter, we introduce a method for perceptual grouping based on a pair of related images that addresses the issues of background clutter and object deformation using holistic evaluation via many-to-many matching of image contours between the images.

The pair of images may be a stereo pair, frames from a video, or two similar images (images containing similar objects). Relative motion of contours in one image to their matching contours in the other provides a cue for grouping - contours that undergo similar motion should be grouped together. The contours themselves are detected bottom-up without a model, and are provided as input to our method. While contours already represent a kind of grouping (of edges), they typically lack

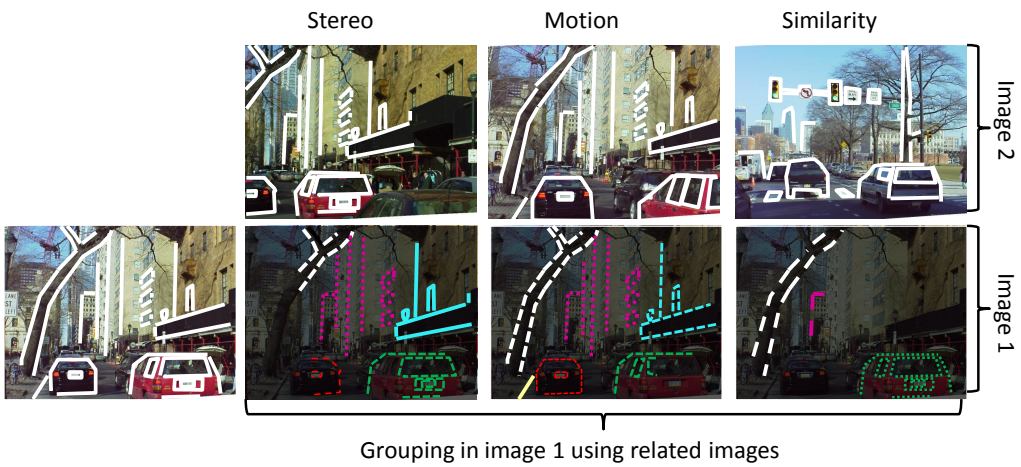


Figure 3.1: Contours (white) in the image on the left can be further grouped using the contours of a second, related image (top row). The bottom row shows idealized groupings in the original image according to the inter-image relationship.

large spatial support. Region segments, on the other hand, have large spatial support, but lack the structure that contours provide. Therefore, additional grouping of contours can give us both qualities. This has important applications for object recognition and scene understanding, since groups of contours are often large pieces of objects.

Figure 3.1 shows a single image in the 1st column, with contours; in the other columns, top row, are different images related by stereo, motion and similarity to the first, shown with their contours. Below each of these images are idealized groupings of contours in the original image. Note that internal contours on cars and buildings are grouped, providing rich, structured shape information over a larger image region.

3.1 Related Work

Stereo, motion, and similar image matching have been studied largely in isolation, and often with different purposes in mind than perceptual grouping. Much of the stereo literature focuses on per-pixel depth recovery; however, as [32] noted, stereo

can be used for perceptual grouping, with the advantage that precise depth estimation is not required to infer a good grouping. Motion is often used for estimating optical flow or dense segmentation of images into groups of pixels undergoing similar motion [68]. These approaches to motion and stereo are largely region-based, and therefore do not provide the same internal structure that groups of contours provide. Similar image matching has been used for object recognition [7], but is rarely applied to image segmentation.

In work on contours, Sherman and Peleg ([56]) matched contour points in the context of aerial imagery, but use constraints such as ordering of matches along scanlines and disparity gradient that are not appropriate for motion or similar images, and do not provide grouping information. Liu et al. [42] grouped image pixels into contours according to similar motion using optical flow as a local cue. While the result addresses the long-standing aperture problem in motion estimation, the framework does not extend to large inter-image deformations or matching across similar images. Hedau et al. [34] grouped and matched image regions across different images and unstable segmentations (as we do with contours), but the regions lack internal structure. Others ([10, 30]) used stereo pairs of images to detect depth discontinuities as potential object boundaries. However, these methods will not detect and group group contour points in the interior of fronto-parallel surfaces.

3.2 Grouping Criteria

We first present some definitions and our basic criteria for grouping contours. The inputs to our method are:

1. Images: $\mathbf{I_1}, \mathbf{I_2}$; for each image $\mathbf{I_i}$ we also have:

2. A set of points (typically image edges) $P_i \subset \mathbb{R}^2$. We restrict the set of points to those that lie on image contours, defined next.

3. A set of contours C_i, where $C_i^j \in C_i$ is an ordered subset of points in P_i: $C_i^j = [P_i^{k_1}, P_i^{k_2}, ..., P_i^{k_n}] \subseteq C_i$.

Our goal is to assign each contour in the first image I_1 to a group $1, ..., n$. For each contour C_1^j, we create a label variable $l_j \in \{0, 1, ..., n\}$, where $l_j == 0$ means the contour is ungrouped, and value $k \in \{1, .., n\}$ means the contour belongs to group k. The collective set of all label variables is denoted as L.
We would like the groups to possess the following criteria:

1. Good continuation - for contours that overlap significantly, we prefer that they are present in the same group, if they are grouped at all. For overlapping contours C_1^i, C_1^j, we prefer that $l_i == l_j$.

2. Common fate by shape: each contour point in a grouped contour should be matchable to a point in the second image where the local shape of the two points are similar. In addition, there should exist a single transformation that explains the motion of the contours in a group to the other image.

3. Maximality/simplicity: We would like to group as many of the contours as possible into as few groups as possible, while still maintaining the similarity of local shape described above.

We can write down a score function for this grouping problem in terms of a unary score ShapeMatchScore and pairwise score LabelSmoothness. The unary terms encapsulate our desire for common fate by shape, while the pairwise term captures the criteria of good continuation and maximality/simplicity for contours that overlap (represented by the set of pairs E; $(i, j) \in E$ if and only if contours C_1^i and C_1^j overlap; our goal is to maximize the score function $\mathbf{F}$:

$$\mathbf{F}(\mathrm{L}) = \sum_{C_1^i \in C_1} \text{ShapeMatchScore}(l_i) + \sum_{(i,j) \in E} \text{LabelSmoothness}(L_i, L_j) \tag{3.1}$$

The binary term LabelSmoothness(l_i, l_j) has two possible values, depending on whether or not the labels agree:

$$\text{LabelSmoothness}(l_j = a, l_k = b) = \begin{cases} 1 & a == b \\ 1 - \tau & a \neq b \end{cases} \tag{3.2}$$

If the two labels are the same, the score received is 1, while if they are different, the score received is $1 - \tau$ for some $0 \leq \tau \leq 1$; τ is specified by the user.

3.3 Shape Matching

The unary term ShapeMatchScore captures how well a particular contour in the first image can be matched in terms of shape to the second image. This requires knowing the underlying motion of the contour, correspondences of individual contour points to the second image, and correct context of shape with which to match the contours for each group. Specifically, we define for each group i:

- A transformation T_i that maps points (and hence contours) in image 2 to image 1: $T_i : \mathbb{R}^2 \to \mathbb{R}^2$.
- Sets of contours in each image that provide a context for shape matching, represented by indicator vectors $\text{Con}_1^i, \text{Con}_2^i$. $\text{Con}_1^i \in \{0,1\}^{|C_1|}$, and maps contours in image 1 to $\{0,1\}$: $\text{Con}_1^i : C_1 \to \{0,1\}$; Con_2^i is defined analgously for image I_2.
- Group-specific correspondences Corr_i^j of each point $P_1^j \in P_1$ to P_2: $P_2^{\text{Corr}_i^j} \in P_2$.

We describe how to propose each of these in turn from the input images and their contours, and then how these quantities combine together to form a group assignment score that is robust to object pose variation and background clutter using many-to-many contour matching.

3.3.1 Inferring Transformations

In general, there may be many different transformations that can align different objects between two images, and each motion can characterize a different one. Therefore, by proposing transformations, we are also proposing different groups. We begin with extracting and matching SIFT ([43]) features between the two images, giving a set of corresponding points $\{(a_i, b_i)\}$, where each a_i lies in the first image and each b_i lies in the second. Via RANSAC, we can find a transformation (homography in our case) with a maximum number of inliers. We add this transformation to our set of transformations, and remove all inliers from the set of correspondences. We then repeat the process until no transformation can be extracted with fewer than a pre-defined number of inliers. The result is a set of transformations $\{T_1, ..., T_n\}$, each of which is a transformation for a different group.

3.3.2 Many-to-Many Matching for Determining Contextual Shape Information

Given a transformation T_i that aligns image I_2 to image I_1, we can use many-to-many matching to find groups of contours that potentially move under that motion between the two images. Recalling our formulation of many-to-many matching from Chapter 2, we require two sets of contours, M and C, and an aligning transformation $\mathbf{T}$ that aligns contours C with M. Identifying M with the contours in image I_1, C_1, and C with the contours in image I_2, C_2, and the transformation $\mathbf{T}$ with the group specific transformation T_i, we can achieve many-to-many matching. Because we need to capture the shape of contours throughout the entire image, for the histogram we use evenly spaced shape contexts to measure local shape, as in Figure 2.2. Solving the many-to-many matching linear program results in selection indicator vectors $\mathbf{x}^{\text{sel}}$ and $\mathbf{y}^{\text{sel}}$, which we identify with our context indicator vectors Con_1 and Con_2 respectively.

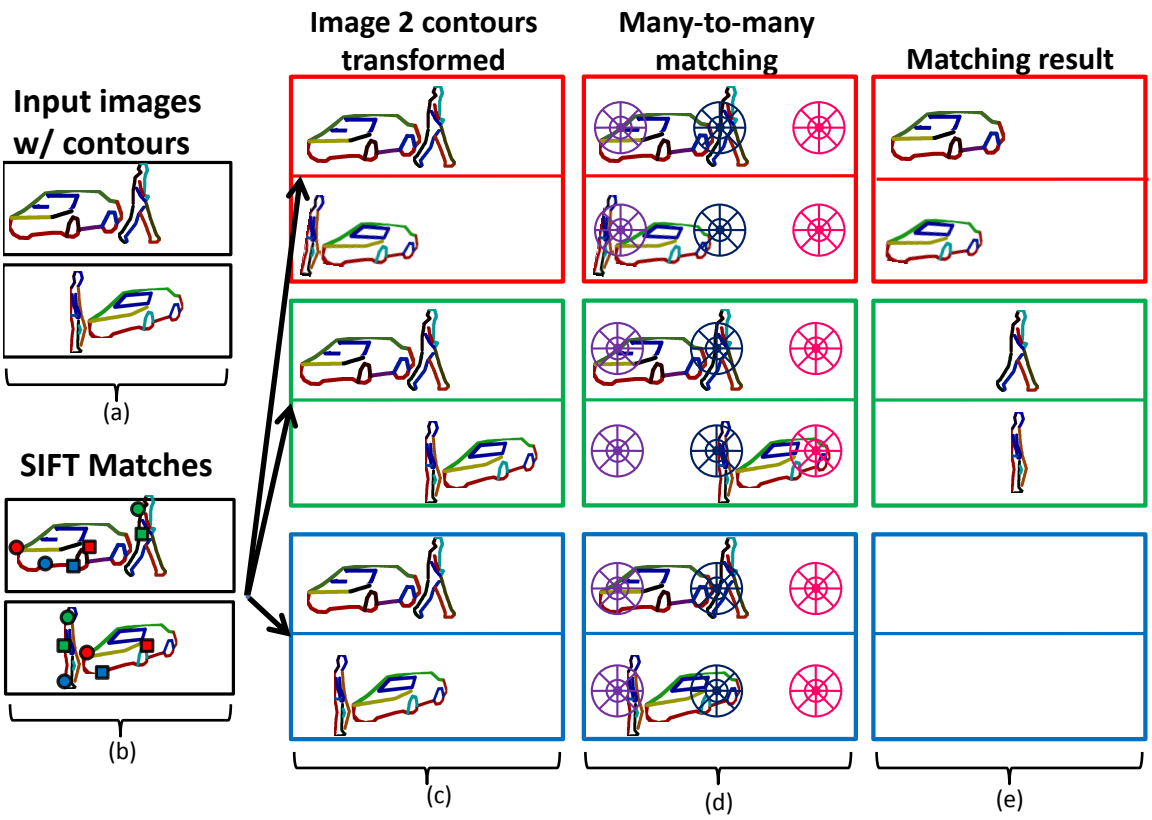


Figure 3.2: For a pair of images, SIFT matches propose different transformations of the contours in image 2 to align with contours in image 1. The many-to-many matching process is performed for each transformation to infer a context suitable for evaluating contour point correspondences via contextual shape information.

3.3.3 Correspondences using Contextual Shape

Different hypotheses for the group assignment of a point P_1^j in image I_1 lead to different hypotheses for the motion and hence correspondence of the point. Therefore, for each group hypothesis, we propose a different correspondence in the second image. The score of the best possible correspondence under a particular group assignment results in a score for the assignment of the point to that group, which contributes to the unary terms of contours that contain that point.

Given the particular contextual information $\text{Con}_1^i, \text{Con}_2^i$ for group i, we can find a best possible correspondence for a point P_1^j using the transformation T_i, which provides a general idea of the motion of the point, and the contextual shape information

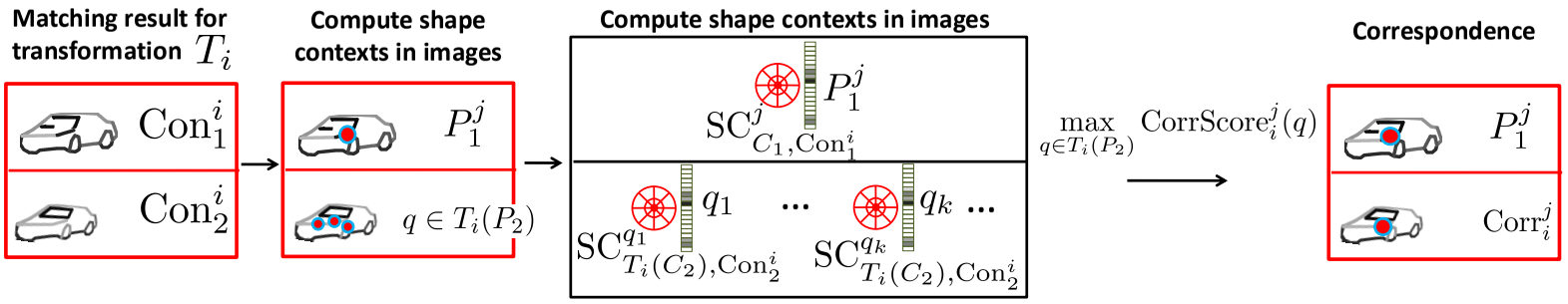
to evaluate different correspondence hypotheses. A single shape context centered at P_1^j in the first image can be computed to characterize the contextual shape information around the point. This shape context can be denoted as $\mathrm{SC}^j_{C_1,\mathrm{Con}_1^i}$ or the shape context computed for point P_1^j using contours selected from the first image via Con_1^i. Similarly, for a possible correspondence point $q \in T_i(P_2)$, we can compute a shape context centered at q over the transformed contours summarizing the shape information in the second image as: $\mathrm{SC}^q_{T_i(C_2),\mathrm{Con}_2^i}$. The similarity of these two shape contexts can be written in terms of the histogram difference and intersection features using a a weight vector $w^{\mathrm{app}} \geq 0$:

$$\mathrm{CorrScore}_i^j(q) = w^{\mathrm{app}\mathsf{T}} \begin{bmatrix} |\mathrm{SC}^j_{C_1,\mathrm{Con}_1^i} - \mathrm{SC}^q_{T_i(C_2),\mathrm{Con}_2^i}| \\ \min(\mathrm{SC}^j_{C_1,\mathrm{Con}_1^i}, \mathrm{SC}^q_{T_i(C_2),\mathrm{Con}_2^i}) \end{bmatrix} \tag{3.3}$$

In practice, we restrict the allowed correspondences to be within a neighborhood of P_1^j. Our goal is to find a corresponding point q in the second image that maximizes the above score. Using these scores for each point, we can compute the score for assigning a particular contour C_1^k to layer i as the sum of the scores of the points P_1^j contained within the contour:

$$\mathrm{ShapeMatchScore}(L_k == i) \propto \sum_{P_1^j \in C_1^k} \max_{q \in T_i(P_2)} \mathrm{CorrScore}_i^j(q) \tag{3.4}$$

Figure 3.3 shows an example of the correspondence process for a single point P_1^j in the first image under transformation hypothesis T_i. Shape contexts are used to characterize the contextual shape around P_1^j and potential correspondences using the matched image contours. The shape contexts are then compared using CorrScore to find a best correspondence. The resulting score is also used to compute the assignment score of the contour containing P_1^j to group i.



$$\mathrm{CorrScore}_i^j(q) = w^{\mathrm{app}\mathsf{T}} \begin{bmatrix} |\mathrm{SC}^j_{C_1,\mathrm{Con}_1^i} - \mathrm{SC}^q_{T_i(C_2),\mathrm{Con}_2^i}| \\ \min(\mathrm{SC}^j_{C_1,\mathrm{Con}_1^i}, \mathrm{SC}^q_{T_i(C_2),\mathrm{Con}_2^i}) \end{bmatrix}$$

Figure 3.3: Given the many-to-many matching for a particular transformation T_i, we need to establish correspondence hypotheses for each point P_1^j in a matched contour in image 1 to a point in image 2. A shape context is computed at point P_1^j using the matched contours in image 1 to characterize the shape around P_1^j. Similarly, shape contexts at possible corresponding points $q_k \in T_i(P_2)$ in the transformed contours of the second image are computed. These shape contexts are compared using the histogram difference and intersection features via CorrScore, and the best match is kept as the correspondence for point P_1^j under transformation T_i.

3.3.4 Baseline Comparison

As a baseline comparison, we attempted grouping using an MN that involved no selection information. The binary potential remained the same, while the unary potential was a function of the distance of each contour point in contour C_1^j to its closest match in P_2, under the transformation T_i:

$$\text{ShapeMatchScore}(l_j == i) \propto \sum_{l=1}^{n} [\min_{q \in T_i(P_2)} (||p_{k_l} - q||_{L2}^2, \text{occlusionThresh}^2)] \quad (3.5)$$

The constant occlusionThresh serves a threshold in case a contour point had no nearby match in P_2 under the transformation T_i. Points which had no match within occlusionThresh distance were marked as occluded for the hypothesis $l_j = a$. If more than half the points in the final assignment l_j^* for a contour were occluded, we marked the entire contour as occluded, and it was not assigned to any group (equivalently, it was assigned label $l_j == 0$). Since we omitted all selection information, all contours in the 1st image were included in the MN as nodes, and their contour points were allowed to match to any contour point in P_2. We again optimized the MN energy with the $\alpha - \beta$ swap graph cut. Free parameters were tuned by hand to produce the best result possible.

3.4 Experiments

We tested our method and the baseline over stereo, motion and similar image pairs. Input contours in each image were extracted automatically using the method of [72]. SIFT matches were extracted between each image, keeping only confident matches as described in [43]; matches proposing similar transformations were pruned to a small set, typically 10-20. To capture large scale shape, we used very large shape contexts (radius 90 pixels, in images typically of size 400 by 500), which made matching very robust. The shape contexts were augmented with edge orientation bins in addition to the standard radial and angular bins. Shape contexts were placed on a uniform

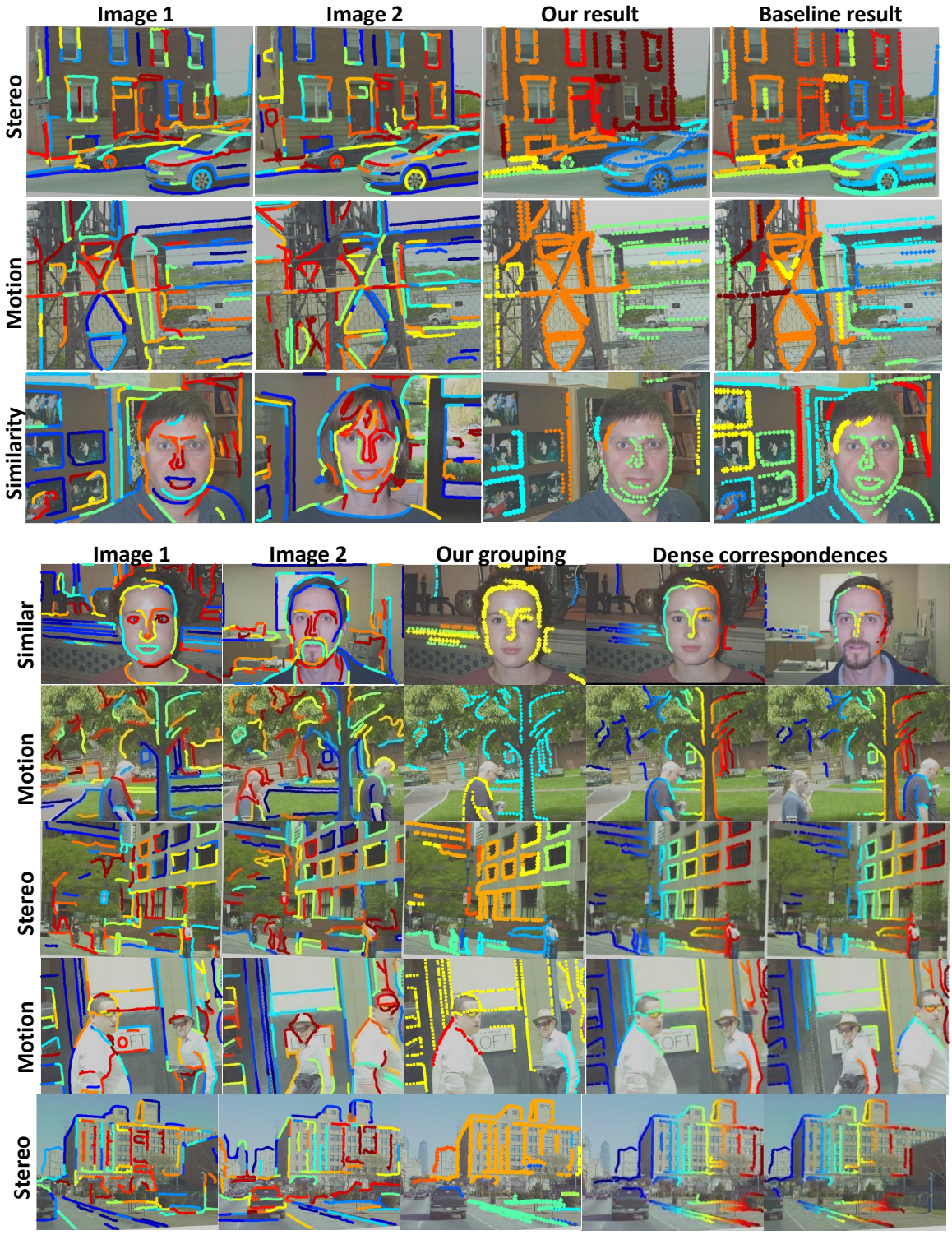


Figure 3.4: Baseline comparison (top) and additional results (bottom)

grid with a spacing 50 pixels in the x and y dimensions. Image pairs were taken from the Caltech 101 dataset [21] and from a stereo rig with 1m baseline mounted on a car from our lab (providing stereo as well as motion pairs from the same camera over time). The running time of our unoptimized MATLAB implementation was several minutes for each image pair.

Figure 3.4, top block, shows the results of our method and the baseline method on stereo, motion and similar images. We can see that our method provides superior groupings that better respect object boundaries. Groups for stereo image pairs are colored according to disparity. Due to the lack of large context, the baseline method is able to find a good match for a given contour point under almost any group hypothesis $l_j = a$, since in cluttered regions, there are always nearby matches. However, by being able to reason over a much larger, optimized context, our method exploits large-scale shape information and is better able to infer about occlusion, as well as layer assignment. We present additional results on different images in Figure 3.4, bottom block, and also show the dense correspondences. Interesting groups found in our results include facades of buildings, people, and a car (top row).

3.5 Observations

Using only longer contours for the matching is critical to the success of the method. As contours become shorter and shorter, hallucinating shapes out of the contours becomes easier, resulting in false matches. In addition, extracting as many possible transformations as possible out of the SIFT features is also important; if there is a correct transformation, the matching will confirm it with a good score, while incorrect transformations are easily pruned. In retrospect, the choice of shape context histograms for measuring the similarity of shape may not have been as good a choice as using a grid histogram, which would allow for matching under slightly less accurate alignments than shape contexts, whose smaller bins require very precise

alignment to properly match. Also, moving from the min-cut graph cut to clustering based on normalized cuts could also be useful, as it is relatively easy to extend the segmentation from pairs of frames to segmentation across several frames at once. For two edge points in the same image, an affinity can be established between the two using the similarity of motion profiles for each point. The motion profile, used by Shi and Malik ([59]) for motion segmentation in video, encodes a distribution over the possible motions for a specific image edge. An affinity between two image edges can be obtained by considering the similarity of their motion profiles, and normalized cuts can be used to partition a graph on these image edges using the affinities. In this setting, the motion profile for two edges in the same image can be computed using the shape matching score for each edge across the several different aligning transformations for the image with respect to another image. The motion profiles can then be used to construct affinities for partitioning via normalized cuts, allowing for partitioning of edges across multiple frames of video simultaneously, without the need for an explicit modeling of segment motion, which min-cut based techniques typically require.

Chapter 4

Holisitic Human Pose Estimation

Estimating the pose of a human from a single image is a key problem in both image and video understanding; pose is an important clue for understanding activity and intent of an individual. Naturally, this problem exhibits both key challenges of background clutter and pose variation. Pose variation for many human activities such as sports or dancing can be highly exaggerated, greatly complicating the recognition problem. There has been good previous work on this topic, but significant challenges remain ahead. We divide the previous literature on this topic into three main areas:

Top-down approaches: Felzenszwalb and Huttenlocher ([24]) developed the well-known pictorial structures method and applied it to human pose estimation. In the original formulation, pictorial structures does probablistic inference in a tree-structured graphical model, where the overall cost function for a pose decomposes across the edges and nodes of the tree, usually with the torso as the root. Pictorial structures recovers locations, scales and orientations of rigid rectangular part templates that represent a body. Pairwise potentials were limited to simple geometric relations (relative position and angle), while unary potentials were based on image gradients or edge detections. The tree structure is a limitation since many cues (e.g., symmetry of appearance of right and left legs) cannot be encoded. Ramanan ([50]) extended the original model to encode the fact that symmetric limb pairs have

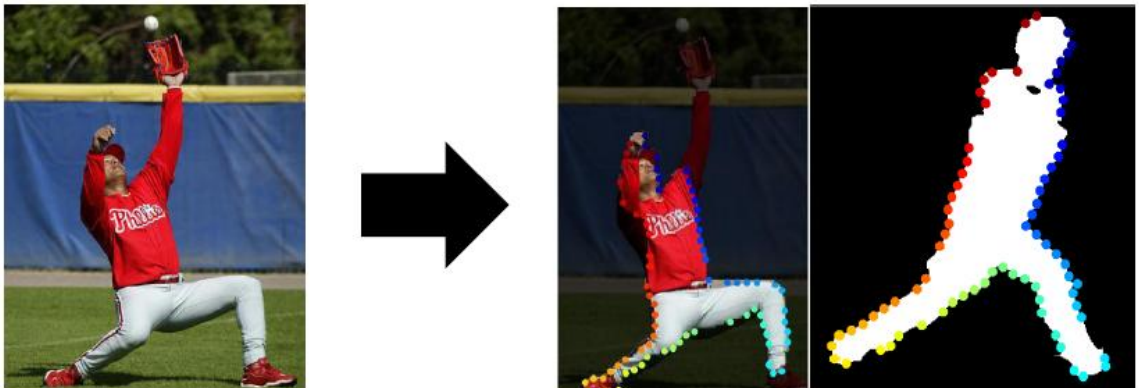


Figure 4.1: Given a single image of a human (left), our goal is to segment them from the background and estimate their pose via correspondence to an exemplar shape (right).

similar color, and that parts have consistent color or colors in general, but how to incorporate more general cues seems unclear. A top-down pictorial structures model was used in [51] to track people by repeatedly detecting them in subsequent video frames. Sigal et al. ([61]) introduced a non-parametric belief propagation method with occlusion reasoning to determine the pose. Andriluka et al. ([3]) used a pictorial structures model to simultaeously detect people in images while estimating their pose. All these approaches estimate pose, and do not provide an underlying segmentation of the image. Their ability to utilize more sophisticated cues beyond pixel-level cues and geometric constraints between parts is limited.

Search approaches: Mori et al. [47] utilized heuristic-guided search, starting from limbs detected as segments from Normalized Cuts (NCut) ([16]), and extending the limbs into a full-body pose and segmentation estimate. A follow up to this by Mori, [46], introduced an Markov-Chain Monte Carlo (MCMC) method for recovering pose and segmentation. Lee and Cohen [37] developed an MCMC technique for inferring 3-D body pose from 2-D images, but used skin and face detection as extra cues. Zhang et al. [71] utilized a combination of top-down, MCMC and local search to infer 2-D pose.

Bottom-up/Top-down approaches: Ren et al. ([53]) used bottom-up detection of parallel lines in the image as part hypotheses, and then combined these hypotheses into a full-body configuration via an integer quadratic program. Zhang et al. ([71]) also fit into this category, as they use bottom-up cues such as skin pixel detection. Similarly, Hua et al. ([35]) integrated bottom-up skin color cues with a top-down, non-parametric belief propagation process. Mori ([46]) used superpixels to guide their search. While Borenstein et al. ([11]) estimate only segmentation and not pose for horses and humans in upright, running poses, they best utilize shape and segmentation information in their framework. Ronfard et al. ([54]) use bottom-up part detectors to detect part hypotheses, and then piece these hypotheses together using a simple dynamic programming procedure in much the same way as in [24].

4.1 Overview of Our Parsing Method

Our goal is to combine a subset of salient shapes S (in our case, represented as binary masks, and provided by segmenting the image via NCut) detected in an image into a shape that is similar to that of a human body. Because the body has a very distinctive shape, we expect that it is very unlikely for this to occur by chance alone, and therefore should correspond to the actual human in the scene.

We formulate this as a parsing problem, where we provide a set of parsing rules that lead to a parse (also represented by a binary mask) for the body, as see in Figures 4.2 and 4.3. A subset of the initial shapes S are then parsed into a body. The rules are unary or binary, and hence a non-terminal can create a parse by *composing* the parses of one or two children nodes (via the pixel-wise OR operator). In addition the parses for a node can be formed directly from a shape from S, in addition to being formed from a child/children. Traditional parsing methods (dynamic programming methods) that exploit a subtree independence (SI) property in their scoring of a parse can search over an exponential number of parses in polynomial time.

We can define a traditional context-free grammar as a tuple

$$\langle V, T, A, R, S \rangle \tag{4.1}$$

V are parse non-terminals and T are the terminals, where A is the root non-terminal,

$$R = \{A_i \rightarrow B_i, C_i\} \tag{4.2}$$

is a set of production rules with $A_i \in V$ and $B_i, C_i \in V \cup T$ (we restrict ourselves to binary rules, and unary rules by making C_i degenerate), and S_i is a score for using rule R_i. Further, for each image, a terminal $T_i \in T$ will have potentially multiple instantiations $t_i^j, j = 1, ..., n_i$ each with its own score u_i^j for using $T_i \rightarrow t_i^j$ in a parse. Each terminal instantiation $t_i^j \in S$, corresponds to an initial shape S drawn from NCut segmentation. If the root is $A \in V$, then we can compute the score of the best parse (and therefore the best parse itself) recursively as

$$P(A) = \max_{r_i | r_i = (A \rightarrow B_i, C_i)} (S_i + P(B_i) + P(C_i)) \tag{4.3}$$

However, this subtree independence property greatly restricts the type of parse scoring function that can be used.

By contrast, our approach seeks to maximize a shape scoring function F_A for A that takes as input two specific child parses b_i^j and c_i^k (or one, as we allow unary rules) corresponding to rule $A \rightarrow B_i, C_i$:

$$P(A) = \max_{r_i = (A \rightarrow B_i, C_i)} \max_{j,k} \ (F_A(b_i^j, c_i^k)) \tag{4.4}$$

Recall that we represent a parse b_i^j or t_i^j as a binary mask, not as the parse rules and terminals that form it. Note that the exact solution requires all parses for the children as opposed to just the best, since the scoring function F_A does not depend on the scores of the child parses. Because the exact solution is intractable, we instead solve this approximately by greedily pruning parses to a constant number. However, we use a richer parse scoring function that has no subtree independence property. We

can view the differences between the two methods along two dimensions: **proposal** and **evaluation**.

Proposal: Dynamic programming methods explore all possible parses, and therefore have a trivial proposal step. Our method recursively groups bottom-up body part parses into increasingly larger parts of the body until an entire body parse is formed. For example, a lower body could be formed by grouping two Legs, or a Thigh+Lower leg and a Lower leg, or taken directly from S. In the worst case, creating parses from two children with n parses each could create n^2 new parses. Therefore, pruning occurs at each node to ensure that the number of parses does not grow exponentially further up the tree. To prune, we eliminate redundant or low scoring parses. Because there is pruning, our method does not evaluate all possible parses. However, we are still able to produce high quality parses due to a superior evaluation function.

Evaluation: On the evaluation side, dynamic programming employs evaluation functions with special structure, limiting the types of evaluation functions that can be used. Usually, this takes the form of evaluating a parse according to the parse rule used (chosen from a very limited set of choices) and the scores of the subparses that compose it, as in Equation (4.3). However, this does not allow scoring of the parse in a holistic fashion. Figure 4.4 gives an example; two shapes that on their own are not clearly parts of a disk, but when combined together, very clearly form a disk. Therefore, we associate with each node i a scoring function F_i (as in Equation (4.4)) that scores parses not based on the scores of their constituent parses or the parse rule, but simply based on their shape. The scoring function also allows for pruning, as parses can be ranked and low-scoring parses can be discarded to control the number of parses. It is important to note that our choice of F_i does not exhibit an SI property. Because of this, we are primarily interested in the actual result of the parse, a binary mask, as opposed to how it was generated from child parses or from S. In contrast to dynamic programming methods, a parse is evaluated irrespective of how it was generated.

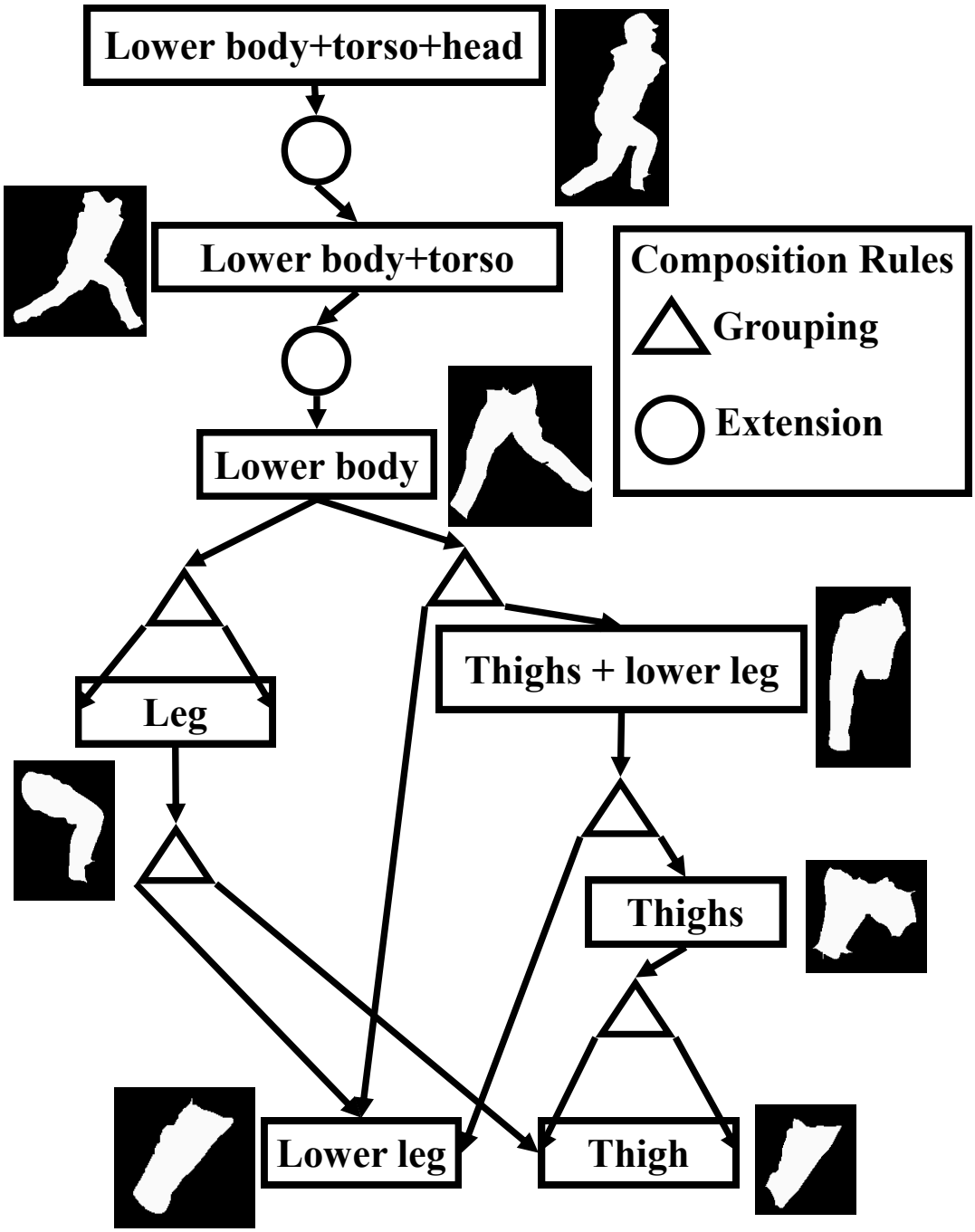


Figure 4.2: Our body parse tree, shown with an exemplar shape from our training set for each node; the exemplars are used for shape scoring. Shape parsing begins at the leaf nodes of thigh and lower leg and proceeds upwards. Note that in addition to composing parses from children nodes, parses can always come from the initial shapes S.

- {Lower leg, Thigh} → Leg
- {Thigh, Thigh} → Thighs
- {Thighs, Lower leg} → Thighs+Lower leg
- {Thighs+Lower leg, Lower leg} → Lower body
- {Leg, Leg} → Lower body
- {Lower body} → Lower body+torso
- {Lower body+torso} → Lower body+torso+head

Figure 4.3: Our parse rules. We write them in reverse format to emphasize the bottom-up nature of the parsing.

Figure 4.4: The two shapes on the left bear little resemblance to a disk in isolation. However, when combined, the disk is clear.

4.1.1 Multiple Segmentations

To initialize our bottom-up parsing, we need a set of intial shapes S. Mori et al. ([47]) noted that human limbs tend to be salient regions that NCut segmentation often isolate as a single segment. To make this initial shape generation method more robust, we consider not one segmentation as in [47], but 12 different segmentations provided by NCut. We vary the number of segments from 5 to 60 in steps of 5, giving a total of 390 initial shapes per image. This allows us to segment out large parts of the body that are themselves salient, e.g. the lower body may appear as a single segment, as well as smaller parts like individual limbs or the head. Figure 4.5 shows for an image 2 of the 12 segmenations with overlaid boundaries. Segments from different segmentations can overlap, or be contained within another. In our system, these segments are all treated equally. These initial shapes could be generated by other methods besides segmentation, but we found segmentation to be very effective.

4.1.2 Shape Comparison

For each node i, we have an associated shape scoring function F_i. For the root node, this ranks the final parses for us. For all other nodes, F_i ranks parses so that they can be pruned. All the shape scoring functions operate the same way: we match the boundary contour of the mask that represents a parse against boundary contours from a set of exemplar shapes using the inner-distance shape context (IDSC) of [41].

4.1.3 Parse Rule Application Procedure

Our parsing process consists of five basic steps that can be used to generate the parses for each node. For a particular node A, given all the parses for all children nodes, we perform the following steps:

Figure 4.5: Two segmentations of an image, 10 and 40 segments. Red lines indicate segment boundaries for 10 segments, green lines indicate boundaries for 40 segments, and yellow indicates boundaries common to both segmentations (best viewed in color).

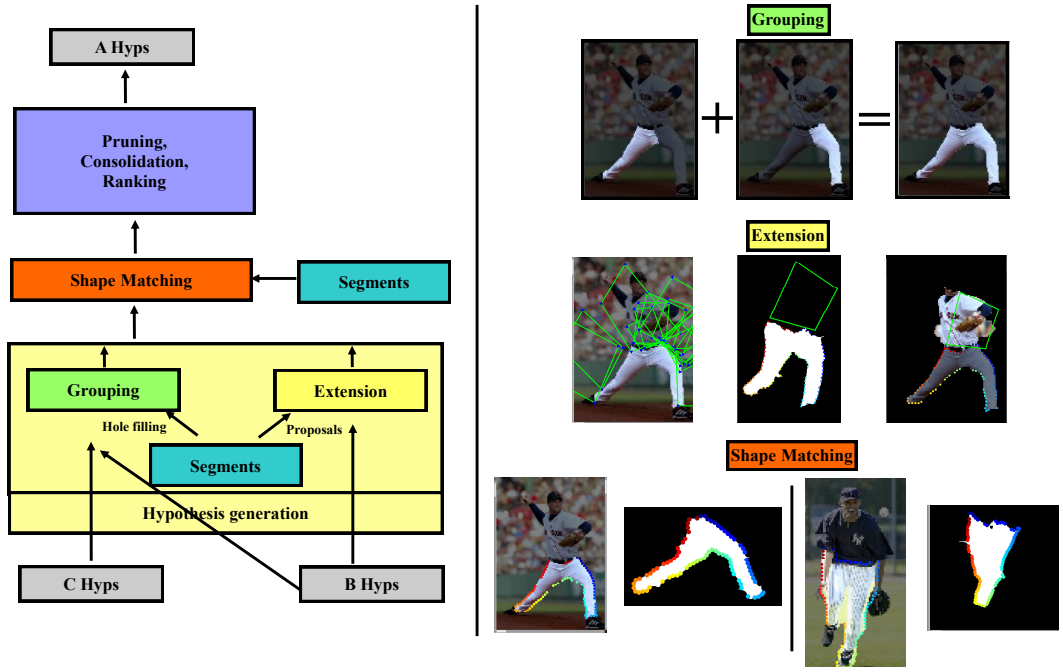


Figure 4.6: **Left:** parse rule application procedure. For binary rules, all pairs of child parses that are within 10 pixels of each other are composed via grouping, with hole filling provided by segments if needed. For unary rules, the child parses undergo extension using projected quadrilaterals and segment proposals. Shape matching is performed on both the original segments as well as the composed parses. For leaf nodes, shape matching is performed only on the segments. After shape matching, the parses are consolidated, pruned and ranked. **Right:** Grouping: two legs, on the left, are grouped into a lower body parse, on the right. Extension: the leftmost image shows a lower body parse with multiple different torso quadrilaterals projected from exemplars on to the image using the correspondence between the lower body parse and the lower body exemplars; the center image shows the exemplar with its torso quadrilateral that yielded the best torso parse, seen in the right image. Shape matching: two examples of shape matching. The lower body on the right was detected directly from the segments S, underscoring the importance of injecting the shapes from S into all levels of the parse tree.

Algorithm 1: $P_A = \text{Parse}(A, S)$: for a particular image, given initial segments S and part name A, produce ranked and pruned parses for A.

Input: Part name A and initial shapes S
Output: P_A: set of ranked and pruned parses for A
$P_A = S$; `// Include all of` S `as parse candidates`
foreach *rule* $\{B_i, C_i\} \rightarrow A$ *(or* $B_i \rightarrow A$*)* **do**
    $P_{B_i} = \text{Parse}(B_i, S)$; `// Recurse`
    $P_{C_i} = \text{Parse}(C_i, S)$; `// If binary rule, recurse`
    $P_A = P_A \cup \text{Group}(P_{B_i}, P_{C_i})$ (or $\text{Extend}(P_{B_i})$); `// Add to parses of A`
$P_A = \text{RankByShapeMatchingScore}(P_A)$;
$P_A = \text{Prune}(P_A)$; `// Prune redundant/low scoring parses`
return P_A; `// Return parses`

Parse rules

Segment inclusion: applies to all nodes We include by default all the masks in S as parses for A. This allows us to cope with an input image that is itself a silhouette, which would not necessarily be broken into different limbs, for example. A leg will often appear as a single segment, not as separate segments for the thigh and lower leg; it is easier to detect this as a single segment, rather than trying to split segments into two or more pieces, and then recognize them separately. For nodes in the parse tree with no children, this is their only source of masks.

Grouping: $\{B, C\} \rightarrow A$ For binary rules, we can compose parses from two children such as grouping two legs into a lower body, e.g. {Leg, Leg} $\rightarrow$ Lower body. For each child, based on the alignment of the best matching exemplar to the child, we can predict which part of the segment boundary is likely to be adjacent to another part.

A pair of masks, b from B and c from C, are taken if the two masks are within 30 pixels of each other (approximately 1/10th of the image size in our images), and combined with the pixel-wise OR operator. Because we need a single connected shape for shape comparison, if the two masks are not directly adjacent we search for a mask from the segmentations that is adjacent to both, and choose the smallest

such mask m. m is then combined with b and c into a single mask with a single connected component. If no such mask exists, we just keep the larger of a and b. Figure 4.6 provides an example of the parse rule, {Leg,Leg} $\rightarrow$ LowerBody.

Extension: $\{B\} \rightarrow A$ For unary rules we generate parses by projecting an expected location for an additional part based on correspondence with exemplars. This is useful when bottom-up detection of a part by shape, such as the torso or head, is difficult due to wide variation of shape, or lack of distinctive shape. Once we have a large piece of the body (at least the lower body), it is more reliable to directly project a position for other parts. Given a parse of the lower body and its correspondence to a lower body exemplar shape, we can project the exemplar's quadrilateral representing the torso on to the parse (we estimate a transform with translation, rotation and scale based on the correspondence of two contour points closest to the two bottom vertices of the torso quadrilateral).

Similarly, given a mask for the lower body and torso, and its correspondence to exemplars, we can project quadrilaterals for the head. With these projected quadrilaterals, we look for all masks in S which have at least half their area contained within the quadrilateral, and combine these with the existing mask to give a new parse. For each parse/exemplar pair, we compose a new parse.

Complexity Control

Scoring Once parses have been composed, they are scored by matching to the nearest exemplar with IDSCs and dynamic programming. Correspondence is also established with the exemplar, providing an estimate of pose.

Pruning Many parses are either low-scoring or redundant or both. We prune away these parses with a simple greedy technique: we order the parses by their shape score, from highest to lowest (best to worst). We add the best parse to a representative set, and eliminate all other parse which are similar to the just added parse. We then recurse on the remaining parses until the representative set reaches a fixed size. For

mask similarity we use a simple mask overlap score O between masks a and b:

$$O(a,b) = \frac{area(a \bigcap b)}{area(a \bigcup b)} \tag{4.5}$$

where $\bigcap$ performs pixel-wise AND, and $area(m)$ is simply the count of pixels with value 1 in the mask. If $O(a,b)$ is greater than a particular threshold, a and b are considered to be similar. After this step, we have a pruned set of parses that can be passed higher in the tree, or to evaluate in the end if the node A is the root. Figure 4.6 illustrates the stages of the parsing process for generating the parse for a single node. Also included are examples of grouping, extension, and shape matching/scoring.

Algorithm 1 sums up the parsing process for a particular part A, given intial set of shapes S from segmentation. It recursively generates parses for the children parts, and therefore to parse the torso+lower body+head (TLBH), we would call $Parse(TLBH, S)$. Note that if the part is a child in the parse tree, then no recursion occurs, and only the shapes S can form parses.

4.2 Results

We present results on the baseball dataset used in [47] and [46]. This dataset contains challenging variations in pose and appearance. We used 15 images to construct shape exemplars, and tested on $|I| = 39$ images. To generate the IDSC descriptors, we used the code provided by the authors of [41]. Boundary contours of masks were computed and resampled to have 100 evenly-spaced points. The IDSC histograms had 5 distance and 12 angle bins (in $[0, 2\pi]$). The occlusion penalty for dynamic programming matching of contours was 0.6 * (average match score), and 10 different alignments were used to initialize contour registration. For pruning, we used a threshold of 0.95 for the overlap score to decide if two masks were similar (a, b are similar $\iff O(a,b) \geq 0.95$) for the lower body+torso and lower body + torso + head, and 0.75 for all other pruning. In all cases, we pruned to 50 parses.

For parsing via grouping of parses from two different nodes, we can compose at most $50^2 = 2500$ parses. In practice, we typically found this to be between 500 and 1500 parses. For parsing via extension, for each of the 50 child parses, we create 15 new parses, 1 per exemplar, for a total of 750 parses. For each node, we examine an additional 390 parses from S. Given that there are 8 nodes, 2 extension relationships, and 5 grouping relationships, this gives an upper bound # of $2500 * 5 + 750 * 2 + 390 * 8 = 17120$ parses. With 15 exemplars, the number of shape comparisons is at most $15 * 17120 = 256800$.

Because we limit ourselves to shape cues, the best mask (in terms of segmentation and pose estimate) found by the parsing process is not always ranked first; although shape is a very strong cue, it alone is not quite enough to always yield a good parse. We expect that incorporating other cues would allow us to rank the best parse at, or very close to, the top. Our main purpose was to investigate the use of global shape features over large portions of the body via shape parsing. We evaluate our results in two different ways: segmentation score and projected joint position error. To the best of our knowledge, we are the first to present both segmentation and pose estimation results on this task.

4.2.1 Segmentation Scoring

We present our results in terms of an overlap score for a mask with a ground truth labeling. Our parsing procedure results in 50 final masks per image, ranked by their shape score. We compute the overlap score $O(m, g)$ between each mask m and ground truth mask g. We then compute the cumulative maximum overlap score through the 50 masks. For an image i with ranked parses $p_1^i, ...p_n^i$, we compute overlap scores $o_1^i, ..., o_n^i$. From these scores, we compute the cumulative maximum $C^i(k) = \max(o_1^i, ..., o_k^i)$. The cumulative maximum gives us the best mask score we can hope to get by taking the top k parses.

To understand the behavior of the cumulative maximum over the entire dataset, we

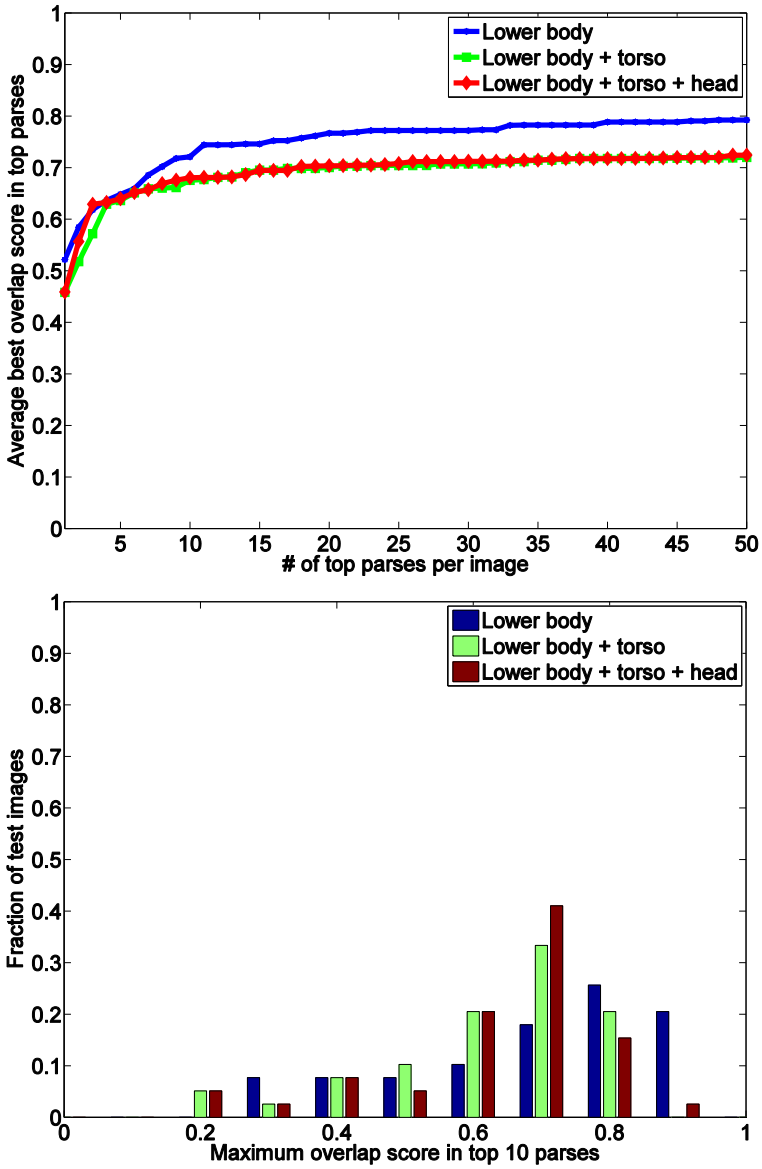


Figure 4.7: **Top:** We plot the average of each image's maximum overlap score as a function of the number of final parses retained, and do this for each region. **Bottom:** To give greater insight into the distribution of overlap scores, we focus on the top 10 parses, and histogram the best overlap score out of the top 10 for each image and region.

compute $M(k) = \frac{1}{|I|}\sum_{i=1}^{|I|} C^i(k)$, or the average of the cumulative maximum over all the test images for each $k = 1, ..., n$ ($n = 50$ in our case). This is the average of the best overlap score we could expect out of the top k parses for each image. We consider this a measure of both precision and recall; if our parsing procedure is good, it will have high scoring masks (recall) when k is small (precision). On top in Figure 4.7, we plot $M(k)$ against k for three different types of masks composed during our parsing process: lower body, lower body+torso, and lower body + head + torso. We can see that in the top 10 masks, we can expect to find a mask that is similar to the ground truth mask desired, with similarity 0.7 on average. This indicates that our parsing process does a good job of both generating parses as well as ranking them.

While the above plot is informative, we can obtain greater insight into the overlap scores by examining all $C^i(k)$, $i = 1, ..., |I|$ for a fixed $k = 10$. We histogram the values of $C^i(10)$ on the bottom in Figure 4.7. We can see that most of the values are in fact well over 0.5, clustered mostly around 0.7. This confirms our belief that the parsing process is effective in both recalling and ranking parses, and that shape is a useful cue for segmenting human shape.

4.2.2 Joint Position Scoring

We also examinine the error in joint positions predicted by the correspondence of a parse to the nearest exemplar. We take 5 joints: head-torso, torso-left thigh, torso-right thigh, left thigh-left lower leg, right thigh-right lower leg. The positions of these joints are marked in the exemplars, and are mapped to a body parse based on the correspondence between the two shapes. For a joint with position j in the exemplar, we locate the two closest boundary contour points p, q in the exemplar that have corresponding points p', q' in the shape mask. We compute a rotation, scaling and translation that transforms p, q to p', q', and apply these to j to obtain a joint estimate j' for the parse mask. We compare j' with the ground truth joint position

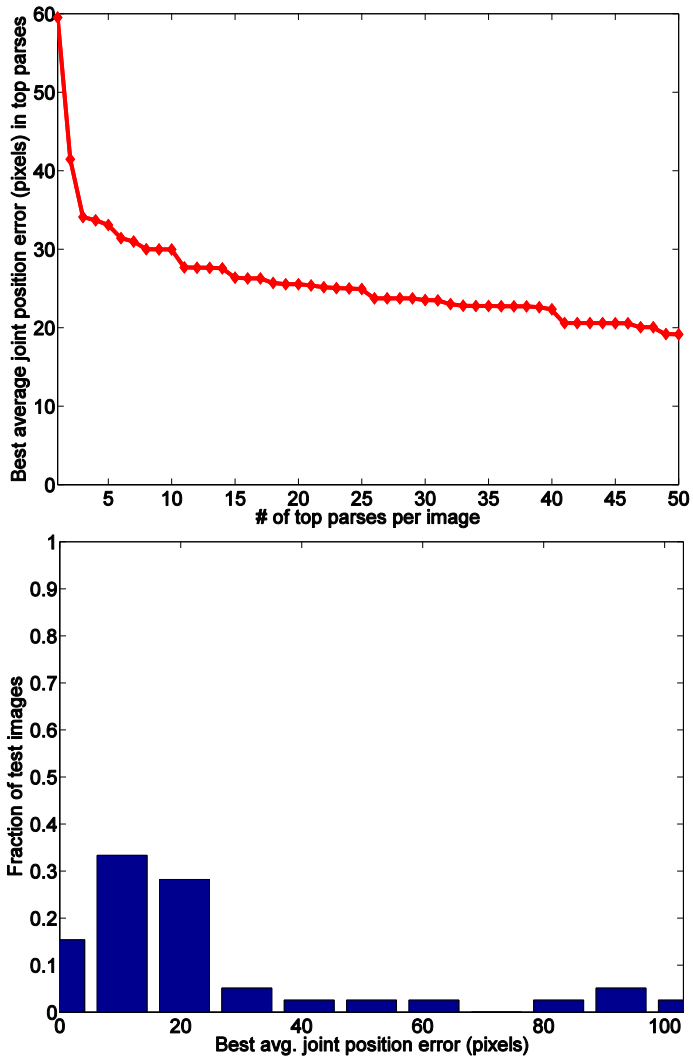


Figure 4.8: **Top:** We plot the average, across all images, of the minimum average joint error in the top k parses as a function k, the number of parses retained. **Bottom:** Taking the top 10 parses per image, for each image we compute the minimum average joint error from these top 10. We then histogram these values to show that taking 10 parses is likely to lead to recall of a good body parse. We can see that the vast majority of average errors are roughly 20 pixels or less.

via Euclidean distance. For each mask, we compute the average error over the 5 joints. Given these scores, we can compute statistics in the same way as the overlap score for segmentation. On the top in Figure 4.8 we plot the average cumulative *minimum* $M(k)$, which gives the average best-case average joint error achieveable by keeping the top k masks. We see again that in the top 10 masks, there is a good chance of finding a mask with relatively low average joint error. On the bottom in Figure 4.8, we again histogram the data when $k = 10$.

We show several example segmentations/registrations of images in Figure 4.9. Note that with the exception of the arms, our results are comparable to those of [46] (some of the images are the same), and in some cases our segmentation is better. As noted in [46], although quantitative measures may seem poor (e.g., average joint position error), qualitatively the results seem good. Figure 4.10 shows the top five parse results (according to exemplar matching score) for each of the images in the test dataset, along with the correspondences of the body mask to the nearest exemplar. We can see that the inner-distance shape context is effective at matching very different poses.

4.3 Observations

Clearly, the quality of the segmentation plays an important role in the performance of the method, but as long as large segments are used, sampling many different segmentations does not seem to negatively impact the performance of the method. Some poses of course also have more distinctive shapes (e.g. legs apart versus legs together) that are easier to recognize. The size of the exemplar set need not be very large; most likely, the number of shape exemplars can be cut from the 15 used by default to a smaller number like 5, for example. Shape exemplars could be automatically clustered using similarities measured from inner-distance shape context matching between different exemplars. Internal features of the body have

been used before (as in Mori et al. [48]), but shape seems to be the best cue (and results in [63] corroborate this), especially given that large segments and the holistic nature of the inner-distance shape context convey rich shape information. Recent work has also focused on the use of image contours (Zhang et al. submitted to CVPR 2011) for human pose estimation, which often provide better recall and grouping of image boundaries, and may serve as a better set of bottom-up structures for pose estimation. Lastly, the develop of shape features specific to interesting body configurations could be especially useful for difficult aspects of human pose such as the upper body, where features such as junctions can indicate important occlusion relationships.

Figure 4.9: We present additional pose estimation results. The segmentation of the person has been highlighted and the contour drawn as colored dots, indicating correspondence to the best matching exemplar. All the parses were the top scoring parses for that image (images are ordered row-major), with the exception of images 4 (2nd best), 8 (3rd best), 6 (3rd best). Some images were cropped and scaled for display purposes only. Full body overlap scores for each image (images are ordered row-major): 0.83, 0.66, 0.72, 0.74, 0.76, 0.70, 0.44, 0.57 and 0.84. Average joint position errors for each image: 12.28, 28, 27.76, 10.20, 18.87, 17.59, 37.96, 18.15, and 27.79.

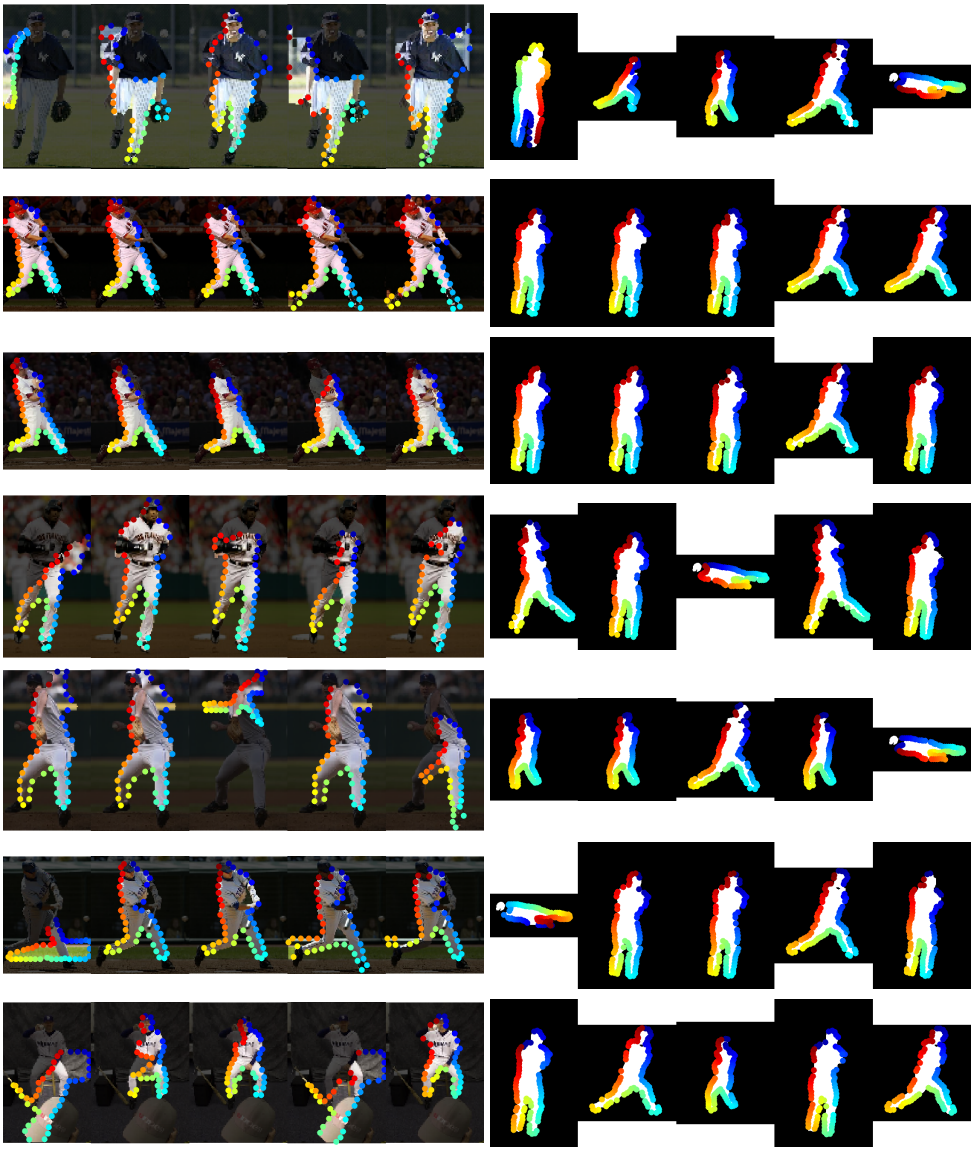


Figure 4.10: Top 5 parse results for each of the test images in the baseball images dataset. Each row contains, on the left, the top 5 parse results in terms of overall score (best shape matching score to a torso+lower body + head exemplar); on the right is the correspondence of the image to the best exemplar. The images are ordered by accuracy of the best match in the top 5, measured with respect to ground truth.

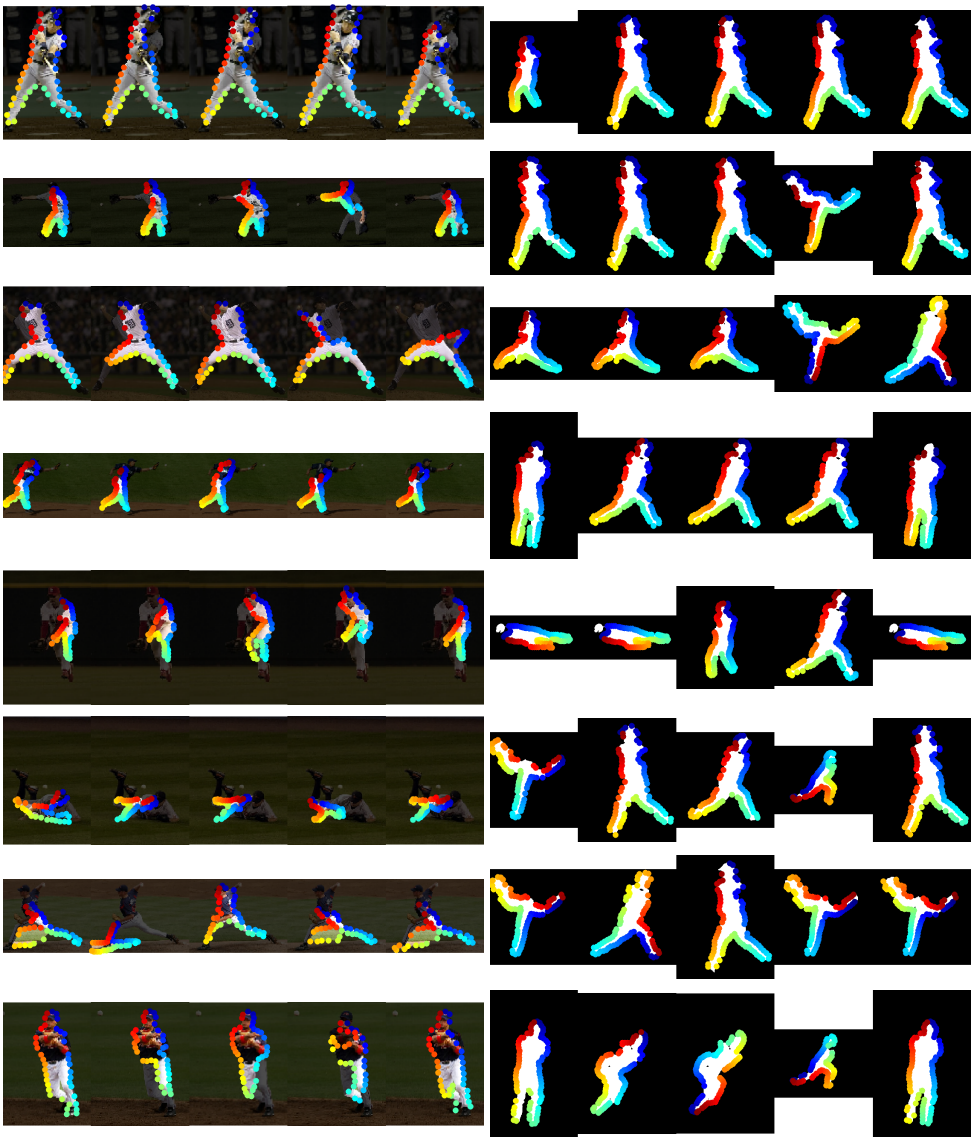


Figure 4.10: Top 5 parse results, continued.

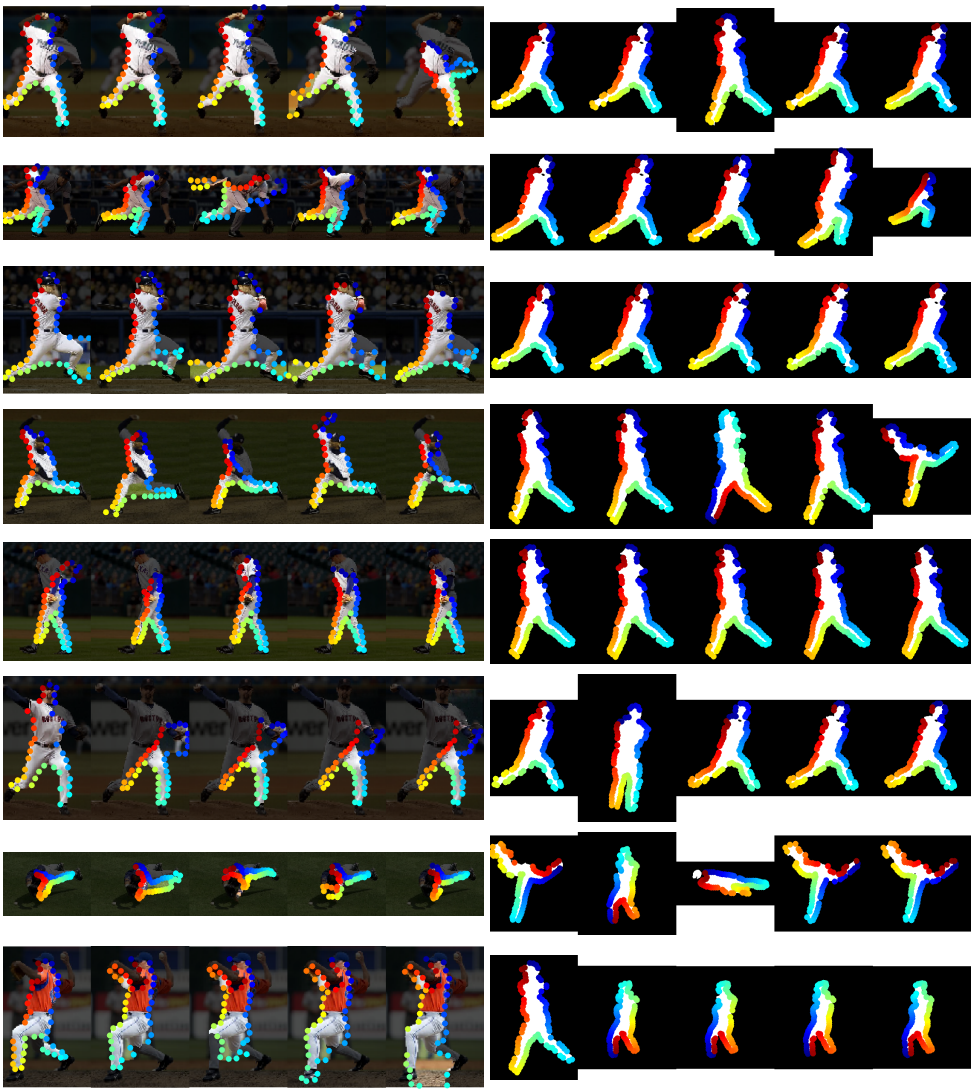


Figure 4.10: Top 5 parse results, continued.

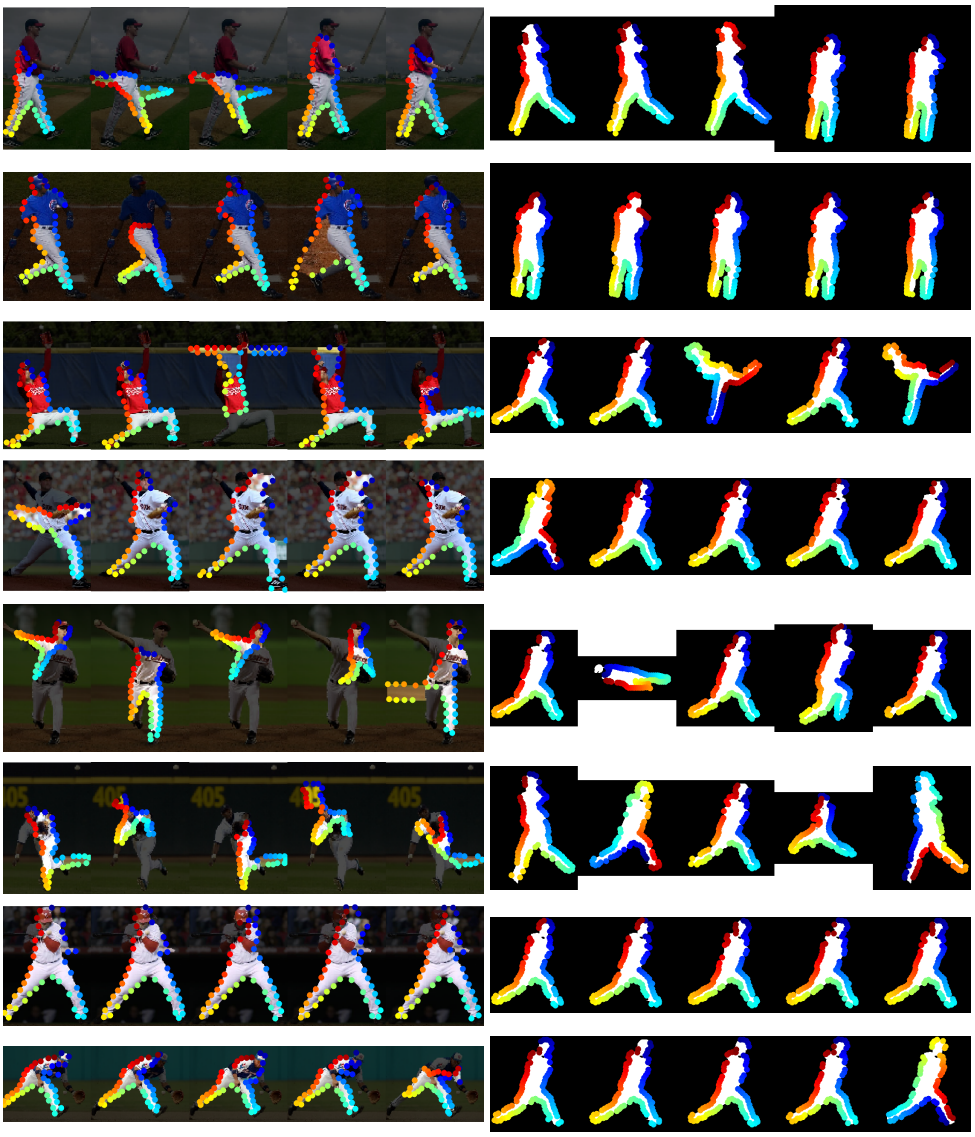


Figure 4.10: Top 5 parse results, continued.

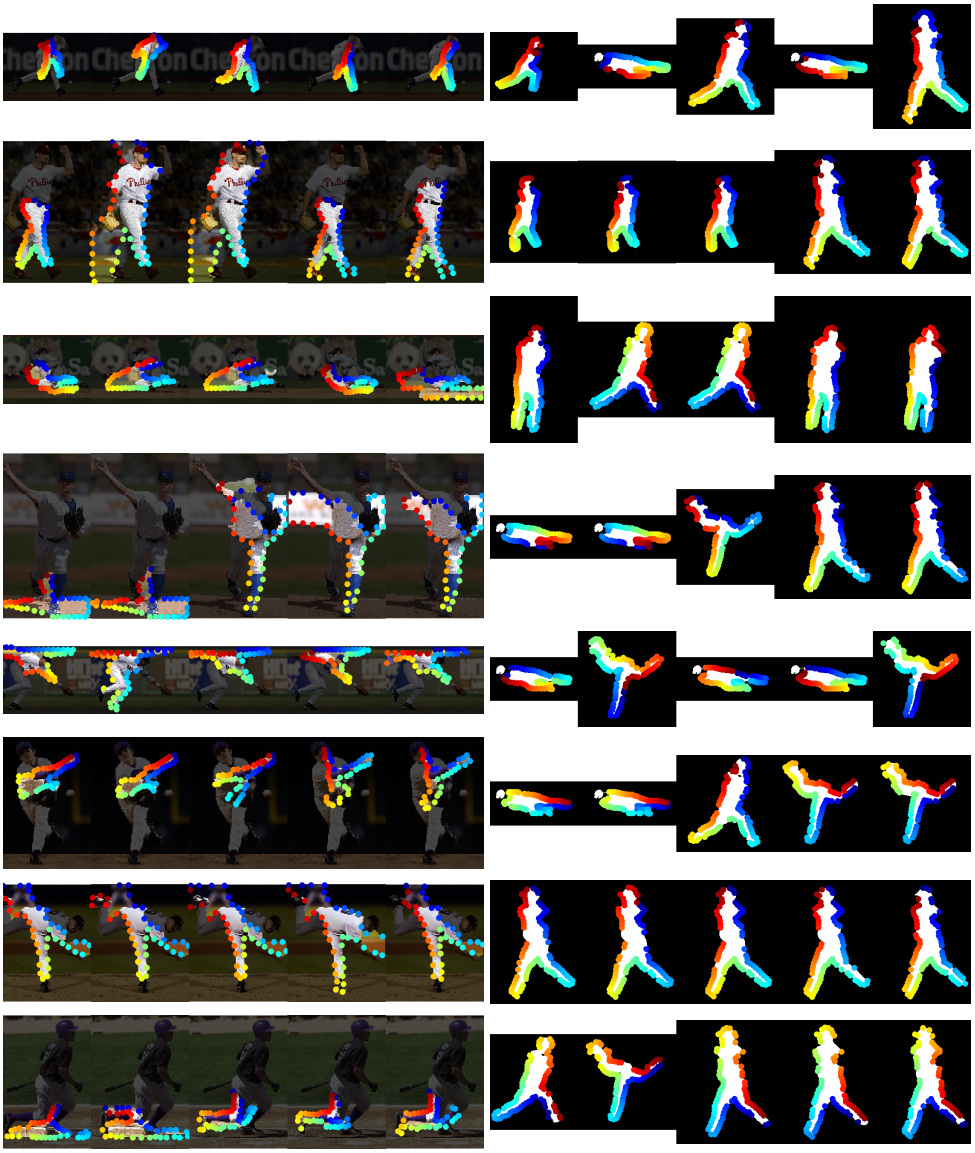


Figure 4.10: Top 5 parse results, continued.

Chapter 5

Many-to-one Matching For Describing and Discriminating Object Shape

While the previous chapter concerned the use of bottom-up image segmentation for human pose estimation, here we are concerned with recognizing more general non-articulated objects. Although the variation of pose of non-articulated objects is not as substantial, there is sufficient deformation nevertheless for the relative spatial relationships of both clutter and foreground shape features to vary substantially. Again, to address this problem we use a cost function that incorporates holisitic shape matching. One of the weaknesses of the parsing approach was the need for at least some regions of the body to appear clearly as individual segments. In practice, this assumption is not always satisfied, and so in this chapter we focus on recognition that is invariant to bottom-up fragmentation, using image contours. The advantage of image contours is that they tend to leak into the background less often, resulting in more accurate portions of object boundary shape, as well as being easily adapted for fragmentation-invariant recognition (following [73]).

Our approach is summarized in Fig. 5.1. Given positive and negative images for an

object category, and object bounding boxes, we use a bottom-up grouping method ([72]) to obtain long, salient image contours. From these contours, we first learn a descriptive shape model from the positive images. We then discriminatively tune the model using additional negative images for good detection performance. Both the learning steps use many-to-one matching of image contours to a shape model as a key process.

This chapter first provides an overview of related previous work on object recognition, covering contour-based detection, learning of object shape, and discriminative training of object detectors. Then the method for learning object shape is discussed, followed by the discriminative training of object detectors using the Latent-SVM learning framework ([22]), which allows for discriminatively training object detectors that have latent variables (such as the positions of object parts and their segmentation).

5.1 Overview

There are three different areas related to our approach: image contour-based object detection, object shape learning, and discriminative object detection.

Image contour-based detection methods: There are several works that use bottom-up image contours for object detection; [26] is most related. They used voting of configurations of bottom-up contours followed by a refinement scheme to discover shape from positive examples. In [44], the authors extracted contours and used a particle filter method for recognizing and grouping contours given a hand-drawn model. However, both methods expect image contours to have consistent fragmentation across images, which is not always satisfied. The method of [73] used many-to-one (and also many-to-many) matching of image contours to a shape model for a detection, but required a hand-drawn model and hand-tuned detection parameters.

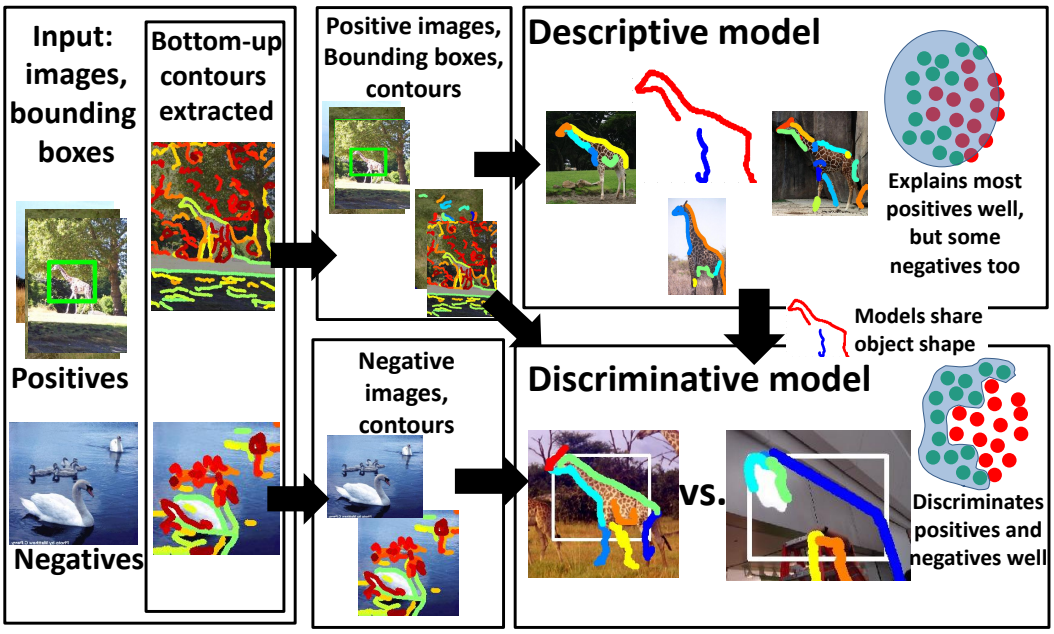


Figure 5.1: Overview of our proposed method. Positive and negative images labeled with bounding boxes for instances of a single object category are provided. Long, salient, bottom-up image contours are extracted from these images. The positive images are used to learn a descriptive shape model of the object shape by finding "lucky" image contours that can explain much of the shape of the positive examples via a many-to-one matching criterion. A discriminative detection model is trained using all the training images, their image contours and the learned shape model.

Object shape learning: In [20], the authors learned the outline shape of an object category from contours (found using a snake model), but were limited to learning from clutter-free "cartoon-images" and did not learn interior contours. There has been substantial other work in learning object shape, such as [38], [70], [39], and [60] but they do not leverage bottom-up image segmentations, leaving them at risk for accidental alignments in clutter. In addition to detection, [26] also learned object shape using a combination of discovery of recurring configurations and refinement steps; however this again expects that configurations of bottom-up contours recur consistently.

Discriminative object detection: Topic-model approaches, such as [28], have been used to discover a low-dimensional representation for image regions as a set

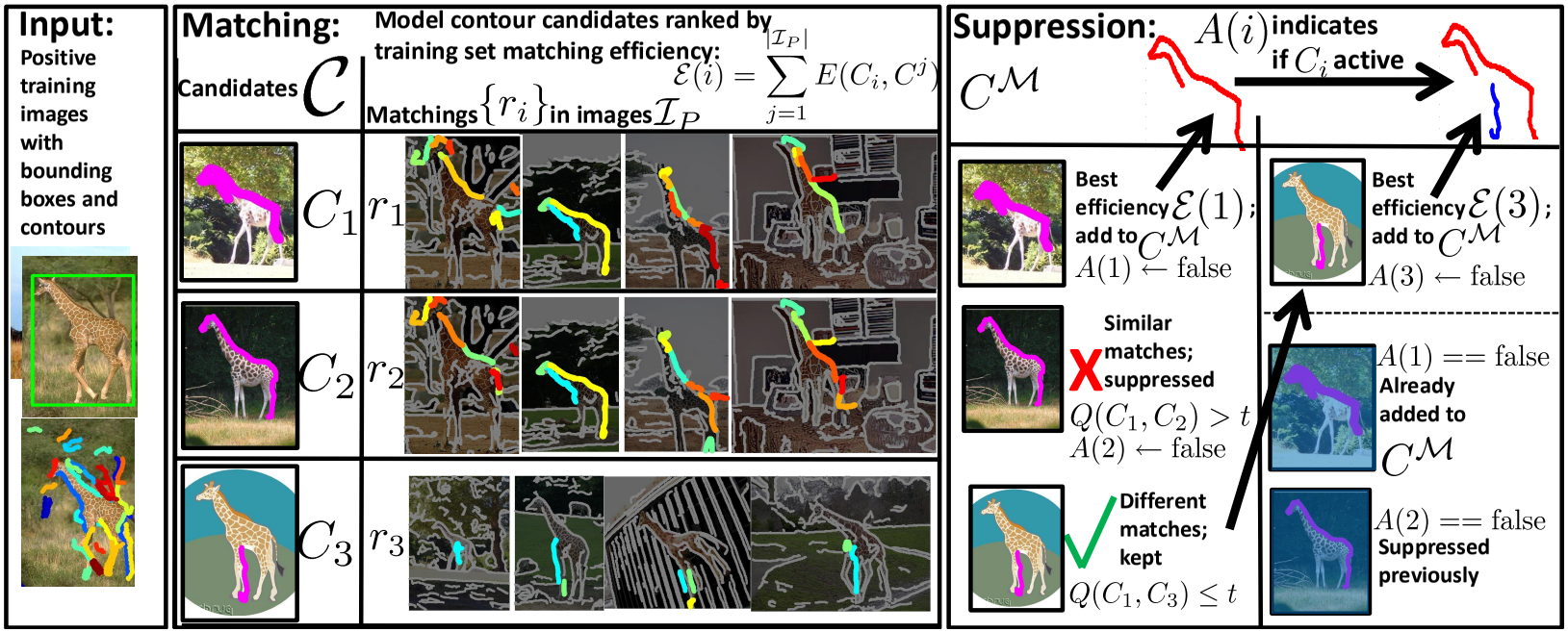


Figure 5.2: Model shape learning process, simplified example. Left, input consists of positive images w/ bounding boxes and contours (see Fig. 5.1). Middle: image contours are matched ($\{r_i\}$) against model candidates $\{C_i\}$ to compute training set matching efficiency $\mathcal{E}$. Right: model $C^{\mathcal{M}}$ constructed greedily using efficiency scores $\mathcal{E}$ (see Algorithm 2). Top candidate (according to $\mathcal{E}$) added to model and similar ones (Q) suppressed (A is active list; $A(i) \leftarrow$ false); process repeats. Contours shown in order of efficiency score for simplicity.

of responses to the topic model. This representation provided features for an SVM classifier. Local feature voting for hypothesis proposal was combined with intersection kernel SVM verification by [45]. However, these patch-based methods do not leverage the non-accidental nature of bottom-up segmentation. Boosting has been used to learn an object model shape made of contours extracted from training images, as in [49]. However, they used a chamfer matching cost for scoring placements of object parts in the image which often matches clutter. Some work ([52]) used a hand-drawn model of the object of interest, broke the model contours into fragments, and searched the image for fragment matches. Matches voted to yield object detections, which were followed by refinement and verification steps. In contrast, our method learns an object shape model automatically.

The histogram of oriented gradients (HOG, [18]) image feature was used in a discriminatively trained deformable part object model where part placements were scored by correlation of a learned part filter with the HOG features ([22] used). However, the support of a HOG feature in the image may overlap between the object and background (also noted in [38]), which can lead to unpredictable changes in the features.

This chapter first discusses the model shape learning, also explaining the use of many-to-one matching, and then explains the discriminative learning.

5.2 Learning a Descriptive Shape Model

We assume as input a set of positive images $\mathcal{I}_P$ containing the object $\mathcal{I}_P = \{I_1, ..., I_{|\mathcal{I}_P|}\}$. Each image has associated with it one or more bounding boxes that indicate the rough location and size of the instances of the object in the image. Also, for each image I_j we use a bottom-up contour grouping process (we use the method of [72]) to obtain image contours:

$$C^j = \{C^j_1, ..., C^j_{|C^j|}\} \tag{5.1}$$

We restrict the contours to those that lie inside a bounding box. We define $\mathcal{C}$ as the

union of these sets of contours for all the images. Instead of learning the object shape from scratch ([38]), we use $\mathcal{C}$ as a set of natural candidates for object model. The key insight is that in some of the images a significant portion of the object is clearly visible, resulting in a long, salient object contour. By combining several of these "lucky" contours, we can obtain a good shape model. We write the model contours as $C^{\mathcal{M}} = \{C_1^{\mathcal{M}}, ..., C_{|C^{\mathcal{M}}|}^{\mathcal{M}}\}$, and have two criteria for selecting these contours:

- Matching efficiency: each model contour in $C^{\mathcal{M}}$ should be able to match efficiently (defined below) to object contours C^j in a training image I^j
- Non-overlapping: each model contour in $C^{\mathcal{M}}$ should be matched to mostly different contours $\mathcal{C} = \{C^1 \cup C^2 \cup ... \cup C^{|\mathcal{I}_P|}\}$ in the training images

We explain score functions for both separately and then integrate them into a single score.

Matching efficiency score: The matching efficiency score evaluates the ability of a single model contour $C_i^{\mathcal{M}}$ to match well to many fragmented contours C^j in an image I_j: $E(C_i^{\mathcal{M}}, C^j)$. The key difficulty is that there is no one-to-one correspondence between image contours and the model, making the problem well-suite for the many-to-one matching framework of Chapter 2. Recall that to perform many-to-one matching, we need a transformation that relates image contours to the model. In this case, the bounding boxes provided with the positive images allow us to infer rough alignment of the training images. We can derive a transformation $\mathbf{T}$ for each positive example from the center of the bounding box and its dimensions, translating the image contours and scaling them in x and y in order to align all training bounding boxes. For a particular set of matched image contours $\mathbf{x}^{\text{sel}}$, we again have the histogram comparison features $K(\mathbf{T}, \mathbf{x}^{\text{sel}})$.

We can now write the **matching efficiency** score E in terms of K, normalized by model contour length $l(C_i^M)$ (the length of a contour refers to its physical length, not simply the number of points in it):

$$E(C_i^{\mathcal{M}}, C^j) = \frac{1}{l(C_i^{\mathcal{M}})} \max_{\mathbf{x}^{\text{sel}} \in \{0,1\}^{|\mathrm{C^j}|}} w^{\text{app}\mathsf{T}} K(\mathbf{T}, \mathbf{x}^{\text{sel}})$$

We summarize the outputs of the matching process: training set matching efficiency $\mathcal{E}$ for each candidate in $\mathcal{C}$ (for $C_i \in \mathcal{C}$, sum of efficiencies over all training images; $\mathcal{E}(i) = \sum_{j=1}^{|\mathcal{I}_P|} E(C_i, C^j)$; see Fig. 5.2) and matching indicator vectors $\{r_1, ..., r_{|\mathcal{C}|}\}$, where each $r_i \in \{0,1\}^{|\mathcal{C}|}$ is an indicator vector. This vector tells us which contours in the images candidate $C_i \in \mathcal{C}$ was matched to.

In practice, we use the linear program to approximate $\mathcal{E}(i)$ for all candidates and prune those with low approximate score. We solve an integer program for the remaining candidates to get $\mathcal{E}(i)$ exactly, and discard those with $\mathcal{E}(i) < 0$ as they are unlikely to be good model contours.

Overlapping candidate suppression: Two different candidate contours in $\mathcal{C}$, from the same or different images, may both correspond to the same part of the object. Additionally, the contour grouping method we use ([72]) produces overlapping contours within a single image, unlike most bottom-up grouping methods. So, we must be careful not to include overlapping contours in our model. We define for any two candidates $C_i, C_j \in \mathcal{C}$, $Q(C_i, C_j)$ as a measure of overlap in terms of the which contours matched to the candidates:

$$Q(C_i, C_j) = \frac{\sum_{k=1}^{|\mathcal{C}|} r_i(k) r_j(k) l(C_k)}{\sum_{k=1}^{|\mathcal{C}|} r_j(k) l(C_k)} \tag{5.2}$$

The numerator computes the sum of lengths of contours that both C_i, C_j matched, while the denominator computes the sum of lengths of contours that C_j matched. If Q is large, then C_i matched most of the contours (according to total length) that C_j matched. Q lies in the interval $[0, 1]$. If C_j matched no contours, Q is defined to be 1 for all C_i.

Model shape score function: Given score functions for matching efficiency and overlap, we can now define a joint score function that optimizes the total matching

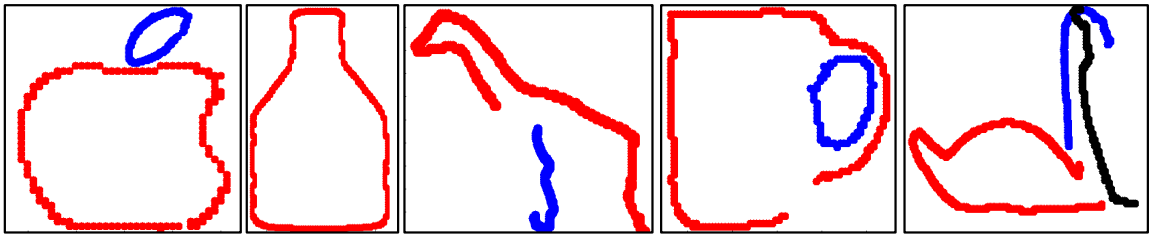


Figure 5.3: Models learned for Applelogos, Bottles, Giraffes, Mugs and Swans from the ETHZ Shape Classes dataset using half the images for each category. Contour colors indicate different candidates combined to form the model.

efficiency of the model contours subject to an overlap constraint $Q(C_i^{\mathcal{M}}, C_j^{\mathcal{M}}) \leq t$, where t is a pre-defined threshold on the contours:

$$\max_{C^{\mathcal{M}} | C^{\mathcal{M}} \subseteq \mathcal{C}} \sum_{i=1}^{|C^{\mathcal{M}}|} \sum_{j=1}^{|\mathcal{I}_P|} E(C_i^{\mathcal{M}}, C^j) \quad \text{s.t.} \quad \forall k, l \;\; Q(C_k^{\mathcal{M}}, C_l^{\mathcal{M}}) \leq t \tag{5.3}$$

Algorithm 2: Model shape learning: optimizing Eq. 5.3

Input *candidates* $\mathcal{C}$*, match eff. scores* $\mathcal{E}$*, matches* R;
Output *Model shape* $C^{\mathcal{M}}$*; local maximum for Eq. 5.3*;
Initialization $C^{\mathcal{M}} \leftarrow \emptyset$, $A \leftarrow \{1\}^{|\mathcal{C}|}$ *(indicates whether candidate still active)*;
while A *has a true entry* **do**
  $C_i \leftarrow$ active candid. w/ best match eff. score $\mathcal{E}(i)$;
  $C^{\mathcal{M}} \leftarrow C^{\mathcal{M}} \bigcup \{C_i\}$, $A(i) \leftarrow$ false ;
  $\forall j$ s.t. $Q(C_i, C_j) > t$, $A(j) \leftarrow$ false ;
end

Optimization over $C^{\mathcal{M}}$: Because the space of $C^{\mathcal{M}}$ includes all possible subsets of $\mathcal{C}$, there are an exponential number of states. We use a greedy approach which incrementally adds a contour candidate to the model according to training set matching efficiency and avoiding overlap, summarized in Algorithm 2. $C^{\mathcal{M}}$ is initialized to be empty ($\emptyset$), and all candidates are initially active ($A(i)$ is true). We add to $C^{\mathcal{M}}$ the active contour candidate with highest efficiency, and de-activate all conflicting candidates ($A(i) \leftarrow$ false, according to Q), repeating until no candidates remain. This process is guaranteed to improve the score in Eq. 5.3 at each step while obeying Q,

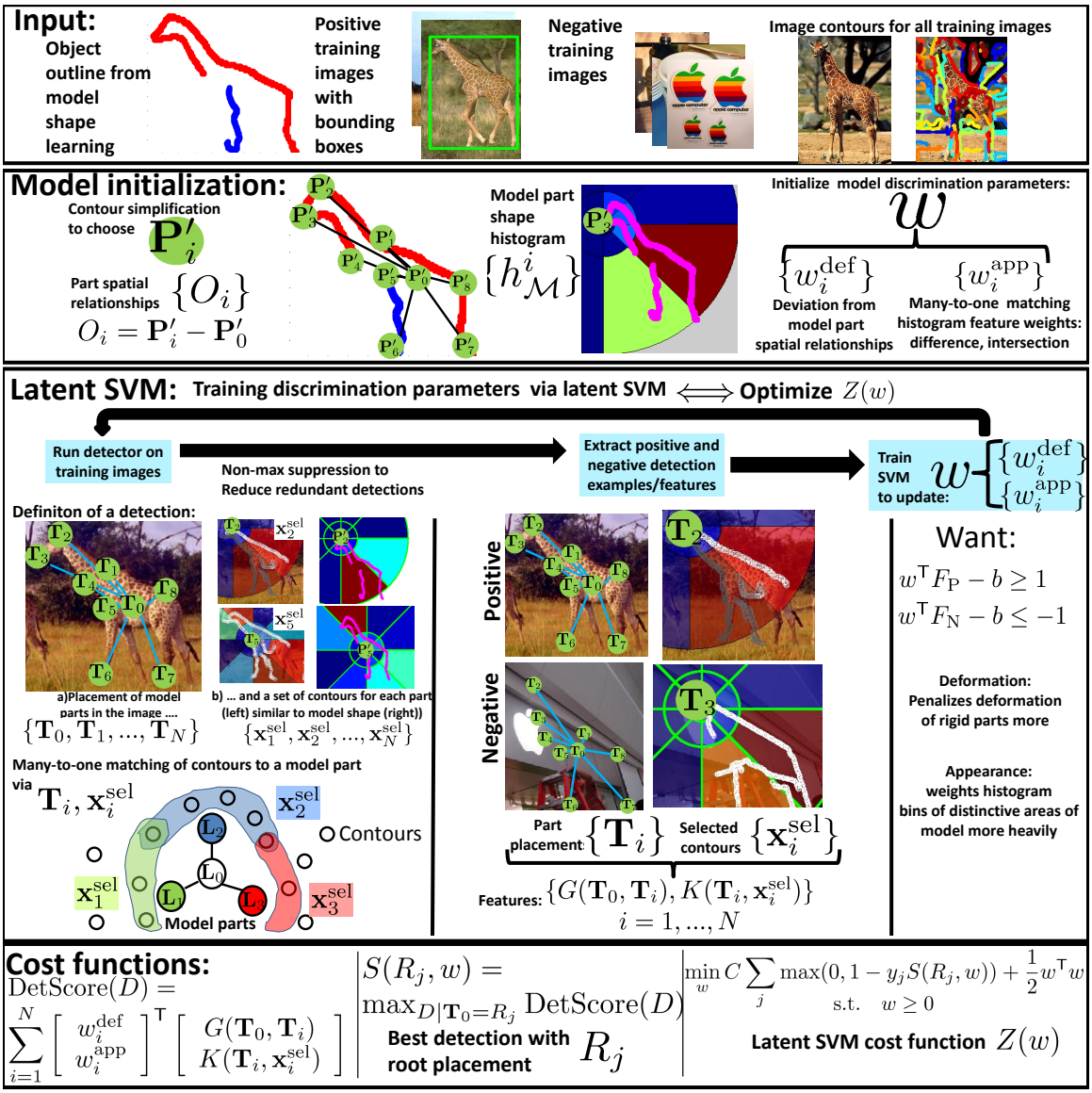


Figure 5.4: Overview of discriminative tuning of many-to-one matching score function."Input" panel: input to this step is the shape model automatically discovered previously, along with positive and negative training images and their image contours. "Model initialization": relative spatial relationships of parts w.r.t. to root part are computed along with shape histograms to represent part appearances. The many-to-one matching paramters (to be learned) are initialized. "Latent SVM": latent SVM training iterates between a) running the detector on images (left)/extracting positive, negative detection examples and features F_P and F_N, resp. (middle) and b) updating model parameters w, b (right). A detection (left) involves placements of the parts $\mathbf{T}_i$ and many-to-one matchings of contours to the parts $\mathbf{x}_i^{\text{sel}}$.

since we only keep candidates with $\mathcal{E}(i) > 0$. Fig. 5.2 depicts the learning process, and Fig. 5.3 shows results on the ETHZ Shape Classes dataset.

5.3 Discriminative Detector Learning

5.3.1 Object Detection as Many-to-one Matching

For object detection, we still perform many-to-one matching of image contours to a shape model (learned above), but with a matching score tuned for discrimination. Previously we assumed just one model part (single contour selection + part placement) because the bounding boxes provided the placement/warping. For detection, we extend to have $N + 1$ parts to better accomodate object deformation:

$$\text{Parts}: \ P_0, P_1, ..., P_N$$

Parts may deform relative to the root part, P_0, that represents the center of the object. In the model, parts $P_1, .., P_N$ are located at points of high curvature on the model shape contours: $P'_1, ..., P'_N$ (Fig. 5.4, panel "Model initialization"). We use the discrete curve evolution method of [36] (a contour simplification technique) to find these points P'_i from the model shape contours, and P'_0 on the model is computed as simply the mean of $P'_1, ..., P'_N$ in the model. Part appearances for parts P_i, $i = 1, ..., N$ are represented with model part histograms $h^i_{\mathcal{M}}$ centered at P'_i computed over the model shape (P_0 has no appearance term, although our formulation can accomodate one); we use shape context histograms (Fig. 5.4, "Model initialization").

For an image I_j with contours C^j, a detection of consists of a many-to-one matching for each part: transformations $\mathbf{T}_i$ for each part (which align placement of the part in the image back to the location of the part on the model), and selected contours for matching to each part $P_i, i = 1, ..., N$, $\mathbf{x}^{\text{sel}}_i$ (with the exception of the root part, which only serves to spatially relate the other parts):

$$\begin{aligned} \mathbf{T}_i &: P_i \to \mathbb{R}^2 \\ \mathbf{x}_i^{\text{sel}} &: C^j \to \{0,1\}^{|\mathrm{C^j}|} \end{aligned} \tag{5.4}$$

We define a detection as $D = \{\mathbf{T}_0, \mathbf{T}_1, \mathbf{x}_1^{\text{sel}}, \mathbf{T}_2, \mathbf{x}_2^{\text{sel}} ..., \mathbf{T}_N, \mathbf{x}_N^{\text{sel}}\}$. For simplicity, we abuse notation and refer to the correspondence of model point P_i' in the image using T_i. Following the terminology of [22], we also call T_i a *placement* of part P_i, since it refers to the location in the image where part P_i is hypothesized to lie. Fig. 5.4, "Latent SVM", left, also describes a detection.

Placement score: For each part P_i, we need to be able to score a placement $\mathbf{T}_i$ of the part in the image along with matching contours $\mathbf{x}_i^{\text{sel}}$. We use the same many-to-one shape matching features K, with a part-specific weight vector $w_i^{\text{app}} \geq 0$: $w_i^{\text{app}\mathsf{T}} K(\mathbf{T}_i, \mathbf{x}_i^{\text{sel}})$. The corresponding model part shape histogram is $h_{\mathcal{M}}^i$ (Fig. 5.4, "Model initialization", center).

Deformation score: For each part $i = 1, ..., N$ we use a part offset $O_i = (O_i^x, O_i^y)$ that describes the expected spatial position of P_i in the image, $\mathbf{T}_i$, relative to $\mathbf{T}_0$, the position of the root part in the image: $\mathbf{T}_0 + O_i$ (Fig. 5.4, "Model initialization"). O_i is computed as the difference between the locations of parts P_i and P_0 in the model: $P_i' - P_0'$. The deviation of a part P_i from its expected position relative to the root provides part deformation features G:

$$G(\mathbf{T}_0, \mathbf{T}_i) = \begin{bmatrix} -(\mathbf{T}_i^x - (\mathbf{T}_0^x + O_i^x))^2 \\ -(\mathbf{T}_i^y - (\mathbf{T}_0^y + O_i^y))^2 \end{bmatrix} \tag{5.5}$$

A set of parameters w_i^{def}, $i = 1, ..., N$ (to be learned, along with w_i^{app}) penalizes deviation of part P_i from its expected position relative to P_0. The overall score function for a particular detection D is:

$$\text{DetScore}(D) = \sum_{i=1}^{N} \begin{bmatrix} w_i^{\text{def}} \\ w_i^{\text{app}} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} G(\mathbf{T}_0, \mathbf{T}_i) \\ K(\mathbf{T}_i, \mathbf{x}_i^{\text{sel}}) \end{bmatrix} \tag{5.6}$$

In contrast to [22], our appearance term does not depend simply on the placement $\mathbf{T}_i$ of part P_i, but also on the contours chosen for matching, $\mathbf{x}_i^{\text{sel}}$.

Inference for detection: The space of possible detections is exponential in the number of possible placements for each part and number of image contours. To cope, we create a regular grid of possible root part locations $\mathcal{R}$ in the image and only keep the highest scoring detection per root part location $R_j \in \mathcal{R}$, as in [22]. For each possible root location R_j, we fix $\mathbf{T}_0 = R_j$, and then maximize the detection score subject to this root constraint to obtain score $S(R_j, w)$:

$$S(R_j, w) = \max_{D|\mathbf{T}_0=R_j} \text{DetScore}(D) \Longrightarrow$$
$$\max_{\{\mathbf{T}_1,...,\mathbf{T}_N,\mathbf{x}_1^{\text{sel}},...,\mathbf{x}_N^{\text{sel}}\}} \sum_{i=1}^{N} \begin{bmatrix} w_i^{\text{def}} \\ w_i^{\text{app}} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} G(R_j, \mathbf{T}_i) \\ K(\mathbf{T}_i, \mathbf{x}_i^{\text{sel}}) \end{bmatrix} \tag{5.7}$$

We note that $w_i^{\text{app}\mathsf{T}} K(\mathbf{T}_i, \mathbf{x}_i^{\text{sel}})$ does not depend on $\mathbf{T}_0$, and hence for each part P_i the maximization over $\mathbf{x}_i^{\text{sel}}$ can be pre-computed for each possible placement $\mathbf{T}_i$. In [22], this step corresponds to convolving the image with the filter associated with a part. In our case, we use the previously described linear programming relaxation to efficiently and accurately approximate the many-to-one matching score.

Given $w_i^{\text{app}\mathsf{T}} K(\mathbf{T}_i, \mathbf{x}_i^{\text{sel}})$ for each possible placement $\mathbf{T}_i$ of each part P_i,

$$\max_{D|\mathbf{T}_0=R_j} \text{DetScore}(D)$$

can be computed easily by picking the best part placement for each part P_i individually. For fixed $\mathbf{T}_0$, the scores for parts $P_1, ..., P_N$ are independent. The set of possible placements are sampled from points of high curvature along image contours, following the method of [36]. We take as a bounding box the bounding box of the part locations. A detection with center R_j can be labeled as a true or false positive (label $y_j = \pm 1$) according the overlap of its bounding box with a ground truth bounding box. Non-maximum suppression allows us to eliminate many redundant/overlapping detections, reducing the complexity of learning.

5.3.2 Latent SVM For Discriminative Detector Learning

Given detections centered at $R_j \in \mathcal{R}$ with labels $y_j = \pm 1$ from the training images, we learn discriminative model parameters $w = [w_1^{\text{def}\mathsf{T}} w_1^{\text{app}\mathsf{T}} \cdots w_N^{\text{def}\mathsf{T}} w_N^{\text{app}\mathsf{T}}]^{\mathsf{T}}$ to optimize detection performance.

Loss function: Following the standard SVM formulation, we can write a hinge-loss function associated with the above score function as:

$$\begin{aligned} Z(w) &= C \sum_j \max(0, 1 - y_j(S(R_j, w) - b)) + \frac{1}{2} w^{\mathsf{T}} w \\ \text{s.t.} \quad & w \geq 0 \end{aligned} \tag{5.8}$$

where $\max(0, 1 - y_j(S(R_j, w) - b))$ is the hinge loss and $\frac{1}{2} w^{\mathsf{T}} w$ is the regularization term. The learned constant b is the usual SVM bias term. We require $w \geq 0$ so that many-to-one matching score remains concave and maximizable.

Latent SVM training: We adapt the "coordinate descent" method from [22] for minimizing Eq. 5.8. Beginning with an initial set of parameters w_0, b_0, we run the detection procedure on the training images. The following two steps are iterated as shown in Fig. 5.4 ("Latent SVM"):

- *Update model parameters w, b given argmax detection for each root placement R_j:* Given the detection results from the previous model parameters, we train an optimal model w, b using the the features from the argmax of Eq. 5.7 (also see in Fig. 5.4) for each root R_j. This is equivalent to traditional SVM using these features, but with the added constraint $w \geq 0$.
- *For each root placement R_j, update part placements $\mathbf{T}_i$ and $\mathbf{x}_i^{\text{sel}}$ for parts $1, ..., N$:* Given new model parameters w, b, we recompute the optimal selections and placements $\mathbf{x}_i^{\text{sel}}, \mathbf{T}_i$, $i = 1, .., N$ for each root placement R_j by running the detection procedure.

We iterate the two update steps until the average precision of the detection precision-recall curve stops increasing, up to a maximum number of iterations. As in [22], we use a cache of hard negative examples that is grown on each iteration to ensure that the update of w, b does not cycle from one iteration to the next.

5.3.3 Placement Refinement and Joint Matching

The above voting procedure produces a reasonable detection result (see Table 5.3 and associated discussion in experiments section). However, for efficiency reasons, the sampling of different possible placements, or correspondences of each model part to the image, is relatively coarse. Subtle but discriminative shape features, particularly those at a small scale, may not be measured well without more accurate alignment of the image to the model. Additionally, the set of image contours matched to each model part is independent of one another, while in fact they should be the same for each part. This leads us to several different strategies for improving both the alignment of the image to the model as well as the matching of image contours to the model for a specific detection $D = \{T_0, ..., T_n, \mathbf{x}_1^{\text{sel}}, ..., \mathbf{x}_n^{\text{sel}}\}$:

- More accurate part placement: with denser sampling of possible part placements, we can achieve more accurate localization of the object parts and better measure subtle shape features.

- Affine warping of the image contours to align with the model: while the space of deformations that the object may undergo is large, affine transformations are appealing as they can help with out-of-plane transformations of the object.

- Joint or consistent matching of image contours to each of the model parts: instead of computing a different $\mathbf{x}^{\text{sel}}$ for each model part, have a single $\mathbf{x}^{\text{sel}}$ or matching of image contours to all model parts.

More accurate part placement: For each part placement T_i, we can sample a set of additional nearby part placements $Z_i = \{T | ||T - T_i||_2 \leq \delta\}$, and repeat the voting procedure with these denser sets of placements for each part. The result is a more accurate set of part placements that can help better localize subtle but important shape features. We can write the voting as a modification of Equation 5.7, where each part placement T_i must lie in Z_i, or $T_i \in Z_i$:

$$S(R_j, w) = \max_{D|\mathbf{T}_0=R_j} \text{DetScore}(D) \Longrightarrow$$
$$\max_{\{\mathbf{T}_1 \in Z_1, ..., \mathbf{T}_N \in Z_n, \mathbf{x}_1^{\text{sel}}, ..., \mathbf{x}_N^{\text{sel}}\}} \sum_{i=1}^{N} \begin{bmatrix} w_i^{\text{def}} \\ w_i^{\text{app}} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} G(R_j, \mathbf{T}_i) \\ K(\mathbf{T}_i, \mathbf{x}_i^{\text{sel}}) \end{bmatrix} \tag{5.9}$$

Joint matching: For a fixed set of part placements, we can solve the many-to-one matching problem simultaneously for all parts. In essence, this is like treating the shape contexts of all the object parts as a single large histogram, and solving the many-to-one matching problem with this unified histogram. The result is a single $\mathbf{x}^{\text{sel}}$ that encodes which contours are matched to the object as a whole. Given an exsting detection $D = \{T_0, ..., T_n, \mathbf{x}_1^{\text{sel}}, ..., \mathbf{x}_n^{\text{sel}}\}$, we can write the maximization problem as:

$$\max_{\mathbf{x}^{\text{sel}}} \sum_{i=1}^{N} w_i^{\text{app}\mathsf{T}} K(\mathbf{T}_i, \mathbf{x}^{\text{sel}}) = \max_{\mathbf{x}^{\text{sel}}} \begin{bmatrix} w_1^{\text{app}} \\ w_2^{\text{app}} \\ \vdots \\ w_N^{\text{app}} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} K(\mathbf{T}_1, \mathbf{x}^{\text{sel}}) \\ K(\mathbf{T}_2, \mathbf{x}^{\text{sel}}) \\ \vdots \\ K(\mathbf{T}_N, \mathbf{x}^{\text{sel}}) \end{bmatrix} \tag{5.10}$$

This can also be approximated via the same linear program (considering of all the shape contexts together forming a single large histogram) as the usual single-part many-to-one matching.

Affine warping of image contours: To estimate a good affine transformation of the image to the model, we first enrich the set of correspondences of image to model. With the above part model, the number of correspondences is relatively few, typically on the order of about 10. However, with the known matching of image contours ($\mathbf{x}^{\text{sel}}$ from the joint matching step) to the model, we can compute shape contexts at many different locations on the object in the image using only edge points from the matched contours. If we densely sample these shape contexts on the image and match them precisely in the model, we can establish many good correspondences that can be used to estimate an affine transformation.

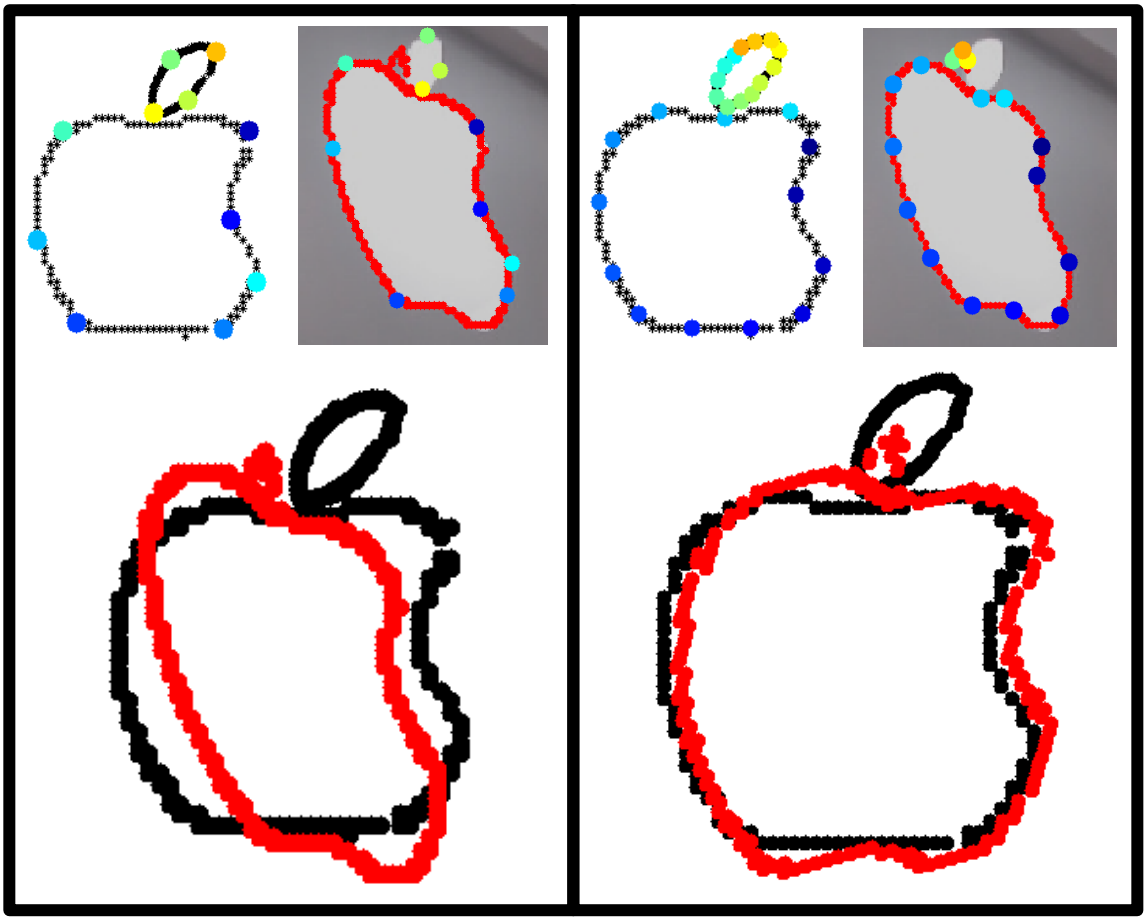


Figure 5.5: Affine transformation of image to model improves alignment. Left panel: at the top are shown the object shape model with part locations as colored dots and the placement of the parts ($\{T_i\}$) in the image, along with the matched image contours in red. Using these correspondences, the best rigid transform that aligns the image contours to the model is shown below, and we can see that the alignment is not very good. Right panel: at the top are shown denser correspondences between the model and the image, found via shape-context matching between the model shape and the matched image contours (red). Only high scoring correspondences, according to the Chi-Square score between the two shape contexts, are kept. From these correspondences, we can estimate an affine transformation that warps the image contours back to the object model (below). We can see that the alignment is much better under the new affine transformation.

Given the matched set of image contours, establishing correspondences between the image and the model reduces to the correspondence problem solved by Belongie et al. [6]. For each sampled shape context in the image, we try matching it at many different locations in the model using the Chi-Square similarity measure from [6]. Correspondences with a matching cost below a fixed threshold are retained, and used to estimate an affine transformation of the image to the model using a least-squares criterion. Figure 5.5 shows an example of the improved alignment due to estimation of an affine transformation. In the left panel are shown, clockwise from top left, the model shape with part locations in colored points, the placements of those parts in the image found via the original detection voting along with the matched image contours from joint matching, and lastly the rigid alignment of the matched image contours to the model shape using a rigid transformation estimated from the correspondence of part locations in the image to part locations on the model. We can see that alignment is rather poor, since this particular Applelogo instance has undergone out-of-plane rotation, and hence no rigid transformation could align it to the model shape. In the right panel of Figure 5.5 are, clockwise from top left, dense points on the model shape, the correspondences of those points to the points on the matched image contours in the image, and lastly the image contours registered back to the model using the affine transformation estimated from these new correspondences. We can see that the affine transformation does a very good job of correctly aligning the image and model shapes, thereby drastically improving the detection score of this particular Applelogos instance.

Joint matching and final evaluation: Given the final part placements, we can again perform joint matching of image contours to all the object parts. On the histogram comparison features and the geometric relationships of parts to the object center, we can train an SVM classifier to provide a final detection score.

5.4 Experiments

We implemented our method in MATLAB, using additional software packages obtained from the authors of [72] and [2], and plan to release the code. We tested our method on the ETHZ Shape Classes dataset ([26]; freely available online), with five classes: Applelogos, Bottles, Giraffes, Mugs and Swans. We follow the train/test split described in [45]; for training for each category we used the first half of the images from that category as positive examples, and an equal number of negative images chosen equally from the remaining classes. Each category had 32 to 86 training images. Model shape learning was first performed, and a detector was trained using the latent SVM formulation. During shape learning, we used a 4-by-4 grid histogram with trilinear interpolation (orientation binning in addition to spatial binning; similar to [18]). Each image averaged 85 contours. Using the linear programming approximation to the matching efficiency score, these candidates were pruned to a pool of about 100 candidates, for which the exact efficiency score was computed using an integer linear program solver (also in GLPK; [2]). Histogram difference and intersection features were weighted uniformly as 1.2 and 1, respectively. We used overlap threshold $t = 0.8$.

During detection, images were searched at 6 different scales, 2 per octave. Each part had up to 200 different possible placements in the image; for each part, placement, scale tuple, a separate linear program was solved, taking a few minutes per image. Latent SVM parameters w were initialized uniformly as in model shape learning, and convergence took 3-7 iterations. After training the initial detector, learning was done for part placement refinement, affine transformation estimation and joint selection using high-scoring detections from voting (< 200 detections). All our results used 0.5 overlap score threshold for determining if a detection bounding box overlaps with a ground truth bounding box (PASCAL criterion). Each detector was tested on remaining 169 (Giraffe) to 223 (Swan) test images.

We compare our approach against the reported results from [45] and the method of

[22] with the same train/test split. We show the precision/recall (PR) curves (Fig. 5.6) as well as plots of false positives per image (FPPI) vs. detection rate (DR), and also the result of [44] (which tested on full dataset). Our method is comparable in Applelogos/Bottles and substantially outperforms on Mugs/Giraffes/Swans which have large deformation. Table 5.2 shows the interpolated average precision (AP; as used in the PASCAL VOC Challenge) for the methods. Our APs for the five classes are (0.845/0.916/0.787/0.888/0.922; mean: **0.872**), much better than the next best result at (mean: 0.771; [45]). Table 5.1 compares DR at 0.3/0.4 FPPI for several methods. Our detection rates at 0.3/0.4 FPPI of (0.95/0.95; 1/1; 0.872/0.896; 0.936/0.936; 1/1) and mean across classes of **0.952/0.956**, are a substantial improvement over the results of [45], 0.919/0.932 and [52], 0.930/0.952 (hand-drawn models). We also outperform methods using hand-drawn models ([73], [52], [44]). Fig. 5.7 shows detections/segmentations from our method. Both internal and external contours (e.g. mug handle/outline) are segmented out.

To gain further insight into the results, we display selected detections (ordered by detection score in decreasing order) from the ETHZ test data for each category in Figure 5.8 along with the positions on the PR curve of those detections. We can see that the false positives tend to have the shape of the object we are looking for, while some true positives have low score due to object deformation (e.g., articulation of the Giraffe or out of plane rotation of Applelogos) or missing contours.

We also performed an ablative analysis of the different steps of the method: voting, refinement and joint selection, as seen in Table 5.3. While voting with discriminative training is itself effective, the additional of refinement and joint selection also produce substantial increases in performance. By contrast, removing learning drastically worsens the results.

5.5 Observations

Implementing any object recognition system requires a mix of scientific principles and engineering principles. For the descriptive shape learning, choosing the right histogram binning is an important detail. Too many small histogram bins make it difficult to match contours according to actual shape commonalities relevant to the object and also require very precise alignment, which is difficult with a single transformation provided by the ground truth bounding boxes. Therefore, a shape context binning pattern is not ideal as the small bins near the shape context center require precise alignment in order to record repeatable shape information. Instead, a grid histogram with medium-sized bins is advisable, as it does not require very precise alignment but can still record some useful shape information, enough to discover commonalities of shape among the various positive examples.

For shape discrimination, some of the most important bins are ones that contain no contour points in the vast majority of the positive examples. These bins can be assigned high weight reliably, while other bins that consistently contain a particular part (and the same part) of the object shape may have counts that vary substantially. As a result, they cannot be assigned as high a weight to achieve as high discriminability. One can view the bins that consistently have no contour points as "carving" out the actual object shape using negative regions, or description by subtraction. For the bins that consistently contain object shape, exploring different types of features, exploring different ways of describing the contents of the bin or the relative nature of the contents of bins could be useful. For example, differential features as used by Viola and Jones ([66]) might be more appropriate; in the linear programming formulation, a hinge penalty on the difference between neighboring shape context bins could be useful for encoding such a differential feature (the count of a particular bin should be larger than another bin, and if not, pay a penalty linear in the violation of the constraint).

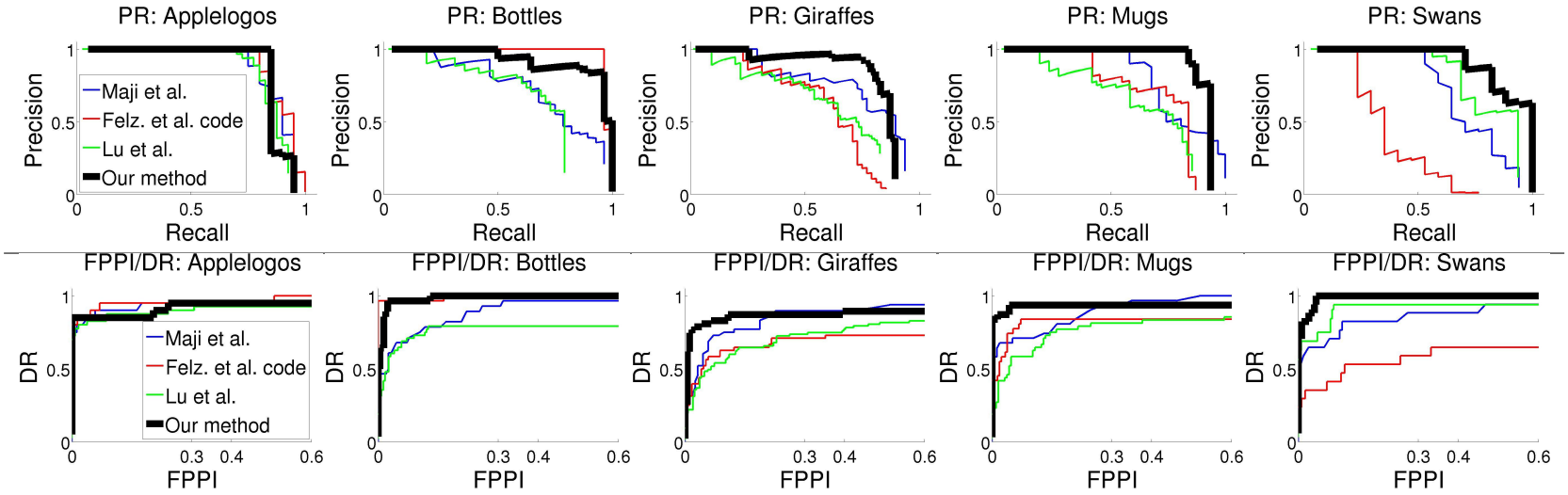


Figure 5.6: Precision vs. recall (PR; top row) and false positives per image vs. detection rate (FPPI/DR; bottom row) curves for our method, Maji et al. [45], code of Felz. et al. [22] and Lu et al. [44] on the ETHZ Shape Classes dataset. We can see that our method is comparable or significantly outperforms the other methods, particularly for those classes with significant deformation such as giraffes and swans.

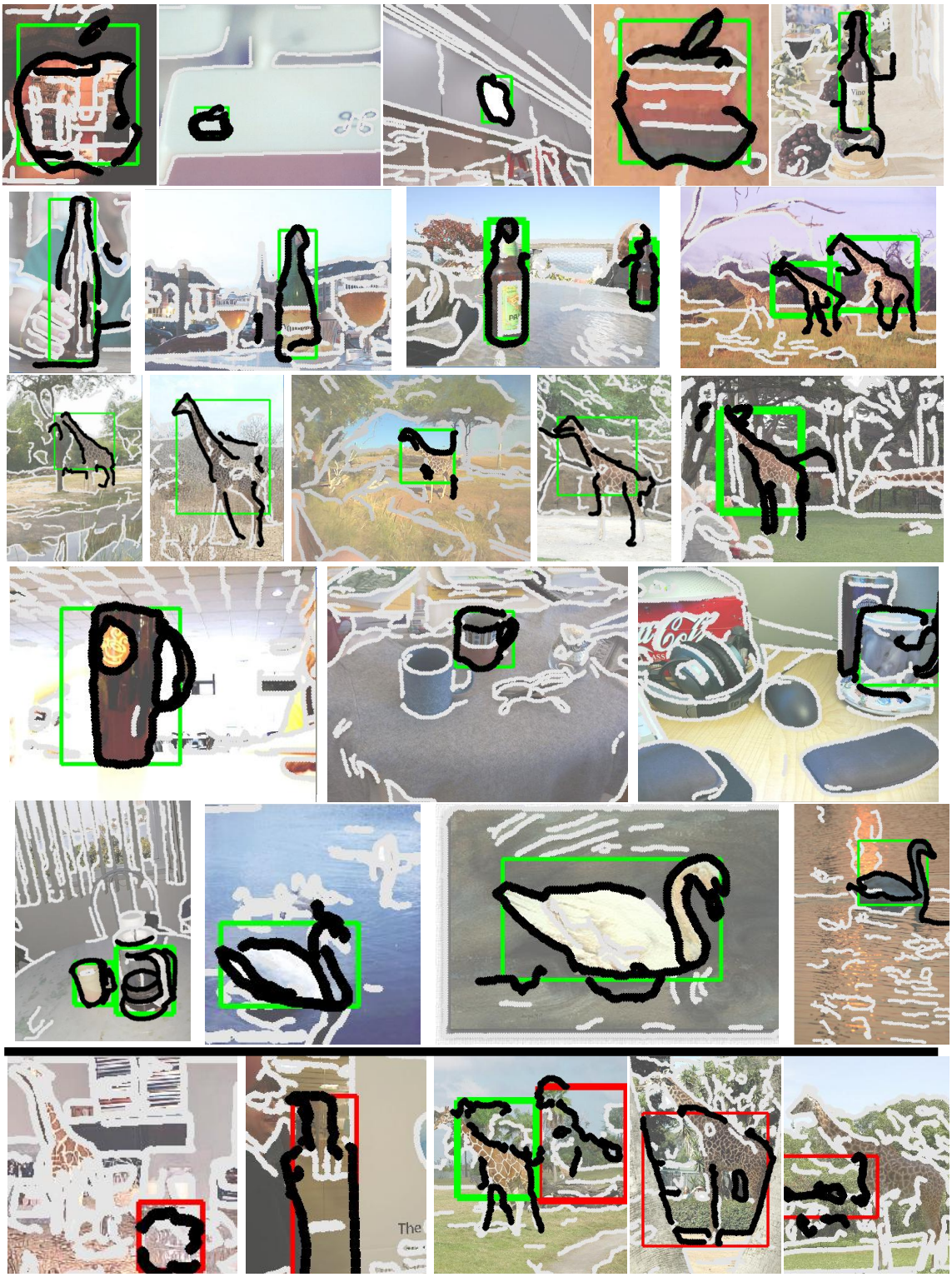


Figure 5.7: Some of our detection results on the ETHZ Shape Classes dataset. Each image shows segmented object contours and bounding boxes for one or more detections. Bottom row shows false positives for Applelogos, Bottles, Giraffes, Mugs and Swans (l-to-r); rest are true positives.

	Applelogos	Bottles	Giraffes	Mugs	Swans	**Mean**
This work	**0.95** / **0.95**	**1** / **1**	0.872 / **0.896**	**0.936** / 0.936	**1** / **1**	**0.952** / **0.956**
Maji et al. [45]	**0.95** / **0.95**	0.929 / 0.964	**0.896** / **0.896**	**0.936** / **0.967**	0.882 / 0.882	0.919 / 0.932
Felz. et al. [22] code	**0.95** / **0.95**	**1** / **1**	0.729 / 0.729	0.839 / 0.839	0.588 / 0.647	0.821 / 0.833
Gu et al. [33]	0.906 / -	0.948 / -	0.798 / -	0.832 / -	0.868 / -	0.871 / -
Ferrari et al. [26]	0.777 / 0.832	0.798 / 0.816	0.399 / 0.445	0.751 / 0.8	0.632 / 0.705	0.671 / 0.72
Stark et al. [64] 0.2 overlap	- / -	- / 0.944	- / 0.917	- / 0.845	- / 0.888	- / 0.899
Fritz et al. [28], 0.2 overlap	- / 0.899	- / 0.768	- / 0.905	- / 0.827	- / 0.754	- / 0.768
Lu et al. [44], hand-drawn	0.9 / 0.9	0.792 / 0.792	0.734 / 0.77	0.813 / 0.833	0.938 / 0.938	0.836 / 0.851
Ravishankar et al. [52], hand-drawn	0.955 / 0.977	0.909 / 0.927	0.912 / 0.934	0.937 / 0.953	0.939 / 0.969	0.930 / 0.952
Zhu et al. [73], hand-drawn	0.800 / 0.800	0.929 / 0.929	0.681 / 0.681	0.645 / 0.742	0.824 / 0.824	0.776 / 0.795

Table 5.1: Comparison of **detection rates for 0.3/0.4 FPPI** on ETHZ Shape Classes. Our method is tied for or leads all methods that do not use hand-drawings in 8 of 10 different detection rates for individual classes, and in both detection rates averaged across all classes. Unless otherwise noted, results were computed with 0.5 overlap score threshold and not using hand-drawn models.

	Applelogos	Bottles	Giraffes	Mugs	Swans	**Mean**
Our method	0.845	0.916	**0.787**	**0.888**	**0.922**	**0.872**
Maji et al. [45]	0.869	0.724	0.742	0.806	0.716	0.771
Felz. et al. [22] code	**0.891**	**0.950**	0.608	0.721	0.391	0.712
Lu et al. [44]	0.844	0.641	0.617	0.643	0.798	0.709

Table 5.2: Comparison of **interpolated average precision (AP)** on the ETHZ Shape Classes dataset. Our method has the highest AP in 3 of the 5 classes, and the highest mean across classes.

Components	**Avg. AP**	**Avg. Rec. at 0.3/0.4 FPPI**
Vote only	0.822	0.877 / 0.883
Vote+ref.	0.844	0.913 / 0.935
Vote+ref.+joint sel. (Tables 5.1 & 5.2)	**0.872**	**0.952 / 0.956**
Same as above (no training)	0.712	0.852 / 0.856
Voting only (no training)	0.574	0.765 / 0.790

Table 5.3: Ablative analysis of different components of our method on the ETHZ Shape Classes dataset. We can see that the additional steps of refinement (ref.) and joint selection (joint sel.) produce significant improvements in performance over voting alone (rows 1-3, all with discriminative training). Removing discriminative training produces substantially worse results (row 4), and removing both training as well as refinement and joint selection severly impacts performance (row 5).

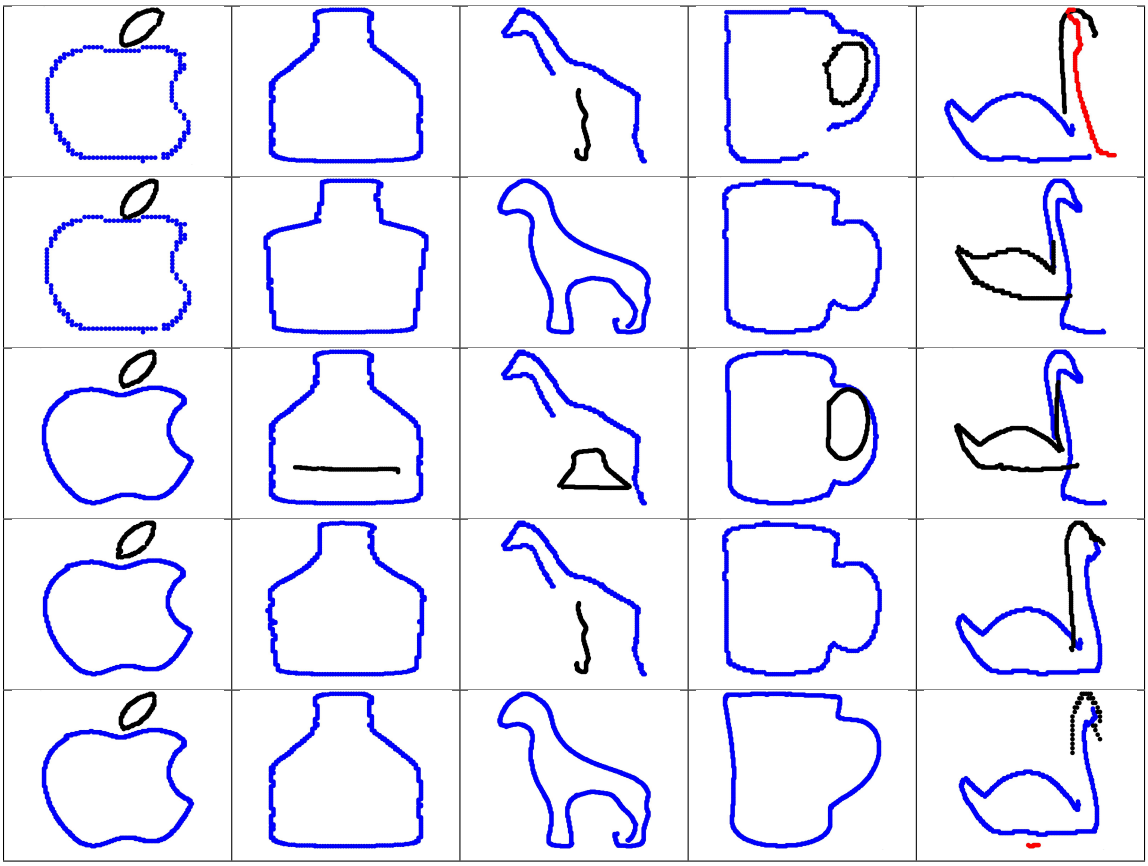


Table 5.4: Shape model learning results over several different training folds. To examine the variation of the shape model learning with different training images, we performed shape model learning with different sets of positive images. Each row represents a different training fold of half the positive images, which were randomly chosen except for the first row, which is the original model shape learning result seen before. Despite variation in the training set of images, the shape learning still produced clear object shape models for all of the classes, with some minor artifacts (e.g., in the Swan category) that could be cleaned up with a refined matching procedure.

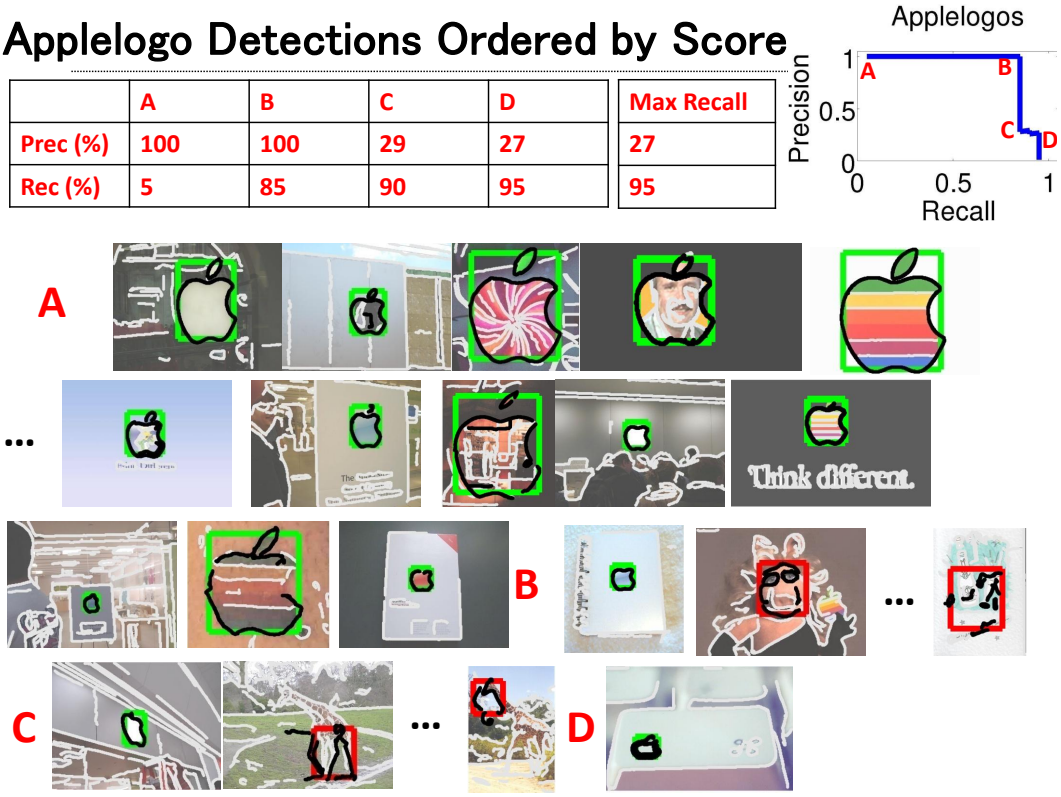


Figure 5.8: Listing of detection results for each class. In the upper right is the precision-recall curve, annotated at several different points with letters "A", "B", etc... Shown below are detections in descending order of detection score, along with the locations on the curve indicated by the letters. Green bounding boxes indicate true positives, while red bounding boxes indicate false positives. Selected contours for each detection are highlighted in black.

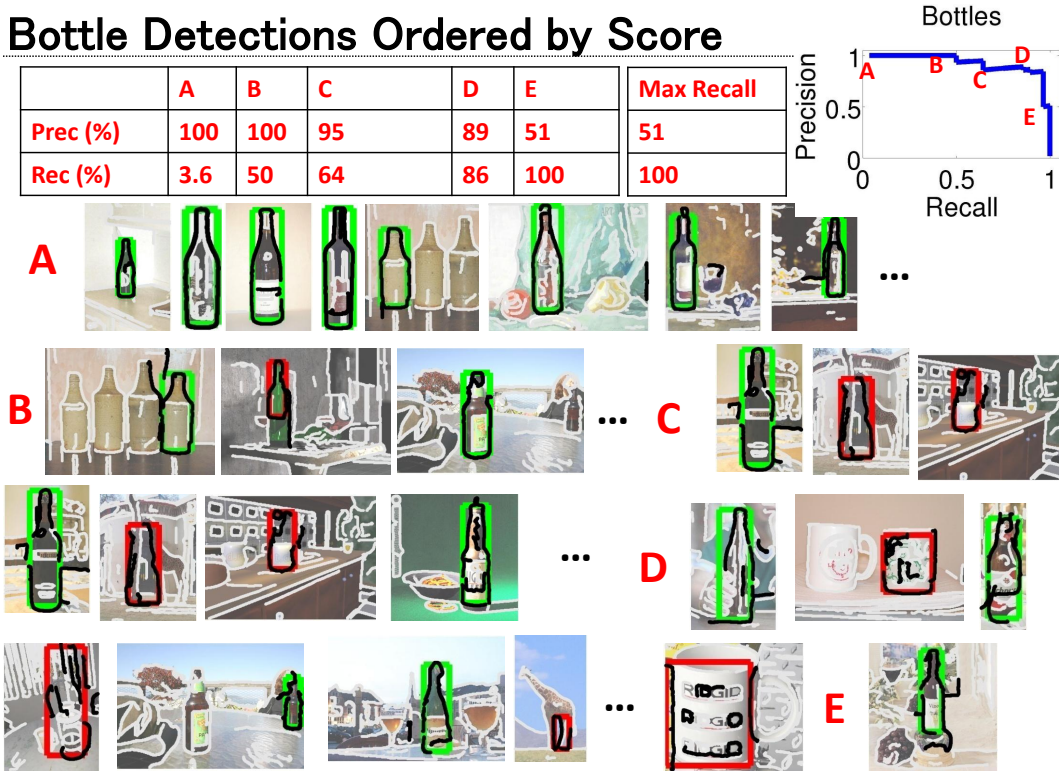
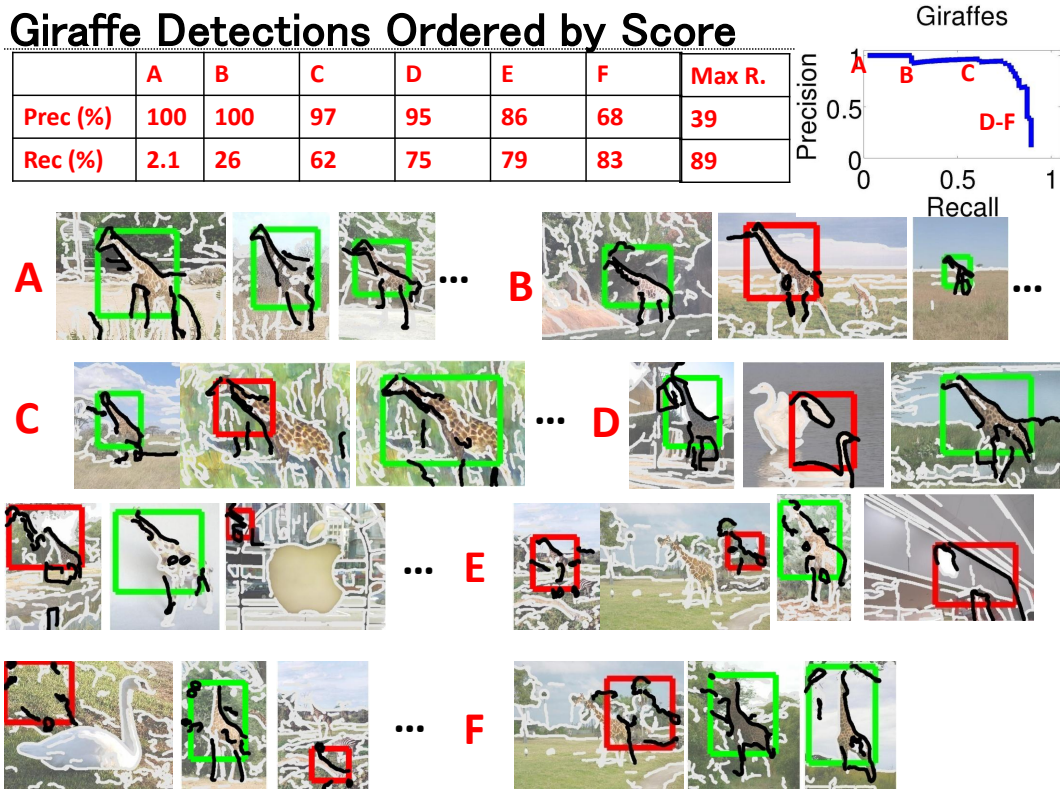


Figure 5.8: Listing of detection results for each class. In the upper right is the precision-recall curve, annotated at several different points with letters "A", "B", etc... Shown below are detections in descending order of detection score, along with the locations on the curve indicated by the letters. Green bounding boxes indicate true positives, while red bounding boxes indicate false positives. Selected contours for each detection are highlighted in black.



	A	B	C	D	E	F	Max R.
Prec (%)	100	100	97	95	86	68	39
Rec (%)	2.1	26	62	75	79	83	89

Figure 5.8: Listing of detection results for each class. In the upper right is the precision-recall curve, annotated at several different points with letters "A", "B", etc... Shown below are detections in descending order of detection score, along with the locations on the curve indicated by the letters. Green bounding boxes indicate true positives, while red bounding boxes indicate false positives. Selected contours for each detection are highlighted in black.

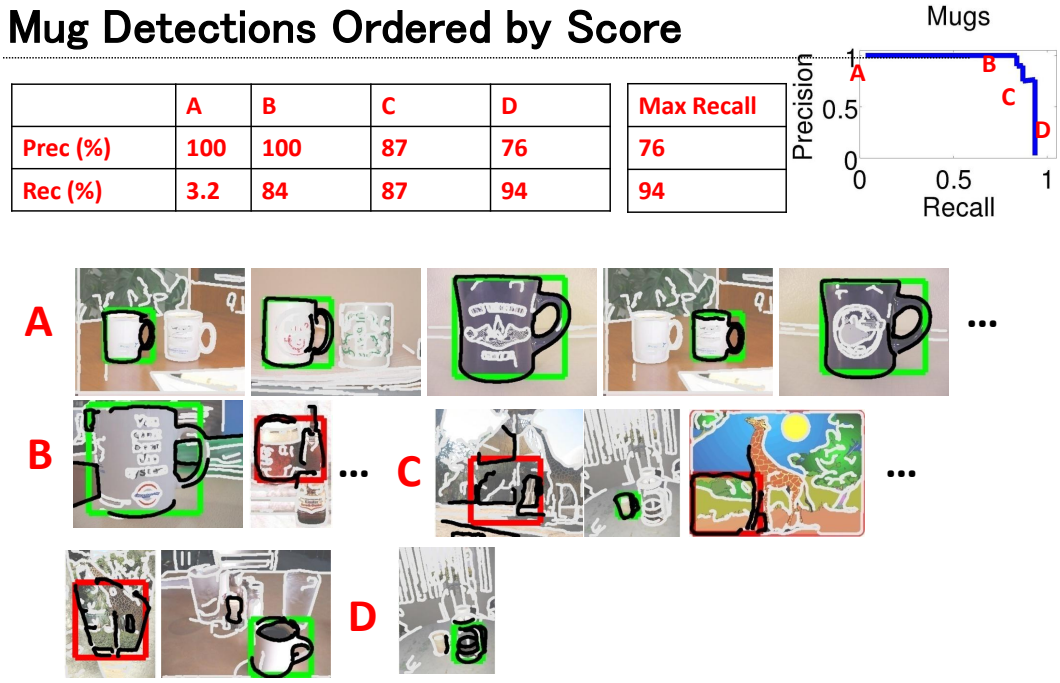


Figure 5.8: Listing of detection results for each class. In the upper right is the precision-recall curve, annotated at several different points with letters "A", "B", etc... Shown below are detections in descending order of detection score, along with the locations on the curve indicated by the letters. Green bounding boxes indicate true positives, while red bounding boxes indicate false positives. Selected contours for each detection are highlighted in black.

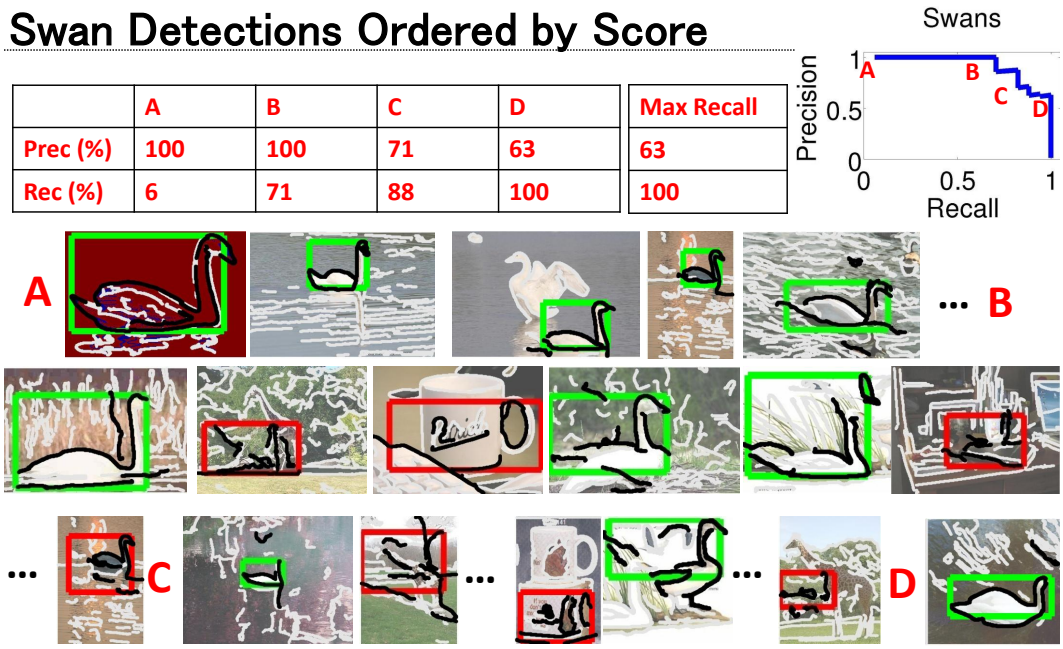


Figure 5.8: Listing of detection results for each class. In the upper right is the precision-recall curve, annotated at several different points with letters "A", "B", etc... Shown below are detections in descending order of detection score, along with the locations on the curve indicated by the letters. Green bounding boxes indicate true positives, while red bounding boxes indicate false positives. Selected contours for each detection are highlighted in black.

Chapter 6

Conclusion

6.1 Summary

In conclusion, we have demonstrated the importance of holistic matching of bottom-up image structures in addressing problems in perceptual grouping, human pose estimation, and object recognition.

We began by reviewing different types of bottom-up image structures, contours and segments, and algorithms for extracting them in images. Methods for evaluating holistic shape of these structures were discussed, including the many-to-many (one) matching framework of Zhu et al. [73] and the inner-distance shape context representation of Ling et al. [41].

Perceptual grouping is one area that can benefit from holistic evaluation of image contours, as seen in Chapter 3. Many-to-many matching was used to establish hypotheses for correspondences of contour points in one image to another in order to collect different hypotheses for the motion of the contour points. Based on these motions, the contours were segmented into different groups using min-cuts [12]. By comparing against a baseline that did not use many-to-many matching to score motion hypotheses, but just proximity of contour points in one image to points in another, a significant improvement in the groups of contours was observed due to

the many-to-many matching.

In Chapter 4, the inner-distance shape context was used to match image segments against exemplar shapes of different regions of the human body, even under substantial deformation due to articulation. A set of human-provided parsing rules allowed hypotheses of smaller regions of the body to be joined together in order to form increasingly larger regions, and eventually the entire body (except for the arms). By having a variety of different possible body regions (some overlapping), we were able to make maximal use of the image segments, as they tend to fragment in many different ways with respect to the human body.

Chapter 5 explored the learning of object shape and discriminative training of an object detector for detecting objects using image contours. From an input of positive images annotated with bounding boxes containing instances of a single object category, and negative images containing no instances, a two-step procedure of shape learning followed by detector training was described. During shape learning, "lucky" contours that captured large portions of the object shape were found and used to greedily construct a shape model that explained the positive examples well. The measure of explanation quality used the many-to-one matching framework. Given this shape model, the Latent-SVM learning algorithm ([22]) was used to discriminatively train an object detector from the positive and negative images. The latent variables included the positions of object parts as well as the actual contours that comprised the boundary of each object instance.

Recent work with Weiyu Zhang has revisited the human pose estimation problem from the perspective of using image contours along with regions and patches. While the regions used in Chapter 4 can often provide substantial information about the pose of a human, in more difficult images the regions tend to leak and do not capture shape as well as long contours. Contours are especially useful because they can be used to propose pose of limbs of the person, eliminating the need for explicit search over all possible positions of all body parts as approaches such as pictorial structures

[24] often use.

6.2 Future Work

There are several important directions for future efforts raised by the work in this thesis. One important theme that was not fully developed is the contrast between evaluating individual bottom-up structures vs. evaluating those structures only as a whole. In the parsing approach of Chapter 4, individual image segments provided important information about the pose of the human, while in the object detection work of Chapter 5, individual contours were never examined by the algorithm. The recent contour-based pose estimation work provides a bridge between these two, using individual contours to propose portions of the body pose, while using many-to-one matching to holistically verify pose hypotheses.

A second important issue is scaling to larger, more difficult datasets. While the ETHZ Shape Classes dataset used in Chapter 5 is a challenging dataset, larger datasets with more object categories exhibiting greater pose and 3D variation such as the PASCAL Visual Object Classes Challenge still present a formidable obstacle. The recent work on pose estimation took a step in addressing this by introducing a hybrid object model that included contours, patches and regions, all of which are likely necessary to effectively tackle more difficult datasets. Proposing 3-D object pose from detected image junctions is also an important task since this can dramatically reduce the search space for determining the object pose.

On the learning front, the shape learning method used in Chapter 5 learned only a single model for each category. Learning multiple models for each category and variations within each model as important areas for future work. Discriminative training for detection should also make sure of higher-level shape information, such as junctions and geometric relationships (e.g. parallelism, straight vs. curved, etc...) to provide better generalization. Also, better learning of the binning patterns for

the histograms used to measure object shape would also be useful since the choice of the log-polar binning pattern of the shape context or the uniform sampling of the grid histogram are well-motivated but somewhat ad-hoc. A boosting-like procedure could be developed to select the minimum number of necessary bins to provide good detection performance, thereby speeding up detection as well as reducing the number of parameters to be learned in the model (perhaps improving generalization).

Bibliography

[1] *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), 24-26 June 2008, Anchorage, Alaska, USA*. IEEE Computer Society, 2008.

[2] GLPK software. http://www.gnu.org/software/glpk/.

[3] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *CVPR*, pages 1014–1021, 2009.

[4] Ronen Basri. Recognition by prototypes. *International Journal of Computer Vision*, 19:19–147, 1996.

[5] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(4):509–522, 2002.

[6] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *PAMI*, 2002.

[7] Alexander C. Berg, Tamara L. Berg, and Jitendra Malik. Shape matching and object recognition using low distortion correspondences. *CVPR*, 2005.

[8] Irving Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94:115–147, 1987.

[9] Thomas O. Binford. Visual perception by computer. In *IEEE Conf. on Systems and Controls*, 1971.

[10] S. Birchfield and C. Tomasi. Depth discontinuities by pixel-to-pixel stereo. In *ICCV*, 1998.

[11] Eran Borenstein and Jitendra Malik. Shape guided object segmentation. In *CVPR 2006*.

[12] Yri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 2001.

[13] Song chun Zhu and David Mumford. A stochastic grammar of images. In *Foundations and Trends in Computer Graphics and Vision*, pages 259–362. Now Publishers, 2006.

[14] Timothy F. Cootes, Christopher J. Taylor, David H. Cooper, and Jim Graham. Active shape models-their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.

[15] James M. Coughlan and Sabino J. Ferreira. Finding deformable shapes using loopy belief propagation. In *ECCV (3)*, pages 453–468, 2002.

[16] Timothee Cour, Florence Benezit, and Jianbo Shi. Spectral segmentation with multiscale graph decomposition. In *CVPR 2005*.

[17] Timothée Cour, Praveen Srinivasan, and Jianbo Shi. Balanced graph matching. In *NIPS*, pages 313–320, 2006.

[18] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[19] M. Fatih Demirci, Ali Shokoufandeh, Yakov Keselman, Lars Bretzner, and Sven Dickinson. Object recognition as many-to-many feature matching. *Int. J. Comput. Vision*, 69(2):203–222, 2006.

[20] Gal Elidan, Geremy Heitz, and Daphne Koller. Learning object shape: From drawings to images. In *CVPR*, 2006.

[21] Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, 2006.

[22] Pedro Felzenszwalb, David McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.

[23] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Distance transforms of sampled functions. Technical report, Cornell Computing and Information Science, 2004.

[24] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005.

[25] Vittorio Ferrari, Frédéric Jurie, and Cordelia Schmid. Accurate object detection with deformable shape models learnt from images. In *CVPR*, 2007.

[26] Vittorio Ferrari, Fredric Jurie, and Cordelia Schmid. From images to shape models for object detection. In *PAMI*, 2009.

[27] M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structures. *IEEE Trans. Comput.*, 22(1):67–92, 1973.

[28] Mario Fritz and Bernt Schiele. Decomposition, discovery and detection of visual categories using topic models. In *CVPR* [1].

[29] Peter C. Gaston and Toms Lozano-Prez. Tactile recognition and localization using object models: The case of polyhedra on a plane. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:257–265, 1983.

[30] M. Gelautz and D. Markovic. Recognition of object contours from stereo images: an edge combination approach. *3D PVT*, 2004.

[31] Steven Gold and Anand Rangarajan. Graph matching by graduated assignment. In *CVPR*, pages 239–244, 1996.

[32] W.E.L. Grimson. Why stereo vision is not always about 3d reconstruction. In *MIT AI Memo, Technical Report AIM-1435*, 1993.

[33] C. Gu, J. Lim, P. Arbelaez, and J. Malik. Recognition using regions. In *CVPR*, 2009.

[34] V. Hedau, H. Arora, and N. Ahuja. Matching images under unstable segmentation. In *CVPR 2008*.

[35] Gang Hua, Ming-Hsuan Yang, and Ying Wu. Learning to estimate human pose with data driven belief propagation. In *CVPR 2005*.

[36] Longin Jan Latecki and Rolf Lakamper. Polygon evolution by vertex deletion. In *SCALE-SPACE*, 1999.

[37] Mun Wai Lee and Isaac Cohen. Proposal maps driven mcmc for estimating human body pose in static images. *CVPR 2004*.

[38] Y. J. Lee and K. Grauman. Shape discovery from unlabeled image collections. In *CVPR*, 2009.

[39] Bastian Leibe, Ales Leonardis, and Bernt Schiele. An implicit shape model for combined object categorization and segmentation. In *Toward Category-Level Object Recognition*, 2006.

[40] Marius Leordeanu and Martial Hebert. A spectral technique for correspondence problems using pairwise constraints. In *International Conference of Computer Vision (ICCV)*, volume 2, pages 1482 – 1489, October 2005.

[41] Haibin Ling and David W. Jacobs. Using the inner-distance for classification of articulated shapes. In *CVPR 2005*.

[42] C. Liu, W. T. Freeman, and E.H. Adelson. Analysis of contour motions. In *NIPS*, 2006.

[43] David G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004.

[44] ChengEn Lu, Longin Jan Latecki, Nagesh Adluru, Haibin Ling, and Xingwei Yang. Shape guided contour grouping with particle filters. In *ICCV*, 2009.

[45] S. Maji and J. Malik. A max-margin hough tranform for object detection. In *CVPR*, 2009.

[46] G. Mori. Guiding model search using segmentation. In *ICCV 2005*.

[47] Greg Mori, Xiaofeng Ren, Alexei A. Efros, and Jitendra Malik. Recovering human body configurations: combining segmentation and recognition. In *CVPR 2004*.

[48] Greg Mori, Xiaofeng Ren, Alexei A. Efros, and Jitendra Malik. Recovering human body configurations: Combining segmentation and recognition. In *CVPR (2)*, pages 326–333, 2004.

[49] Andreas Opelt, Axel Pinz, and Andrew Zisserman. A boundary-fragment-model for object detection. In *ECCV*, 2006.

[50] Deva Ramanan. Learning to parse images of articulated bodies. In *NIPS 2007*.

[51] Deva Ramanan, D. A. Forsyth, and Andrew Zisserman. Strike a pose: Tracking people by finding stylized poses. In *CVPR 2005*.

[52] Saiprasad Ravishankar, Arpit Jain, and Anurag Mittal. Multi-stage contour based detection of deformable objects. In *ECCV*, 2008.

[53] Xiaofeng Ren, Alexander C. Berg, and Jitendra Malik. Recovering human body configurations using pairwise constraints between parts. In *ICCV 2005*.

[54] Rémi Ronfard, Cordelia Schmid, and Bill Triggs. Learning to parse pictures of people. In *ECCV 2002*.

[55] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

[56] D. Sherman and S. Peleg. Stereo by incremental matching of contours. *PAMI*, 1990.

[57] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000.

[58] Jianbo Shi and Jitendra Malik. Motion segmentation and tracking using normalized cuts. In *ICCV*, pages 1154–1160, 1998.

[59] Jianbo Shi and Jitendra Malik. Motion segmentation and tracking using normalized cuts. In *ICCV*, pages 1154–1160, 1998.

[60] Jamie Shotton, Andrew Blake, and Roberto Cipolla. Multiscale categorical object recognition using contour fragments. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(7):1270–1281, 2008.

[61] Leonid Sigal and Michael J. Black. Measure locally, reason globally: Occlusion-sensitive articulated pose estimation. In *CVPR 2006*.

[62] P. Srinivasan and J. Shi. Bottom-up recognition and parsing of the human body. In *CVPR*, 2007.

[63] Praveen Srinivasan and Jianbo Shi. Bottom-up recognition and parsing of the human body. In *EMMCVPR*, pages 153–168, 2007.

[64] Michael Stark, Michael Goesele, and Bernt Schiele. A shape-based object class model for knowledge transfer. In *ICCV*, 2009.

[65] Lorenzo Torresani, Vladimir Kolmogorov, and Carsten Rother. Feature correspondence via graph matching: Models and global optimization. In *ECCV (2)*, pages 596–609, 2008.

[66] Paul A. Viola and Michael J. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR (1)*, pages 511–518, 2001.

[67] John Y. A. Wang and Edward H. Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3(5):625–638, 1994.

[68] J.Y.A. Wang and E.H. Adelson. Layered representation for motion analysis. In *CVPR*, 1993.

[69] Yair Weiss. Smoothness in layers: Motion segmentation using nonparametric mixture estimation. In *CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, page 520, Washington, DC, USA, 1997. IEEE Computer Society.

[70] John M. Winn and Nebojsa Jojic. Locus: Learning object classes with unsupervised segmentation. In *ICCV*, 2005.

[71] Jiayong Zhang, Jiebo Luo, Robert Collins, and Yanxi Liu. Body localization in still images using hierarchical models and hybrid search. In *CVPR 2006*.

[72] Q. Zhu, G. Song, and J. Shi. Untangling cycles for contour grouping. In *ICCV 2007*.

[73] Qihui Zhu, Liming Wang, Yang Wu, and Jianbo Shi. Contour context selection for object detection: A set-to-set contour matching approach. In *ECCV (2)*, pages 774–787, 2008.