Exploiting Object Dynamics for Recognition and Control

by

Philipp Robbel

M.Sc., University of Edinburgh (2005)

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2007

Author ______________________________

Program in Media Arts and Sciences
August 11, 2007

Certified by ______________________________

Deb Roy
Associate Professor of Media Arts and Sciences
Program in Media Arts and Sciences
Thesis Supervisor

Accepted by ______________________________

Deb Roy
Chair, Departmental Committee on Graduate Students
Program in Media Arts and Sciences

Exploiting Object Dynamics for Recognition and Control

by

Philipp Robbel

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
on August 11, 2007, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

This thesis explores a new approach to object recognition for active vision systems that integrates information across multiple observations of an object. The approach exploits the order relationship between successive frames to derive a classifier based on the characteristic motion of local features across visual sweeps. This motion model reveals structural information about the object that can be exploited for recognition. The first contribution of this thesis is a recognition system that extends invariant local features (shape contexts) into the time domain by integration of the motion model. Second, an entropy-based view selection scheme is presented that allows the vision system to "skip ahead" to highly discriminative views. Evaluations on two datasets demonstrate that both the motion model and active view selection yield higher-quality hypotheses about object categories quicker than a baseline system that treats object views as unordered streams of images.

Thesis Supervisor: Deb Roy
Title: Associate Professor of Media Arts and Sciences, Program in Media Arts and Sciences

Exploiting Object Dynamics for Recognition and Control

by

Philipp Robbel

The following people served as readers for this thesis:

Thesis Reader____

Yuri Ivanov
Mitsubishi Electric Research Laboratories
Cambridge, MA

Thesis Reader____

Nicholas Roy
Assistant Professor of Aeronautics and Astronautics
Department of Aeronautics and Astronautics

This thesis is dedicated in loving memory to my grandfather, a true role model.

◇

Acknowledgments

During the course of this thesis I have been supported by a large number of people, all of whom I wish to thank in the following:

Professor Deb Roy acted as the advisor for this thesis and offered invaluable ideas, feedback, literature pointers and motivation throughout this work. I am particularly thankful for his help in establishing the "big picture" and offering structural advice when I couldn't see the forest for the trees. I also thank my readers, Yuri Ivanov and Nick Roy, for providing additional feedback on this thesis despite the short notice from my side.

I am also thankful to a number of friends and supporters: the Cogmac group members, particularly Kai-yuh Hsiao and Soroush Vosoughi, for their knowledgeable advice and the positive and friendly working atmosphere, my friends at MIT, especially Pankaj Sarin, Durga Pandey and Larry Chang, for the wonderful time in Cambridge, and my friends from Edinburgh and beyond for cherished memories.

Finally, my thanks go to those people that matter most. Dear Gabriela, parents, brothers, grandparents, and family: without your help and continued support I would not be where and who I am today.

Philipp Robbel

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis is about object recognition of active perception systems, such as autonomous robots. We understand systems to be *active* if they have the ability to directly influence the signals that they receive about the world they are embedded in. This ability to exert control over the perceptual process – for example, by reaching into the world or by choosing appropriate viewpoints – distinguishes these systems from passive observers such as fixed surveillance cameras.

The systems we are working with additionally operate under stringent time constraints. Concretely, the work in our group is driven by the motivation to build a conversational robot that can communicate in human-like ways about the changing space around it. Maintaining an up-to-date model of the world in this scenario requires the lower-level perceptual processes to be reliable and fast which frequently drives our choice of algorithm for fundamental tasks such as segmentation of the visual input and object recognition. The research we describe in this thesis is therefore valid for other systems that operate under analogous constraints, such as future service robots or similar machines.

In this thesis, we focus our attention on the visual sense and strive to implement an object recognition system that fulfills the requirements outlined above. This recognition system will integrate information from multiple observations to maximize the classification reliability of the objects presented to it. We give a more in-depth motivation of our work in the following section.

1.1 Motivation

The newest robot in our lab named Trisk is equipped with a number of sensors and actuators that are very loosely based on human physiology. In particular, the robot consists of a 6-degree-of-freedom (DOF) arm, a 4-DOF actuated head with stereo vision cameras, and a three-fingered hand with 6-DOF force-torque sensing abilities at the fingertips (cf. Figure 1-1).

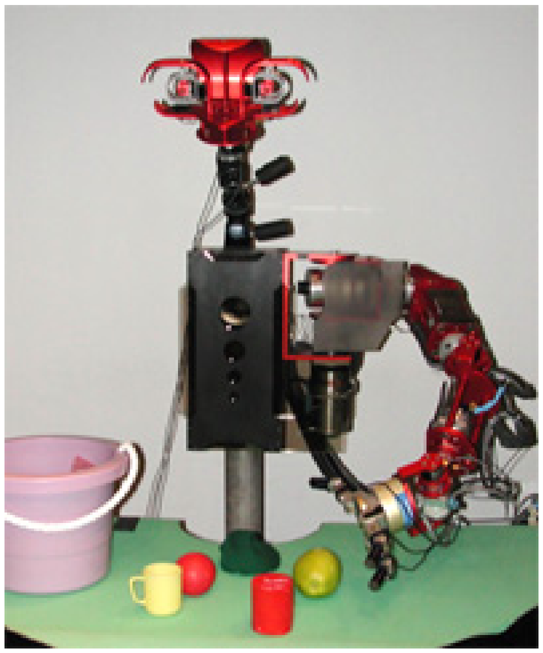


Figure 1-1: Trisk, the current research platform of our group.

Trisk's operating space consists of the table surface in front of it into which objects can be placed for the robot to interact with. The system receives commands through natural language (among others, “pick-up” or “group” commands) that rely on object recognition at one point during the control cycle. This recognition component of the overall system is required to detect *object class membership* for a number of household items, such as different kinds of cups or toys, as well as canonical shapes including spheres, cylinders, and cubes that were introduced for grasping experiments. This set spans fairly textured and non-textured objects that may appear in different sizes and orientations on the working surface.

With its two cameras, Trisk is able to perceive depth and to track objects in full 3D space. At the current point in time, the object recognition component makes use of a single snapshot from one of the cameras after the head has been positioned at a default top-down table view location. Requests for recognition are initiated by the higher cognitive layers of the robot and arrive at various time intervals depending on the vocal command and the number of objects on the table. The significant inter-class variability among our objects, mostly with respect to texture and color, led to the decision to rely predominantly on contour information during the robot’s recognition attempts. Recognition therefore involves an image segmentation step where pixels in the raw camera image are grouped into discrete regions, an edge extraction step where object contours are obtained, and finally a matching step where the new contour data is compared with the known contour models from a database.

Tests with our shape recognition system have shown that classification errors are frequently the result of one or more of the following conditions:

- Bad segmentation of the source image, due to changes in lighting conditions between training and testing phase of the system

- Noisy edge detection or lack of edges due to changes in contrast, specular highlights, etc.

- Ambiguous sensory input, i.e. the obtained view shares characteristics of at least two object classes

The last of these three cases is demonstrated in Figure 1-2 below. It can be seen that for different viewpoints, objects can drastically change their appearance so that classification based on a single random view is inherently unreliable. This effect is worsened under the segmentation and edge extraction issues outlined above.

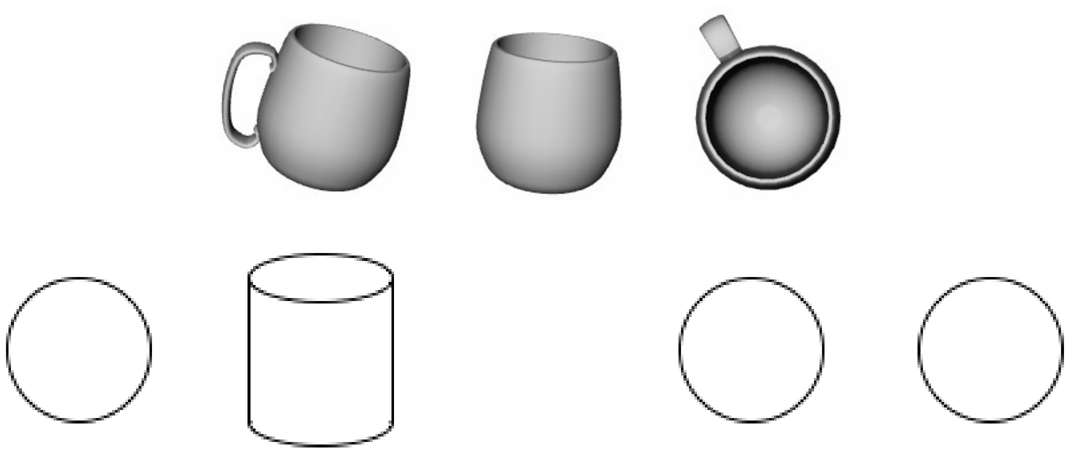


Figure 1-2: Change in object appearance for synthetic and non-synthetic objects. *Below:* A sphere needs at least two views to be distinguished from a cylinder—both share the same top-down view (right side). *Above:* A cup loses its characteristic handle and approaches similarity to a cylinder (center) and a sphere (right side).

The driving motivation for this thesis is based on this observation that reliance on a single frame is generally problematic in real-world scenarios. Instead, we believe that a probabilistic measure aggregated *over time* and based on different observations on the view sphere can feasibly improve on the object recognition performance. In this thesis we strive to document how such a system can be implemented and what performance gains can be achieved, as detailed in the following problem statement.

1.2 Problem Statement

The problem that this thesis addresses is the implementation of an object recognition system that leverages the capabilities of an active vision platform for recognition problems of the kind outlined previously. To achieve the time constraints we face with our interactive robot platform, we pursue two ideas that are core to this thesis.

First there is the observation that the class of cameras we consider here do *not* sample the view sphere independently but are instead controlled along connected trajectories. Essentially, they perform *sweeps* over the object that vary from trajectory to trajectory. Modern view-based object recognition algorithms generally operate on single images, as detailed in Chapter 2. Contrary, we attempt to exploit as much information as possible about the image acquisition process and additionally integrate *inter-frame* information into our object classification scheme. Our assumption is that the way an object transforms under specific camera motions (what we refer to as "object dynamics") allows to disambiguate the object class faster than treating object views as independent observations. Essentially, we are exploiting the knowledge that frames are connected in space and time and postulate that object feature motion leaves a characteristic imprint that we refer to as *sensory trace* in this thesis. The specific implementation we pursue here equips the robot system with a motion catalog that it utilizes to record sensory traces of various objects. At recognition-time motions from this catalog are used to disambiguate the object at hand. We therefore treat recognition as a sequence matching problem that involves recorded and (partially) observed sensory traces.

Second, an active perception system has not only access to how an object transforms over time but additionally knows the sensorimotor context (i.e., motor commands to the head) that yield the respective transformations. A valid question to ask is therefore where to guide the camera in order to classify the objects as quickly as possible. For purposes of this thesis, we strive to develop a quality metric that describes the usefulness of a particular viewpoint for disambiguation. This knowledge will allow us to switch between trajectories from the motion catalog as we attempt to classify a novel object.

1.3 Accomplishments

In this thesis we provide a review of the major historical and current object recognition approaches that have appeared in the literature. It is shown that many of these are designed for single object views from constrained viewpoints and are not necessarily optimal for our active vision scenario. We then present and implement an recognition scheme that extends invariant local features (*shape context*) into the time domain by making use of the order relationship between successive object views from the camera. For our system, we move a robot head along a set of pre-planned trajectories (the *motion catalog*) and record local features and feature motion across time. We attempt to exploit feature motion as an additional source of information about the structure of an object when matching a novel object with the known object database. To the best of our knowledge, combining this *inter-frame* information with local feature-based matching for object recognition has not been suggested before in the literature.

A second implementation associates control commands with the views in the object database and steers the camera to highly discriminative viewing positions in the motion catalog. Using two datasets, one standard (ETH-80) and one collected from the robot head, we show that both feature motion and active view selection achieve a higher-quality hypotheses about the presented object quicker than a baseline system treating object views as an unordered stream. Quality is judged by the entropy of the posterior distribution over object categories resulting from a number of views of the object.

Lastly, we develop a number of tools in the C++ and C# programming languages to allow the user to easily train the recognition system and to visualize the classification results.

1.4 Thesis Outline

In Chapter 2, we introduce past and present approaches to object recognition. We distinguish techniques that detect specific object instances from those that perform category-level recognition. We conclude the review with comments on our particular requirements and give the rationale for the recognition system we develop in this thesis.

In Chapter 3, we develop an object recognition system based on local features and the motion of features across neighboring frames. We derive a probabilistic classifier that joins both sources of information and which allows on-line updates of the posterior class estimate.

In Chapter 4, we join the presented recognition system with the ability to steer the camera to characteristic views of an object. These views are taken to be those leading to quickest disambiguation between object categories. Next to a description of entropy-based view selection, we also demonstrate in an experiment the validity of this approach.

In Chapter 5, we present the experimental results of our object recognition schemes on two datasets. The first is a subset of a standard dataset with eight object categories (ETH-80). The second is a dataset collected from our robot's active vision head and consists of eleven categories with 1010 object views in total. Our results encourage the use of feature-motion information as an additional source for disambiguating between objects.

Chapter 2

Current and Historical Approaches to Object Recognition

This chapter presents an overview of the broad field of object recognition, summarizing both historical and state-of-the-art approaches. Throughout, we comment on the practicability of the reviewed approaches to our problem domain. The goal of this chapter is to establish where and how our work fits into the larger picture of the ongoing computer vision research.

The body of literature on object recognition over the past 40 years is both vast as well as diverse in their approaches to the problem. Our review includes some of the predominant work of that time but excludes spun-off disciplines such as face or handwritten character recognition. It is apparent that the research focus has shifted gradually as more processing power became available and as statistical approaches entered the field. Beginning with the detection of specific object *instances* on backlit tables for industrial purposes, the majority of work is now geared towards object *category* recognition in natural scenes. Despite that shift in focus, some of the earlier, mostly geometry-based work has not become irrelevant: as outlined in [31], systems

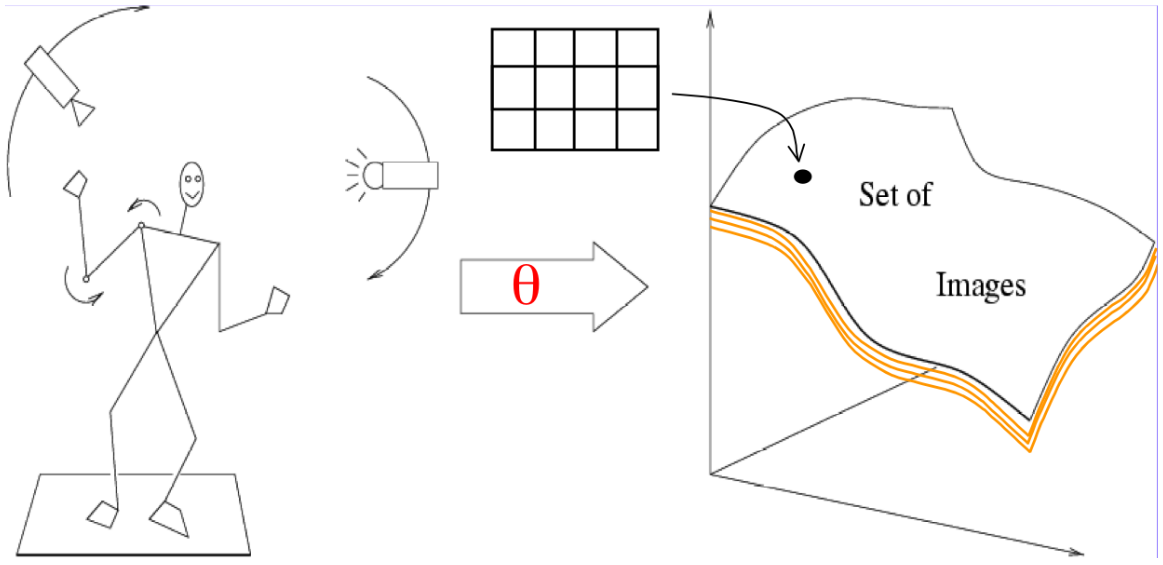


Figure 2-1: From [35]. The general object recognition problem.

like the 1972 *MIT copy demo* [51] where a robot observed a set of stacked blocks and rebuilt the same structure from a set of unordered blocks, have not been attempted yet with the more recent, appearance-based recognition methods.

The general 3D object recognition problem that we refer to in this summary is shown in Figure 2-1. The pixel-array impression of the object depends largely on a set of parameters, both external and internal to the object. Among external parameters are camera pose, illumination and occlusion while internal parameters refer to the joint angle setting of the stick figure in the image, for example. The set of images – or, *appearance manifold* – that a certain object evokes, depends on these parameter settings, collected in the vector θ. Models proposed in the literature differ in the degree to which this manifold structure is approximated and we will frequently refer back to the paradigm of Figure 2-1.

We can divide the approaches to object recognition along different dimensions such as whether they operate on range or intensity images, whether 3D geometry or solely appearance-based information is utilized, or based on their invariance to different image transforms, among others. The broad two classes of algorithms we distinguish in this review are those for object instance recognition and those for object category recognition which roughly follows the historical trajectory of research in this field.

2.1 Object instance recognition

We categorize as historical those methods that are based on static geometrical (CAD) models or global appearance models while we consider as current those approaches that rely on local appearance information (e.g. the distinctive texture) at a number of chosen interest points. This division is made clear in the following two sections.

2.1.1 Historical approaches

Hypothesize-and-test: The early era of object recognition mostly used manually-created geometric models of simple objects. Recognition then boils down to a bottom-up process consisting of contour extraction in the image and a template matching procedure of the stored CAD models against the observed image. With reference to Figure 2-1, recognition becomes a search problem over 3D position, orientation and scale (i.e., θ) that proceeds by *hypothesizing* an object and θ based on observed line features, projecting the 3D model into the manifold and *testing* the resulting appearance against the observed image. A typical candidate for systems from this era that rely heavily on consistent line features in the image is given in Roberts' 1963 PhD thesis [39]. Others work directly with 3D information from range images but essentially pursue a similar template matching approach against the stored 3D models [10].

Alignment: Given that early object recognition was formulated as a search problem, strategies were devised to reduce the search space based on insights into projective geometry. Researchers exploited the fact that the perspective camera model allows to derive the relation between a calibrated camera and the object's pose based on only a three-point *correspondence* between object and 3D model. Making use of this constraint, different groupings of point triples must result in consistent pose estimates so that the set of consistent θs (and, together with that, backprojections of the model into the image for verification) can be reduced. This approach is applied by Huttenlocher and Ullman [49], among others.

Another approach from that era that cuts down the search space via constraints appeared with the *interpretation tree* [22]. Associations between object and model features (such as lines) define branches of a tree structure. Each node in the tree corresponds to an object-to-model feature assignment and a set of conditions that is imposed on the assignment, for instance that the difference in line lengths has to be below a threshold. The tree is then traversed in a depth-first manner until a leaf node is reached, denoting a consistent feature assignment. As soon as a constraint is violated during traversal, the complete respective branch is removed from the tree.

Pose Clustering: Similar to alignment above, the set of objects to backproject into the image for verification is reduced by exploiting viewpoint consistency. For each object there is an associated accumulator array with buckets representing a particular pose of that object. As before, each feature grouping allows to hypothesize an object pose and now votes for that pose in the accumulator array, similar to voting in Hough space during the Hough transform. These votes can be weighted to favor those stemming from reliable feature groupings. For all objects which have sufficient votes at a particular pose, backprojection is executed to verify the match, as before [47].

All of the previously mentioned approaches share their reliance on search through the model database and the pose space associated with each model. Invariance to scale is generally achieved with another layer of exhaustive search through scale space.

To avoid this explosion of the hypothesis space for θ, a large amount of work has focused on *invariants*, i.e., properties of an object that remain the same under different transforms such as orientation and scale. Ideally, different choices of parameter vector θ in Figure 2-1 would project to a single, identical *point* in appearance space rather than span a larger appearance manifold structure.

Geometric invariants: Performing model search independent of pose and other transforms has the potential to significantly reduce the search space. Different geometric properties admit invariance to affine camera transforms, for example a set of coplanar points: given a coordinate frame defined by three of these points in the plane, the coefficients of the other points expressed in that basis is the same for any affine transform of the plane [21]. However, while affine invariants of this kind hold for planar objects, they do not in general extend to true 3D shapes [41].

A successful recognition system for planar objects that exploits invariance properties is presented in [42]. Compared to the methods mentioned previously, the search effort is reduced by moving away from establishing and matching feature groupings to matching invariants.

Geometric hashing: As detailed in Wolfson and Rigoutsos' overview [52], hashing can speed up the run-time recognition performance. In an initial processing step that is applied to every object i in the model database, the algorithm exhaustively computes all possible point triples (i.e., each valid *basis* $\mathbf{b}$) and computes the coefficients $\vec{v}$ of all other object points in that basis. Each distinct result is recorded in a *hashtable* under the mapping $\vec{v} \rightarrow (\mathbf{b}, i)$.

Given a novel object to identify at run-time, an arbitrary point triple is selected as basis $\mathbf{b}'$ and coefficients $\vec{v}'$ are derived for all object points with respect to this basis. Each $\vec{v}'$ then casts a vote for the particular object-basis tuple that it is associated with in the hashtable. Finally, for any such tuple $(\mathbf{b}, i)$ that received a large amount of votes, the familiar backprojection of model i into the image is performed for final verification. By nature of the voting scheme, the geometric hashing algorithm demonstrates a degree of invariance to occlusion but is not robust to noisy or cluttered background scenes [21].

All previously mentioned recognition methods rely on either explicit or derived knowledge of the 3D model structure. Besides these geometrical approaches to object recognition, there are two notable exceptions from that era that instead rely on object *appearance* – i.e., image intensities – for instance recognition. The former and more simple approach assumes a constant appearance over varying θ while the latter attempts to explicitly capture all appearances that an object can evoke.

Correlation-based template matching: This approach is an early example of a *global* appearance-based recognition strategy and was used successfully for industrial part-picking systems. Recognition proceeds by laying (or sliding) model template images over the observed image and calculating the degree of match via normalized cross-correlation. This operator takes its maximum value for arrangements where the template matches the image exactly, up to a constant scale factor. An efficient implementation makes use of the close relation between cross-correlation and convolution and exploits the fact that the latter operation is equivalent to a multiplication in the frequency domain, enabling fast recognition.

Aspect graphs: Exact 3D aspect graphs attempt to capture the entire structure of the appearance manifold from Figure 2-1 assuming a canonical unit viewing sphere around the object. They were introduced by Koenderink and Van Doorn in 1979 [26].

Each view of an object gives rise to an *aspect*, defined as the set of characteristics visible from that point. For a simple planar 2D shape, for example, these characteristics could correspond to the number and the ordering of the contour points visible from that viewpoint. For a range of adjacent views, these characteristics remain unchanged, yielding equivalence classes of aspects. An *aspect graph* contains a representative aspect from each equivalence class as nodes and connects the neighboring classes. The dividing lines between adjacent equivalence classes are referred to as *visual events* where the given object undergoes an appearance (or topology) change. Under certain assumptions, these visual events can be obtained in a mathematically rigorous way with the help of differential geometry and singularity theory [21].

The fact that aspect graphs attempt to systematically capture *all* views that have some geometric relevance at a predetermined scale is also a major stumbling block. Even simple objects can result in very complex aspect graphs, particularly when more complex surfaces such as curves are involved. Major research focus has also shifted away because of the fact that small deformations of the object can result in largely different aspect graphs, complicating their use in object category recognition. However, *approximate* aspect graphs that discard some of the views have been used successfully for instance recognition, for example in [23].

2.1.2 Current approaches

We saw that a lot of early work characterizes objects by their geometrical 3D model. This is in accordance with some of the early psychological theories of image understanding in humans (e.g., [7]) which base object recognition on the detection of canonical volumetric bodies in the image (recognition by components). More recent vision systems follow a paradigm that is more in accordance with the model presented by Riesenhuber and Poggio in [38] where psychophysical evidence for a *view-based* approach to human object recognition is cited.

There is also a fundamental problem with counting on reliable extraction of *high-level* features (such as connected lines in an image for recognition) from bottom-up processes like edge detection and segmentation. While high-level features are positive in that they tend to reduce the matching effort between model and object, they also bring with them the problem of robustly extracting those features. Lines tend to get fragmented or occluded in natural scenes, for example, where control over lighting effects and background is generally not possible.

With a shift in research to recognizing objects in less controlled environments, view- or appearance-based detection mechanisms have become a major research focus and are predominantly used today together with statistical machine learning tools. *Global* appearance models use the entire image intensity patch (or a downsampled version thereof) for classification and are therefore prone to occlusion (see, e.g., [37] for an SVM classifier trained on global object appearance). This problem can be circumvented with *local* intensity features that describe the object appearance at a set of localized points. Here, maching can be made reliable even if some local features are occluded.

Detectors and Descriptors: In recent approaches, local appearance is encoded by a *descriptor* at a set of distinct *interest points*. These points are determined by a *detector* that searches for low-level features in the image that are stable enough to be found repeatably across different views of the same object. Ideally, both descriptor and detector should be invariant to scale and other transforms, yielding a set of *invariant local features* of an object. The spatial arrangement of these features is generally neglected and correspondance between model and test views established based on the similarity of descriptors alone. In the following, we present two frequent choices of detectors before briefly looking at some of the local feature descriptors in use today.

1. Harris-Laplace detector: A popular choice of interest point detector is a scale-invariant version of the Harris corner detector, introduced as the Harris-Laplacian in [30]. Given a window that is shifted over the image, the Harris detector measures the *corner response* based on the approximate intensity change as the current window

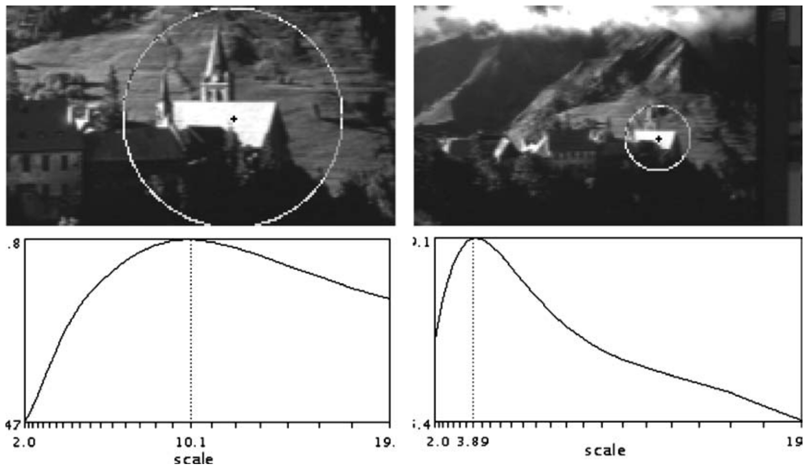
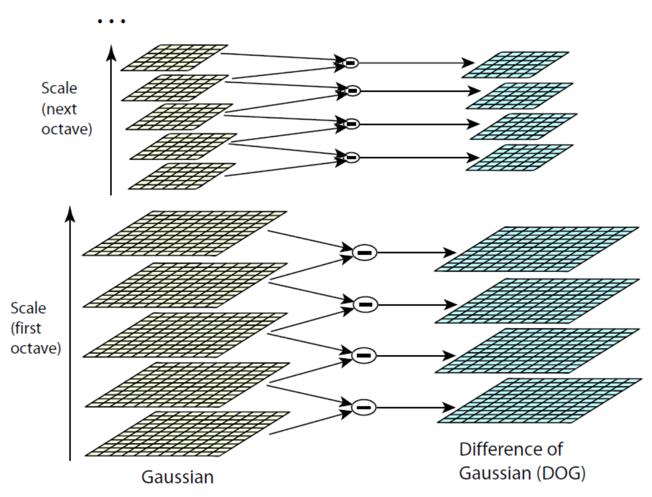


Figure 2-2: From [30]. Variations in characteristic scale and corresponding scale functions (normalized Laplacians) for two images at different resolutions.

position is perturbed. For uniform patches, one expects the windowed intensity to remain constant after pertubation in any direction. For edges, on the other hand, pertubation orthogonal to the edge direction causes an intensity change inside the window while for corners, any direction suffices to induce an intensity change. The Harris detector introduces a response measure invariant to rotation and (uniform) illumination changes that assumes its maximum for corners in the image window.

For scale invariance, the Harris detector is run over the input image at multiple resolutions. At each corner point $\mathbf{x}$, a unique scale factor (also referred to as *characteristic scale*) s is then picked by searching for an extremum of a scale function $F(\mathbf{x}, s_n)$ centered at that point. A good scale function avoids plateaus and multiple extrema to allow reliable estimation of a single, fixed scale s (cf. Figure 2-2).

2. Difference of Gaussian detector: The DoG detector is another popular choice and is used as the interest point detector for the SIFT descriptor [29]. It shares its rotation and scale invariance with the Harris-Laplace method but operates slightly differently: initially, the image is stored in a difference of Gaussian pyramid which can be efficiently computed by subtracting adjacent images in a regular Gaussian



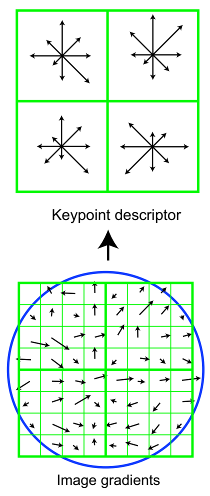


Figure 2-3: From [29]. Difference of Gaussian pyramid from blurred and resized source images (left) and obtaining a SIFT descriptor from a local set of gradients (right).

image pyramid (cf. left side of Figure 2-3). Unlike before, search for interest points now proceeds in this entire 3D scale-space, i.e. over $(x, y, scale)$ at the same time. Detecting maxima in this pyramid is efficiently implementable and leads to distinctive keypoints that compare in their repeatability-rating to those of the Harris-Laplacian (see [44] for a comparison).

3. Local descriptors: After interest points have been found at their respective scale, the corresponding regions must be described in a way that is suitable for matching, i.e. by mapping them into a vector space. There exists a vast number of approaches to creating these local *feature vectors*, including using the grayscale patch around the interest point directly, computing filter responses, or describing the characteristic texture or structure around the point. In the last group we can count the popular SIFT [29] and shape context [5] descriptors, the former being a texture-based and the latter a shape-based description around the local point.

The SIFT descriptor has gained popularity because of its empirically-shown excellent performance for recognizing textured objects in natural scenes, even at real-time speeds. Like all modern descriptors, it makes use of the characteristic scale determined by the detector to enable scale invariant matching. For each interest point, the dominant gradient direction in the direct neighborhood is initially determined. Then, for sixteen $4x4$ pixel patches around each interest point, the image gradients are determined relative to that dominant gradient direction enabling rotational invariance. All gradients are then compressed into sixteen 8-bin histograms, resulting in a $16 \times 8 = 128$ dimensional feature vector for each interest point. A condensed version of this binning operation is shown on the right side of Figure 2-3.

A full-fledged 3D recognition system based on SIFT features is outlined in [28]. Correspondence between model and image features is established with an approximate nearest neighbor algorithm using the Euclidean distance. Similar to the geometric pose clustering method before, each such correspondence then casts a vote for the respective view and all poses consistent with the match in a Hough-like accumulator. For verification, only those views and poses with a large enough number of votes are considered further.

Other recognition methods: A more recent attempt to combine some of the older research into invariants (local appearance) with spatial layout information (geometry) is presented in [40]. The fact that 3D objects can be approximated by planar patches *locally* is used to compute a number of patches, their invariants and their 3D spatial layout. This layout information yields an additional constraint when matching invariants of model and observed object. Unlike current appearance-based approaches that tend to store a large number of views for each object, the described method directly builds a 3D model from the local patches and uses that for recognition.

2.1.3 Comments

We have seen that recent research has shifted from attempting to extract high-level features (such as lines) in images to operating on a large number of localized, low-level features. These low-level features rely predominantly on object appearance, such as the corner responses in the textural pattern of a specific object. Reliability against occlusion is achieved through the sheer number of repeatable features extracted from the images and geometrical information is frequently discarded entirely. Despite this, modern recognition methods have demonstrated greater resistance against changes in illumination and clutter than earlier geometry-based methods ever achieved.

Many of these systems acknowledge that object segmentation based solely on low-level cues is an unsolvable problem and do not attempt segmentation of the object from the background at all. They work under the silent assumption that either the interest detector does not fire on the background or that the sheer number of object features will overwhelm those on the background. This problem gets larger for object category recognition where models are trained across multiple instances and is picked up again in the next section. The reliance on purely textural features has further disadvantages, specifically with respect to its generalization ability to unseen objects. Shape-based methods generally allow for larger variability and are a continued focus of research (see, e.g., [6]). With the latest appearance-based methods introduced in the previous section, however, the real-time recognition of textured instances can be considered a solved problem.

2.2 Object category recognition

This section gives an overview of the sparser literature about the admittedly more difficult problem of object category recognition. For the recognition systems of our time, categories are taken to encompass objects of similar *layout* or *appearance*, but *not* based on similar function, an even harder problem. Over the last years category recognition has seen a surge of interest because of statistical approaches entering the field that allow to make the notion of intra-class variation more rigorous.

2.2.1 Historical approaches

The previously mentioned geometric approaches for instance recognition (see section 2.1.1) limit intra-class variability to changes in texture as they are based on static contours. Systems that relax some of the strict geometric constraints are usually part-based recognition systems.

Part-based methods: Methods that fall into this group separate part detection from the spatial layout of the parts and allow some flexibility with respect to part appearance or the geometry. A well known early system of this kind is Fischler and Elschlager's *template and springs* model [20] which has been applied to face recognition with some success. Here, individual part similarity as well as the spring deformation required to map the model onto the image are combined in a joint cost function that is optimized.

Another famous example is Brooks' ACRONYM system that combines a generalized-cylinder-based part representation with flexible constraints on parts and geometry that can be specified by the user [12]. Customizable parameters include shape and size as well as relative pose of the parts. The resulting object models are inserted into a hierarchy, with less constrained objects at the top and specific object instances at the bottom. Given an object to search for in an image, the system first determines candidate part locations from an edge-extracted version of the image and then searches for part arrangements that fulfill the specific geometric constraints.

As we will see later, the idea of part-based representations for object recognition is as valid today as it was in this early era. A natural application is for example the detection of humans where valid body configurations are encoded as constraints on the different parts of the human body (see, e.g., [24]).

Appearance-based methods: Subspace methods such as principal component analysis (PCA) have been used successfully to capture the global appearance of a set of objects. Referring back to Figure 2-1, these methods attempt to approximate the appearance manifold by detecting commonalities in a large set of training examples. Turk and Pentland use PCA to map cropped frontal face images into a lower-dimensional subspace that captures most sources of variation in human faces (*eigenimages*). Recognition can be performed efficiently in this space using Euclidean distance [48].

A 3D view-based recognition system based on the same approach is due to Murase and Nayar [32]. Each object view (global appearance) is projected to a single point in a lower-dimensional *eigenspace*. By coarsely sampling the view sphere, all views combined trace out a continuous curve in the eigenspace, yielding a lower-dimensional approximation of the object's entire appearance manifold. Recognition can be performed quickly by projecting a novel image into the eigenspace and determining the closest manifold point. This strategy yields both object type and pose as both are implicitly encoded by the curve in the eigenspace.

It is noteworthy that since these early attempts at approximating the appearance manifold, more sophisticated approaches have been suggested. Both locally linear embedding (LLE) and ISOMAP are two nonlinear dimensionality reduction methods that attempt to preserve neighborhood relations of the possibly highly nonlinear appearance surface in the lower-dimensional structure. LLE is applied to face images of varying pose and expression in [43]. In the experiment, given a large amount of images ($N = 1965$) of low resolution (20×28 pixels) depicting a human head at various poses and facial expressions, a lower-dimensional manifold is learned that preserves the characteristic two modes of variation (pose and expression) of the original data. Its usefulness is demonstrated for visualization and animation (e.g., by tracing out a path on the lower dimensional manifold that represents facial expression changes at a fixed pose) but not for face recognition.

2.2.2 Current approaches

Category-level recognition has seen a similar trend away from geometric methods to appearance- or part-based models. To go beyond single instances, models usually combine invariant local features with methods from statistical pattern recognition to learn the distribution of features in an object class. Current research in this field is view-based and focuses on detecting class membership for objects in natural scenes under constrained viewpoints.

Bag-of-features approach: A number of approaches discard structural information entirely and focus on detecting class membership based on texture information alone. This is similar to the bag-of-words model in text categorization where only word presence or absence is used to make inferences about the content. Initially, an interest point detector and a descriptor pair are run to extract a set of regions from all images. In a second step, a *visual vocabulary* is defined by forming clusters of similar regions which are referred to as *visual words* [15]. The signature of an image is then given by the histogram over visual words, i.e., by how many instances of a particular pattern occur in the image. A binary classifier can now easily be trained for every object category that predicts class membership for a given image.

In addition to just modeling the distributions of descriptors over object classes, more complex models add a layer of spatial constraints that are learned from the data and exploited during recognition: knowing that the wheels of a car have to appear in certain numbers and at specific relative positions, for example, certainly improves on

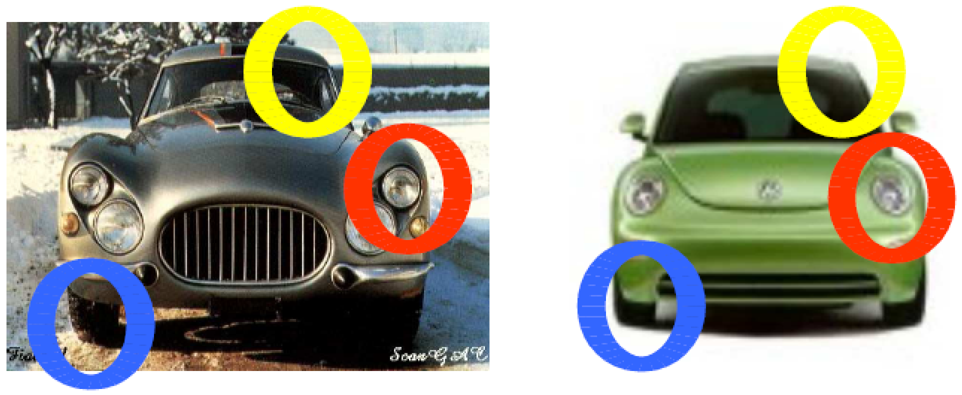


Figure 2-4: From [18]. Shown are a number of characteristic parts of the "*car front*" class in their appropriate spatial configuration.

just relying on the fact that wheels appear somewhere in the image.

Incorporating spatial information leads to the popular *parts and structure models* that can be thought of as a modern probabilistic formulations of Fischler and Elschlager's template and springs model (cf. section 2.2.1). Figure 2-4 shows some of the typical images that current category-level recognition systems are able to classify.

Modern parts and structure models: In this category we include a number of semi-supervised recognition techniques that have appeared over the recent years. The models presented here require no preprocessing – such as segmentation – but instead only receive a set of images, labeled as either containing a specific class or as background. These models can be distinguished into three classes based on how they define parts and structure and how they integrate both during learning [54].

1. Models that emphasize parts: For these models, part learning proceeds as in the earlier bag-of-features approach, i.e., by clustering a number of intensity patches that have been extracted from the positive training examples around a set of interest points. Similarly, part matching is executed without reference to a particular structure model and only after parts have been recognized is a structure term exploited.

Leibe and Schiele's *implicit shape model* [27] uses normalized cross-correlation as a distance metric during part clustering as well as for part recognition. During training, each part also keeps track of the relative position of the object *centroid* to itself which summarizes the structure of the object class. During recognition, parts vote for the centroid in a Hough-like manner and locations with enough votes are classified accordingly.

Agarwal and Roth extend the bag-of-features approach described previously by adding geometric relations to the feature vectors [1]. For each visual word, a binary entry denotes presence or absence, as before. In addition, the relationship between any two parts – discretized into one out of twenty buckets, each denoting a particular angle and distance – is appended to the original vector. As before, a discriminative classifier is trained in this high dimensional space allowing predictions whether a certain object class is present in a given image window or not.

2. Models that emphasize structure: Recognition systems in this category rely on a structural model that defines the spatial relationships between the detected parts. In [17], Felzenszwalb and Huttenlocher describe *pictorial structures* which are closely related to the original template and springs model. Unlike the earlier work, part relationships are now learned directly by observing relative positions and angles of parts in the training data. The final structure model is a probability distribution over configurations of this "skeleton". Given a novel image to classify, part detection is followed by structure fitting that attempts to join the discovered parts in a configuration of high probability. The authors derive a cost function that consists of a deformation cost (based directly on the probability assigned to the configuration under the structure model) and an appearance cost and show that this joint cost term can be minimized efficiently if the structural model is to a tree.

A natural application of this model is recognizing humans and their body pose in an image and is demonstrated in the original paper [17].

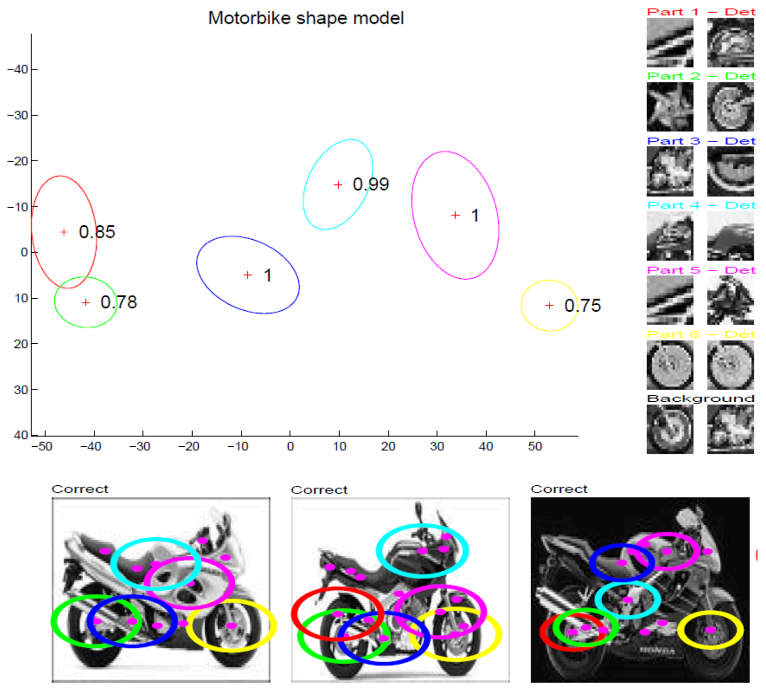


Figure 2-5: From [19]. Shown are the learned relative positions of motorbike parts together with the distribution over part appearance (*main diagram*), some typical parts from each category (*right*), and the model fitted to three other instances from the same motorbike class (*bottom*).

3. Models that learn parts and structure simultaneously: The most sophisticated of the approaches presented in this review is Fergus et al's extension of Weber et al's *constellation model* [19, 50]. The authors present a generative model for object categories that encompasses "shape" (relative position of parts) and part appearance with the additional advantage of being invariant to object scale. The model uses Gaussian distributions to model both shape and appearance and parameters for these distributions are learned simultaneously via the EM algorithm.

For each category, the model is trained with a large number of images (around 400 in [19]). As usual, an interest point detector is invoked initially to detect regions and their characteristic scales. At each interest point, the surrounding region is mapped into a lower-dimensional PCA space to make parameter estimation of the model tractable. During the EM iterations, a single Gaussian with full covariance learns the relative positions of all parts, while simultaneously individual Gaussians take on the appearance of every single part. The output of this operation can be seen in the coordinate frame shown in Figure 2-5. Here, six motorbike parts are shown in their typical spatial configuration and individual covariances show spread in the respective part appearances.

At run-time, matching new images is done by calculating the likelihood ratio of the object belonging to any of the category models versus a separately trained background model.

2.2.3 Comments

The modern techniques presented in this section utilize methods from statistical pattern recognition to create object models that span a variety of instances. While they are suited for recognizing textured object classes even in cluttered backgrounds, there is a number of basic issues that remain for these models. First, some of the models are only able to determine the presence or absence of an object in the image based on the statistics of the extracted descriptors. Localization, on the other hand, reverts to sliding a window across the image and running the algorithm in the specific region. Furthermore, basic invariants such as to rotation and scale are frequently only dealt with at the part level (if at all) but not at the structure level. Leibe and Schiele's method in [27], for example, records the relative position to the centroid for each part but does not account for different orientations of the parts or the overall object. Generalizing this to arbitrarily rotated or scaled objects would in turn require exhaustive rotation and scaling of the input image. Along the same lines, most modern category recognition systems fix the viewpoint of the observer and make clear distinctions between, for example, the "front-of-car" and "side-of-car" categories.

The statistics collected over the local appearance descriptors suffers from similar problems as noted for the instance recognition case earlier. Because there is no explicit segmentation, there exists an inherent assumption that the detectors do not fire on the background or that collecting statistics from a large number of instances will sort out the bad features from the good ones. That this assumption is not always a valid one and can lead to interesting side effects has been observed by many authors (see, e.g., [36]). Here, background rather than object statistics (such as *road* features instead of car- and motorbike features) have been picked out as being most characteristic for the respective classes.

The older, global appearance based methods (such as PCA for faces) suffer from many of the problems that the newer system circumvent. For example, changes in lighting, cluttered backgrounds, occlusions or pose variations may all affect the lower-dimensional projection of the image. Because of that, there is frequently a pre-processing step executed that includes segmentation or cropping of the input image. The sliding window idea is equally valid if a single object has to be picked out from a larger frame.

Lastly, both training and run-time performance of the more modern local appearance based models are far from real-time. For Fergus et al's extension of the constellation model, for example, the authors mention training times of 24-36 hours for 400 images and recognition times of 2-3 seconds per object view [19].

2.3 Summary – Where do we fit in?

This chapter presented an overview of the vast and diverse literature on object recognition as it has unfolded over the past 40 years. We have established that the current research focus is on detecting object *categories* from single, fixed viewpoints in complex natural scenes. For object *instance* recognition, reliable and fast appearance-based methods are available that perform both localization and recognition in similarly cluttered scenes.

This bears the question how our research fits into the larger picture and what unique requirements hold for the recognition system that we develop in the next chapter. A major factor for us is to recognize objects reliably and *quickly* under varying viewpoints, driven by the motivation to have the system run on a physical robot. Many of the sophisticated category-level recognition systems are ruled out for exactly these two reasons. Additionally, our requirements include being able to recognize object classes without characteristic texture patterns, such as sets of uniformly-colored blocks or other simple shapes, in addition to more sophisticated object types.

Of key importance for us is also to determine object location rather than merely telling presence apart from object absence. This is again directly due to our system being run on a physical robot that may choose to associate a tracker with specific objects as soon as the category has been determined. For later grasping attempts on the objects, it is furthermore useful to be able to obtain the orientation of the object in the field of view of the robot.

There are also some unique factors about our system that allow us to go beyond single snapshot-based object recognition. First, we are employing an active vision head and have proprioceptive insight into camera position and orientation. We are also operating under relaxed background constraints which makes segmentation of objects from the background feasible. These features, together with the requirements above, define our research context and give the rationale for the recognition system that we introduce next.

Chapter 3

Our Implementation

In the previous chapter we concluded with some particularities that hold for our active vision platform. We also established a number of requirements for our system, among them:

- The ability to recognize object categories across *viewpoint variations*
- The ability to perform recognition of objects without characteristic texture (for example, uniformly colored objects)
- The operation under time constraints, which are generally based on the idea of having the recognition component run on the robot in (or close to) real-time.

Our particular requirements also allowed a relaxation with respect to camera control and the background. In general, we assume have control of the camera in order to be able to follow trajectories and to sample at a number of points on the viewsphere. Additionally, the fact that our objects generally appear on the table top in the field of view of the robot also allows us to relax the requirement for our system to operate under complex or natural backgrounds.

Two core ideas guide the implementation of our recognition system. First, based on our literature review in the previous chapter, we select a recognition approach that builds on invariant local features. As detailed earlier, these systems generally display higher robustness to occlusions and can be made invariant to different transformations. Many of the recently proposed local features are based on the uniqueness of appearance or texture extracted around a set of interest points. This is contrary to our requirement of detecting objects with surfaces of uniform color or little textural detail. We therefore base our classification decisions on the *shape* of an object and select *shape context* [5] as our local feature descriptor.

As demonstrated in Figure 1-2 in the introduction of this thesis, basing classification decisions off a single view of an object does not, in general, disambiguate the object classes reliably from each other. We therefore accumulate evidence over a number of images but also attempt to reduce the time required to commit to a classification decision. The second idea that we pursue here therefore addresses the question *what we can learn from a series of images (that stem from an "ordered sweep" over the object) without reverting to the full 3D structure of that object.* Concretely, we look at how low-level features extracted from an object evolve over time as the active vision camera follows a set of known trajectories. To the best of our knowledge, combining this *inter-frame* information with local feature-based matching for object recognition has not been suggested before in the literature.

In our terminology for the remainder of this thesis we use the word *sweep* to refer to the process of taking images of an object at specified, fixed intervals while traversing a given trajectory. We also refer to a *motion catalog* to denote a fixed set of trajectories along which the camera can be moved. Before going into more detail about how feature motion information is recorded and matched, we briefly review the shape context descriptor selected for our implementation. It is worthwhile to point out that the idea of utilizing feature dynamics as an additional source of information is quite general and is in principle compatible with other descriptors that allow to establish correspondence between successive frames.

3.1 Local features: Shape Context

Shape context is a robust local feature descriptor introduced by Belongie et al. in [6]. It is utilized to find a similarity measure between a model and a (potentially deformed) target shape. Calculation of this measure requires to establish *correspondence* between both shapes and to assess the warping that the model points undergo to arrive at the target shape.

1. Solving correspondence between model and target shape:

The shape context algorithm represents shapes by their contour as it is obtained from a standard edge extraction procedure. From all points on the contour, the algorithm initially samples a finite subset of points $P = \{p_1, \ldots, p_n\}, p_i \in R^2$ that it assumes to be sufficient to characterize the shape structure. Note that sampling can be as simple as a random selection of contour points and that no interest point detector is run on the shape. The goal of this first part of the algorithm is then to find, for every point p_i on the model shape, the most similar point q_i on the target shape.

For reliable matching, one associates with each point p_i a shape context descriptor that encodes the global layout of the shape relative to that point. As shown on the left side of Figure 3-1, this descriptor corresponds to a log-polar histogram centered at that point. In the paper, there are 60 bins in each histogram with angular displacements discretized into 12 increments and radial displacements into 5. Shape context histograms for three points on two deformed A characters is shown on the right side of the same Figure. As is evident, the algorithm assumes that corresponding points

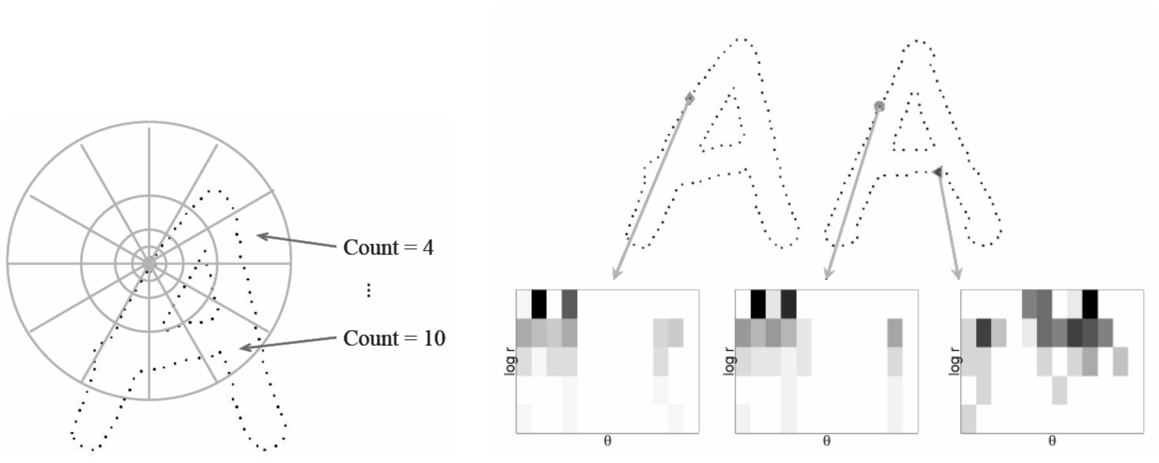


Figure 3-1: From [6]. Creating the log-polar histogram (the *shape context*) at a certain point on the shape (*left*) and comparing histograms at two similar and one unrelated location (*right*).

share similar shape context histograms whereas no such relation holds for unrelated points.

Each shape context h_i can be interpreted as a distribution over relative point positions with $h_i(k)$ denoting the number of points relative to p_i that fall into bin k. To match two such distributions, the authors therefore suggest the χ^2 test

$$C_{ij} = C(p_i, q_j) = \frac{1}{2} \sum_{k=1}^{K} \frac{[h_i(k) - h_j(k)]^2}{h_i(k) + h_j(k)} \tag{3.1}$$

which effectively tests the hypothesis that the observed frequency h_i follows the theoretical frequency distribution h_j. Given cost matrix C with entries C_{ij} for all i, j, solving the entire correspondence problem is equivalent to minimizing the *total cost*

$$H(\pi) = \sum_i C(p_i, q_{\pi(i)}) \tag{3.2}$$

where π is a permutation that assigns each i to a unique j. As outlined by the authors, standard solution methods such as the Hungarian method can be used to solve this type of assignment problem.

2. Modeling Transformation between Shapes:

Given the set of correspondences $\pi^* = \arg\min_\pi H(\pi)$, the algorithm next attempts to estimate the transform $T : R^2 \rightarrow R^2$ between the shapes. The authors use the thin-plate spline (TPS) model from [11] to interpolate a surface through the point constraints imposed by π^*. More concretely, they define two TPS interpolation functions $f_x(x, y)$ and $f_y(x, y)$, one for the translation of points $p_i = (x_i, y_i)$ in x direction and the other for translation of the same points in y direction. As detailed in [11], a TPS interpolant has the form

$$f(x, y) = a_1 + a_x x + a_y y + \sum_{i=1}^{n} w_i U(|p_i - (x, y)|) \tag{3.3}$$

consisting of an affine interpolation surface (described by a_1, a_x, a_y) together with a linear combination of kernel functions (or *warps* or bumps on that surface) $U(\cdot)$. Parameters a_1, a_x, a_y and the w_i can be solved for so that the overall "bending energy" (or surface height) is minimized. The physical analogy is a minimum deformation metal plate that is bent to go through the same points.

In [6], the authors use a regularized version so that the TPS does not have to fulfill the point constraints exactly but instead approximates the deformation.

3. Shape Distance Computation:

A joint distance term between two shapes P and Q can be derived as a weighted sum of the *shape context distances* and the *bending energy* of the approximated TPS $T(x, y) = (f_x(x, y), f_y(x, y))$. The former of the two terms is merely the sum of the individual matching costs of the points on P with the corresponding points on Q (cf. equation 3.1). The rationale for including the second term is that a higher bending energy reduces the expected similarity between both shapes.

The authors outline that additional terms can appear in this joint distance formulation, such as similarity scores for grayscale patches extracted around corresponding points on P and Q.

The shape context algorithm has been shown empirically to perform well in scenes with low background clutter. As described earlier, this restriction is acceptable to us since object recognition is carried out on the table top in front of Trisk.

3.1.1 Modifications

By design, the algorithm is invariant to translation and scale if we normalize all distance computations in the binning operations with the median distance over the whole shape. For our implementation, we make the following modifications to the plain algorithm described above:

Rotational invariance: To make the algorithm more robust to mildly affine transforms, we add rotational invariance as hinted at in [6]. In order to so, we rotate every shape context histogram from the global coordinate system shown on the left of Figure 3-1 to a local coordinate system specific to that point. We simply add all displacements (Δ_x, Δ_y) between the current point and all other points on the shape in both x and y directions and then calculate the angle $\alpha = \tan^{-1}(\frac{\Delta_y}{\Delta_x})$. This is just the unique angle in a right triangle where the opposite leg is Δ_y and the adjacent leg is Δ_x. We subtract α from all binning operations to make the shape context descriptor at that point invariant to rotation.

Uniform sampling: During the initial sampling of the object contour, we attempt to select points in a more uniform way than mere random sampling would produce. Given a set of desired points of size N, we initially obtain a dense random sampling along the contour with kN samples where k is a positive integer (3 in our experiments).

We then compute the $kN \times kN$ matrix D with each entry D_{ij} denoting the Euclidean distance between sample points p_i and p_j. Finally, points from that set of kN samples are dropped until we arrive at the desired number N. At each iteration, the point associated with the smallest inter-point distance is removed resulting in a greedy algorithm that attempts to enlarge spacing between sampled points.

3.2 Feature Dynamics

We established in the beginning of this chapter that our active vision system allows us to move the camera along a set of fixed trajectories collected in a *motion catalog*. Since we are in control of the camera, we are free to select velocities and the image sampling frequency on the trajectories. The crucial observation for the extension we present here is that images stem from an *ordered sweep* over an object rather than being sampled independently from some process. We propose that additional information is revealed about the object by its characteristic variation over time. For active vision systems of the kind considered here, the notion of local features does not only make sense spatially from image to image but also extends *across time*.

Feature change across consecutive images depends on the ego motion of the camera and on the characteristic structure of the object (assuming that the object is stationary during the observation cycle). This is demonstrated in Figure 3-2 for two different trajectories over the identical object. For both examples, the middle picture shows the recorded change between the left and the right frames in the sequence. For the upper trajectory, feature change clearly reveals the cup's handle rotating into place while other parts of the object remain relatively stationary. Similarly, the lower trajectory allows us to deduce that no structural change is expected for two views of the same object when seen from a different camera trajectory. It is this structural information that we hope to exploit during the matching process of novel objects.

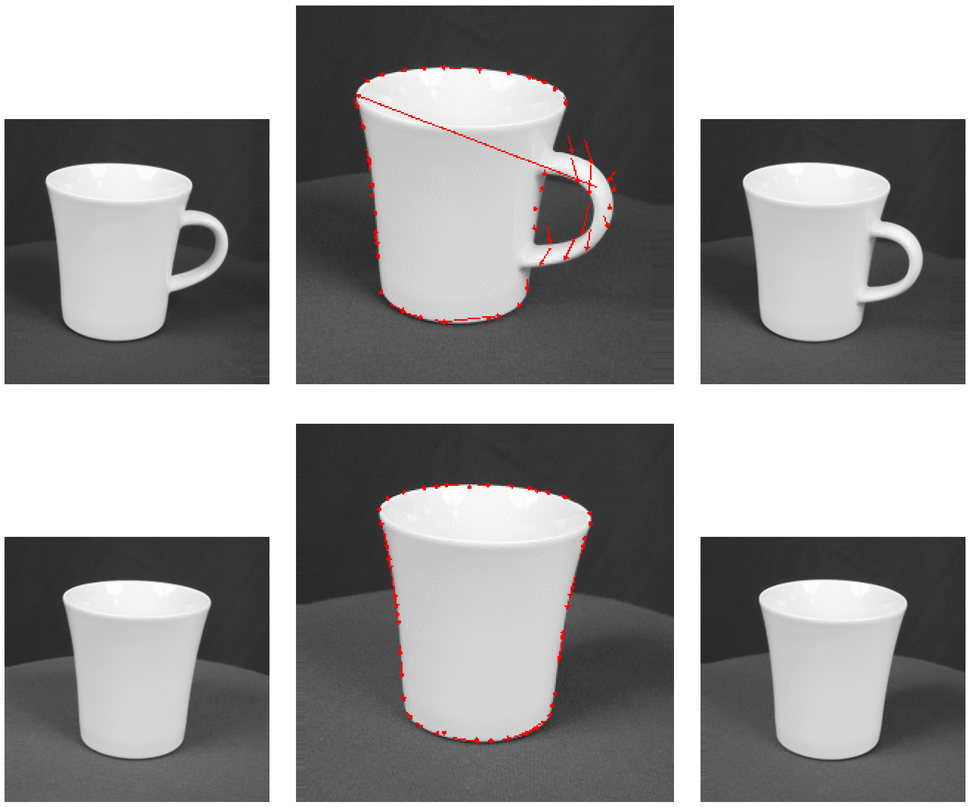


Figure 3-2: Two frames (*left* and *right*) from two distinct trajectories over the same object. Displayed as well is the contour motion information that links both frames (*center*).

3.2.1 Extraction and Encoding of Feature Motion

Given an ordered set of images $I_1, I_2, \ldots, I_k$ from a trajectory, we compute feature motion for all pairs of successive frames, i.e., $(I_1, I_2), (I_2, I_3), \ldots, (I_{k-1}, I_k)$. The approach that we pursue here to compute feature motion relies on the local feature descriptors to establish *correspondence* between both frames in each such pair. We showed in section 3.1 how a correspondence π^* is derived for the shape context descriptor.

With every image pair (I_i, I_{i+1}) we then associate two vectors $\mathbf{v}_{i+1}$ and $\mathbf{d}_{i+1}$ of size N (the fixed number of points sampled from both image contours). Entries in these vectors respectively denote the angles and magnitudes of the displacement vectors between corresponding points in I_i and I_{i+1}:

$$\mathbf{v}_{i+1} = \begin{pmatrix} \angle(p_1 - \pi^*(p_1)) \\ \angle(p_2 - \pi^*(p_2)) \\ \vdots \end{pmatrix}, \mathbf{d}_{i+1} = \begin{pmatrix} ||p_1 - \pi^*(p_1)|| \\ ||p_2 - \pi^*(p_2)|| \\ \vdots \end{pmatrix} \tag{3.4}$$

where $\pi^*(p_i)$ denotes the point corresponding to p_i in the other image. Similar to the case for distances in the shape context descriptor, we can normalize the magnitudes in $\mathbf{d}_{i+1}$ with the median magnitude. Note as well that if both images are of different dimensions, we place the smaller one at the center of the larger one before calculating both vectors.

3.2.2 Matching Feature Motion

For purposes of this thesis we devise a cost function to compare the feature motion obtained from two trajectories at the same point in time t. Given two such motion patterns $(\mathbf{v}_t^{(1)}, \mathbf{d}_t^{(1)})$ and $(\mathbf{v}_t^{(2)}, \mathbf{d}_t^{(2)})$, we incorporate *cosine similarity* and *magnitude difference* for each of the N entries into a joint cost term.

Cosine similarity is just the cosine of the angle between two corresponding entries, i.e., $\cos(v_{t,i}^{(2)} - v_{t,i}^{(1)})$ for all $i = 1, \ldots, N$ and is bounded in $[-1, 1]$. Naturally, it assumes its maximum for the case that the angle between both vanishes. To assess feature motion similarity we additionally compare the difference in displacement vector lengths $|d_{t,i}^{(2)} - d_{t,i}^{(1)}|$ which we normalize to fall into the same range $[-1, 1]$ (the maximum is assumed if both share the same length). If we denote the latter term by Δ_i, we can obtain a joint similarity score as the weighted sum

$$s_i = \cos(v_{t,i}^{(2)} - v_{t,i}^{(1)}) + w_i \Delta_i \qquad \forall i = 1, \ldots, N \tag{3.5}$$

and a total similarity between both motion patterns as

$$S = \sum_{i=1}^{N} s_i \tag{3.6}$$

Referring back to Figure 3-2, the rationale is that we expect similar objects to result in similar contour motion, which is determined by both direction and magnitude of the individual displacement vectors.

In our implementation we use a simple heuristic to select the weight coefficients w_i. In general, we want to avoid that two displacement vectors of similar lengths but in different directions result in high similarity scores s_i. Accordingly, we discount the Δ_i score based on the size of the angle between both displacement vectors. This measure has demonstrated good performance in our empirical tests in chapter 5.

3.2.3 Discussion of Alternative Approaches

It is conceivable that we could have exploited feature motion and correspondence between frames in other ways for object recognition. One approach that comes to mind are *structure from motion* algorithms that estimate a 3D model of the object when observed across multiple images (see, e.g., [25]). Here, 3D points are recovered and are used to estimate planar surfaces on the object. Alternatively, known 3D models are fit or articulated based on the 3D points recovered through multiple frames.

As detailed previously, of our interest here instead is the combination of some of the recent approaches in view-based object recognition (such as invariant local features) with motion information. We show that our coarse 2D approximation of 3D motion suffices to increase recognition performance without having to resort to 3D structure of an object.

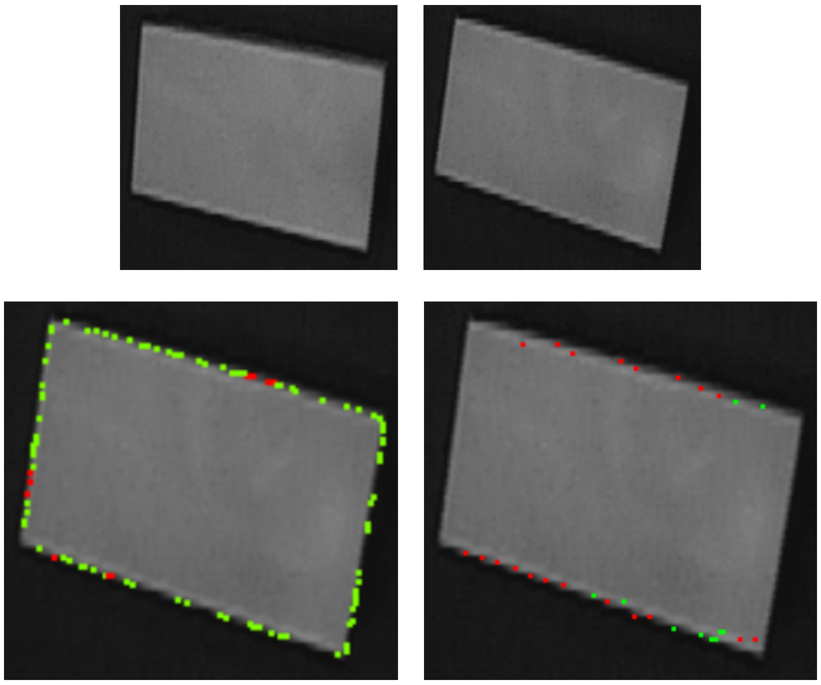


Figure 3-3: Two typical object views from the robot camera (*top*). Shown in green are correspondences found with shape context (*bottom left*) and KLT (*bottom right*). Red pixels denote features that could not be matched up with any feature from the previous frame.

Our approach of recording feature motion can be seen as an instance of *optical flow* in that it approximates the true 3D motion field with 2D motion information albeit only at a select number of points. Traditional optical flow algorithms are appearance-based and establish correspondence for all pixels by matching up image intensities. As detailed in [21], traditional motion fields can fail for objects of uniform intensity.

There are also established methods in the *tracking* literature for estimating motion between successive frames. An example is the Kanade-Lucas-Tomasi (KLT) tracker that comes with its own selection method for features and tracks them from frame to frame assuming a simple, translational model of image motion [45]. KLT selects features based on a corner response measure which may not be appropriate for objects without a distinctive texture. In Figure 3-3 we compare the ability of KLT and our implementation of shape context to maintain features between two successive frames. In both cases, a maximum of 50 features was extracted from the image.

It is noteworthy that there have been approaches to object recognition based on optical flow *alone*. Perhaps closest in spirit to our idea of exploiting object motion for recognition is the work of Arbel and Ferrie in [3]. Here, the authors collect magnitudes of the optical flow field as a camera is moved along the view sphere over an object. The underlying assumption is that changes in velocity in the field reveal *depth* (or structural) information about the object. Moving along 184 trajectories, the authors store a set of "flow images" that record the magnitude of the flow at each pixel. These images are then projected into a 20-dimensional eigenspace where different objects' flow information is represented by distinct Gaussians. The authors show that it is possible to perform recognition in this space based on projecting novel flow images into the eigenspace. The described approach is similar to the global appearance based methods outlined previously. Our implementation, on the other hand, joins local feature-based matching with flow information (both direction and magnitude) collected at the same feature points. Different from global appearance, local features have shown invariance to occlusions which allows us to use as much of the motion information as possible.

Lastly, there also exists a large amount of literature on classifying motion (motion recognition) or using motion for recognition of higher-level behaviors (motion-based recognition). Examples are as diverse as action recognition in video [16], sign language interpretation [46], or general gesture recognition. A survey of approaches in this field can for example be found in [13]. It has to be noted that the focus of this work is different, however. We mentioned in the beginning of this section that observed change from frame to frame is due to ego motion of the camera and the characteristic structure of the object. Whereas we hope to detect and exploit *structural change* over time (such as the handle of a cup rotating into view) for object recognition, the work above focuses strictly on classifying *motion*. Since in our setup the object is fixed while the camera is moving, this would be analogous to making statements about the camera motion rather than the object itself.

3.2.4 Current Limitations

There are two limitations in our current implementation. First, we rely on the object being static, i.e., not undergoing self-motion. This is by design as we interpret feature motion as structural change over time. The same system, however, could be reversed with the object undergoing motion and the camera being fixed. In our robotics domain, this could be equivalent to having the robot move the object along characteristic trajectories while observing from a fixed viewpoint.

Second, like some of the newer category-level recognition systems presented in section 2.2.2, the classifier we construct around feature motion is generally not invariant to in-plane object rotation. However, there are at least two ways to introduce robustness with respect to rotation:

Training across multiple orientations: In this approach, feature motion is recorded multiple times for each trajectory in the motion catalog. Each recording is executed with the object rotated at a different angle. For our implementation, we follow this approach and use 30 degree angle increments as we sweep the object multiple times during training.

On-the-fly trajectory adjustment: If the active vision system supports motion along the entire view sphere, it is feasible to adjust the trajectories in the motion catalog to maintain the same relative camera-object orientation used during training. This assumes that the object rotation can be estimated in the first place: for the shape context algorithm that we implement, the orientation of the local shape context histograms reveals this information (cf. section 3.1.1).

3.3 Joining Both in a Probabilistic Classifier

In the previous sections we have demonstrated our method of obtaining two distinct cost terms, the first one being based on shape similarity and the second based on feature motion similarity. In this section, we attempt to join both terms in a classifier that predicts class membership for the observed object. One requirement for our classifier is that it supports *sequential estimation* of the class membership probabilities. As a new view of the shape or feature motion is obtained, the probability distribution over classes should be revised to represent the current state of knowledge.

For our implementation we pursue standard Bayesian techniques to keep track of the current posterior distribution over object classes. For every new view, after shape matching and feature motion costs have been obtained, we execute the following three steps to update our class membership estimate:

1. Converting costs into probabilities:

To transform cost terms into a set of calibrated probabilities, we postulate that the class associated with the smallest cost is the most probable one and that any other class with a k times higher cost is equivalently k times less likely. Under this assumption, we can obtain $P(\mathcal{C}_k|\mathbf{x}_{SC})$ and $P(\mathcal{C}_k|\mathbf{x}_M)$, denoting the probability of a particular object class $\mathcal{C}_k$ given shape matching cost and feature motion cost, respectively, with a simple computation. Having obtained a set of matching costs $\{c_i\}$ for all classes $i = 1 \ldots K$, we fix one of them, say c_j as our reference. For our distribution $P(\mathcal{C}_k|\mathbf{x} = \{c_i\})$ we then simply have:

$$P(\mathcal{C}_k|\mathbf{x} = \{c_i\}) \propto \frac{c_j}{c_k/c_j} = \frac{c_j^2}{c_k} \tag{3.7}$$

It can be easily verified that after normalization this distribution adheres to the calibration condition above.

2. Joining shape matching and feature motion costs: In the previous step we derived two *discriminative models*, $P(\mathcal{C}_k|\mathbf{x}_{SC})$ and $P(\mathcal{C}_k|\mathbf{x}_M)$, that we need to combine into a single distribution over $\mathcal{C}_k$. A simple way to do that is to assume conditional independence between $\mathbf{x}_{SC}$ and $\mathbf{x}_M$ given the class $\mathcal{C}_k$. This means that given knowledge of the class, no further information is conveyed by $\mathbf{x}_{SC}$ about $\mathbf{x}_M$ or vice versa. More formally, this is the *naive Bayes* assumption:

$$P(\mathbf{x}_{SC}, \mathbf{x}_M|\mathcal{C}_k) = P(\mathbf{x}_{SC}|\mathcal{C}_k)P(\mathbf{x}_M|\mathcal{C}_k) \tag{3.8}$$

For our combination of models it follows that ([8]):

$$\begin{aligned} P(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M) &\propto P(\mathbf{x}_{SC}, \mathbf{x}_M|\mathcal{C}_k)P(\mathcal{C}_k) \\ &= P(\mathbf{x}_{SC}|\mathcal{C}_k)P(\mathbf{x}_M|\mathcal{C}_k)P(\mathcal{C}_k) \\ &\propto \frac{P(\mathcal{C}_k|\mathbf{x}_{SC})P(\mathcal{C}_k|\mathbf{x}_M)}{P(\mathcal{C}_k)} \end{aligned} \tag{3.9}$$

yielding a posterior distribution over the class based on results from both feature motion comparison and shape matching.

3. Online updates of the class distribution:

To continuously adapt our estimation of the class distribution as more views of the object are discovered, we evaluate the naive Bayes classifier above at every time step. Let $P_i(\mathcal{C}_k|\mathbf{x}_{SC})$ and $P_i(\mathcal{C}_k|\mathbf{x}_M)$ denote the predictions of the shape context and feature motion models for the ith object view, respectively. Then, at time t, we have that

$$P_t(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M) \propto \frac{\prod_{i=1\ldots t} P_i(\mathcal{C}_k|\mathbf{x}_{SC})P_i(\mathcal{C}_k|\mathbf{x}_M)}{P(C_k)} \tag{3.10}$$

which allows us to aggregate the numerator in an efficient, recursive manner. Throughout, we assume a uniform prior over the object classes, $P(\mathcal{C}_k)$.

3.4 Software Implementation

We now outline briefly the set of software tools that we developed as part of this thesis in the C++ and C# programming languages. We then present a walk-through of the system's training and testing phases and comment on the computational performance of our implementation.

Our toolchain consists of a range of applications running on the Linux and Windows operating systems and interacting via TCP communication links. We can roughly draw lines between robot-centric software that supports vision and trajectory following functionalities, our core shape matching system, and a set of graphical user interfaces allowing the user to train or otherwise interact with the system.

The trajectory following module presents the entry point to the system: it opens a server port and listens for instructions to play back one of its pre-configured trajectories. In its current form, the system has access to three unique head trajectories stored as XML files in the *motion catalog.* A request for head motion is followed by a traversal of the respective trajectory together with regular broadcasts of the visual field of the robot. Objects are segmented from the background and sent as edge-extracted versions to the core shape system for *training* or *class prediction.*

Supporting the user during the training phase is a graphical user interface shown in Figure 3-6. It receives and displays sweeps over the object and can be instructed to extract and save motion and shape context data from the received images. During the testing phase, the user experiences the interface shown in Figure 3-5. Incoming images are subjected to our shape recognition algorithm and a probability distribution over class membership (based on the current frame alone as well as agglomerated over all previous views) is displayed to the user. Both training and testing phase of the system are outlined in more detail in section 3.4.1.

Lastly, there is software to visualize the inner workings of the algorithm and a number

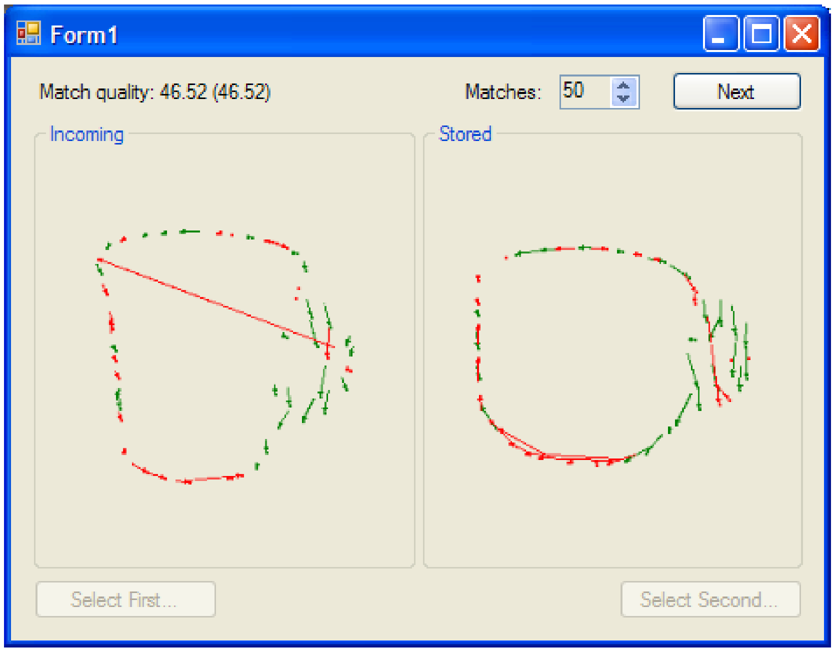
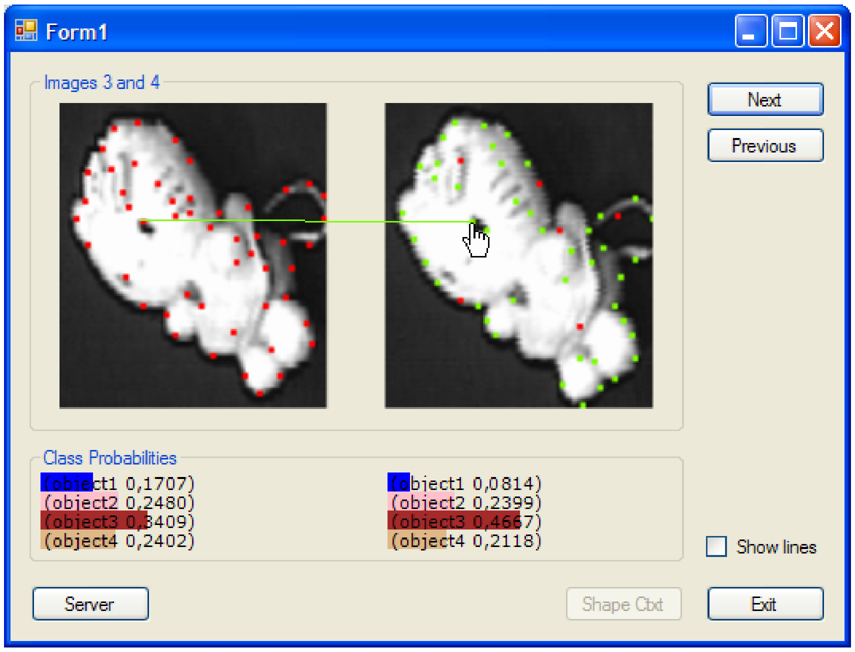


Figure 3-4: Offline visualization and similarity assessment for motion data from two sweeps over different objects from the same *cup* category.



```
(object1 0,1707)        (object1 0,0814)
(object2 0,2480)        (object2 0,2399)
(object3 0,3409)        (object3 0,4667)
(object4 0,2402)        (object4 0,2118)
```

Figure 3-5: Runtime view of the system with matched shape context features highlighted and class membership probabilities indicated below.

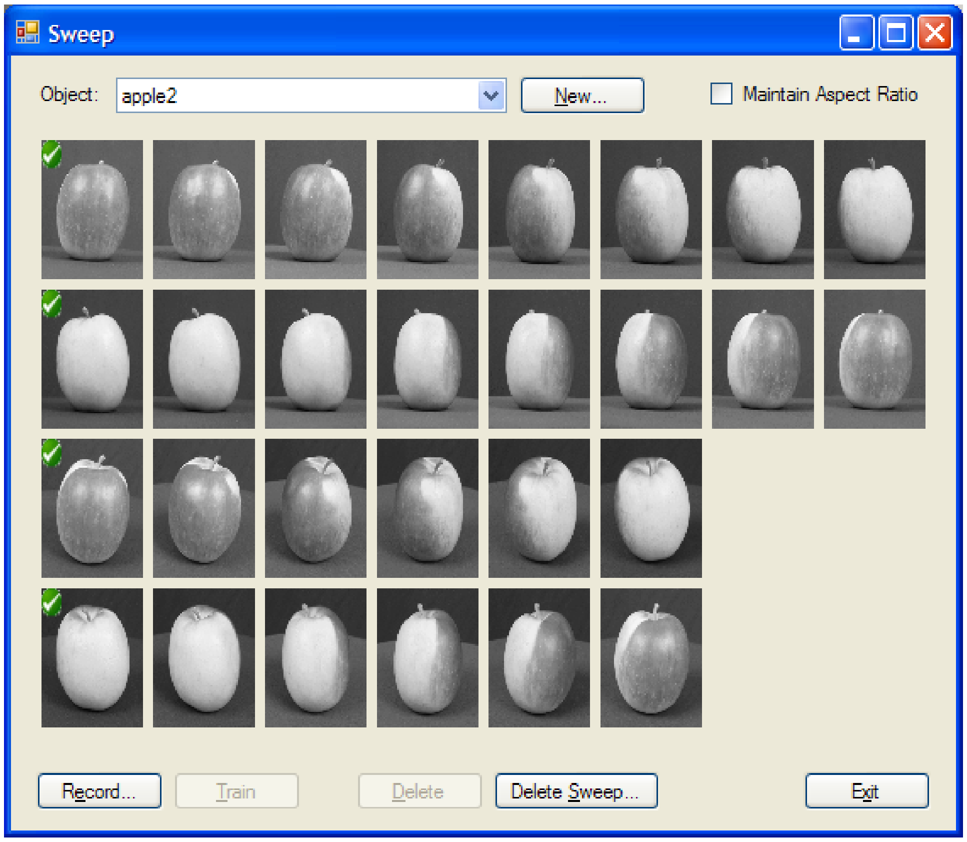


Figure 3-6: Training view of the system showing recorded sweeps of the selected object.

of supportive tools. An example for these is shown in Figure 3-4 where the motion data from two different sweeps is reported and a similarity score reported back to the user. To make testing of the system more effective, we also implemented tools for playing back previously recorded sweeps over an object without having to operate on the physical robot.

3.4.1 Algorithm Outline

Training phase: The training phase consists of receiving images from a sweep over the object and performing two operations on these images. First, for each image

$I_1, I_2, \ldots, I_k$ in the sequence, local shape context features are computed. In our current implementation, we obtain the shape context histograms at 50 distinct points sampled from the edge contour. These histograms are then written into a file on disk that is associated with each individual image I_j. Second, for all $k-1$ pairs of successive images (I_j, I_{j+1}) in the sequence, motion vectors between corresponding features are determined and stored in a second file. The order that these motion vectors is stored in is the same as the order of the shape context features in image I_{i+1}.

Runtime phase: Given a previously unseen image from a novel sweep, I_1^N, followed by $I_2^N, \ldots, I_z^N$, we perform a shape-context based matching between I_i^N and I_i^S for all known sweeps S. This results in distributions $P_i(\mathcal{C}_k|\mathbf{x}_{SC})$ and a permutation π_i^S describing the correspondence between points on I_i^N and all other views I_i^S. For $i = 2$ and following that, we also obtain feature motion information from (I_{i-1}^N, I_i^N). Utilizing the permutation π_i^S, we find corresponding motion information associated with images I_i^N and I_i^S for all sweeps S. This similarly results in distributions $P_i(\mathcal{C}_k|\mathbf{x}_M)$. Based on those two distribution, $P_i(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M)$ experiences Bayesian updates as detailed in section 3.3.

This process continues until either the whole trajectory is completed (i.e., the last image I_z^N is received) or is terminated early if the entropy of $P_i(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M)$ falls below a selected threshold.

3.4.2 Computational Performance

In this section we report real-world speed measurements for our implementation of the shape matching algorithm described above. The numbers reported here are for matching up two shapes from which 100 points have been sampled and include one iteration of setting up the cost matrix, solving the correspondence problem and recovering the TPS parameters. As noted in [6], most parts of the algorithm have between quadratic and cubic scaling behavior in the number of contour sample points. The algorithm scales linearly with the number of instances in the object database, however, so that we report performance values for a single match only.

In the following table, all runtimes have been averaged over ten runs and stem from compiled code with analogous compiler optimization levels.

System	*Matching time*
2 x Intel Xeon 2.80GHz (Linux)	0.038s (100%)
Intel Pentium4 3.06GHz, Hyperthreading (Linux)	0.03s (78.9%)
AMD Athlon64 4000+ (Windows)	0.0198s (52.1%)
IBM PowerPC 3.2GHz (Playstation 3, Linux*)	0.084s (221%)
2 x Intel Pentium3 1GHz (Linux)	0.073s (192%)

Table 3.1: Run-time matching performance of our C++ shape context implementation.

* Note that the code was not distributed across the 6 SPEs of the PS3.

3.5 Conclusion and Possible Extensions

In this chapter we presented a recognition system that joins classification results from shape-based local features with those from feature motion in adjacent object views. The driving assumption has been that the way features translate from frame to frame reveals an additional layer of structural information about the object that can be exploited for recognition. We presented an online recognition system that utilizes a set of fixed camera trajectories to record and match characteristic local shape context features and their evolution through time (an object *sweep*).

For a number of applications there exists a similar paradigm of *ordered* incoming views of a particular object. A baseline classifier that treats views as i.i.d. samples from the view sphere yields identical results for unordered and ordered view sequences and potentially loses out on the additional structural information mentioned above.

The notion of a sweep is generic enough to capture the reversed setup where the camera is fixed and the object undergoes characteristic motion, for example in a robot hand. Similarly, any feature that allows establishing correspondence between successive frames (such as SIFT) is suitable in principle for the extension we presented in this chapter.

We can think of the following extensions to our current implementation. First, one could move from a *cost-based* recognition approach to training a discriminative classifier in a vector space that holds both shape and motion information. In [53] the basic idea for mapping shapes into a vector space is outlined as follows: given a fixed *prototype shape* and a novel object, one records the matching costs between both for each of the (100) local histograms in a vector. The prototype shape serves to anchor the order of the coefficients in the vector. With this representation, standard learning methods, such as logistic regression can be employed to train a classifier. We suggest to add feature motion information to the vector, much like the way that Agarwal and Roth use for adding geometric relations to their feature vectors [1]. Here, angle and distance are discretized into one out of twenty buckets for each point and appended to the vector.

Second, we could adapt our model to allow for different sampling frequencies along the trajectory, stemming for example from different velocities along training and testing trajectories. This may be possible by associating each sweep with a left-to-right HMM similar to the usage in handwritten character recognition [33]. Here, each state corresponds to one view in the sweep and is associated with the angle and normalized distances of the expected feature motion at each point for some fixed frequency. Under the assumption that for lower sampling frequencies, distances extend as constant multiples of the normalized distances above, one can use the Viterbi algorithm to recover a left-to-right state sequence (which may include self-loops) that best fits the observed motion. The sweep with the highest associated probability is then chosen as the one best explaining the observed pattern.

Chapter 4

Adding Control

The focus of this chapter is to reduce the number of pictures required to predict object class membership with a certain degree of confidence. The chapter revolves around the notion of *characteristic views* which we consider particularly salient for disambiguation between object classes.

The recognition model outlined in the previous chapter corresponds to a sequence matching approach that compares sensory traces of a novel object with those stored in a trained model database. Active perception systems, on the other hand, additionally have access to the sensorimotor context (i.e., motor commands) that different observations occur in. In our implementation, for example, we are able to associate each image taken by the camera with the corresponding joint angles of the active vision head. For the online implementation of our system, we therefore consider adding a control layer to the previously outlined model, allowing us to optimize camera motion for more efficient object recognition.

In the following, we first derive a working definition of the characteristic view concept. We attempt to determine whether the notion of a characteristic view is only valid *in relation* to other views or if there are intrinsic properties that render a view "characteristic". This goes hand in hand with the question whether assignments of distinctness to different views have to be revised as more views of the same object are discovered (*dynamic term*). To answer these questions, we also turn to recent literature from psychophysics to determine whether there exists agreement about canonical object views among humans.

In the final section of this chapter we devise a strategy to integrate steering towards high-quality views into the recognition system detailed in the previous chapter.

4.1 Characteristic View Selection

The quality of a view of a particular object can be measured according to different criteria. In the psychophysics literature, for example, the notion of "canonical views" has seen a number of different interpretations ranging from those views associated with the lowest response time or recognition error rate to those subjectively classified as prototypical for an object [14]. Multiple experiments have been conducted to test for common view choices among human subjects for all of these different interpretations, attempting to uncover something "intrinsic" about canonical object views.

Recent experiments in [9] have demonstrated that the preference of human subjects toward similar, canonical views depends largely on

- The exact criteria for view selection chosen in the experiment
- The familiarity of the subjects with the respective objects

In the study, subjects were asked to orient 3D CAD models of familiar and unfamiliar objects into a canonical pose. The first experiment asked subjects to select a representative object view for placement in a brochure while the second experiment asked users to imagine a familiar object and to align the 3D model with the mental image.

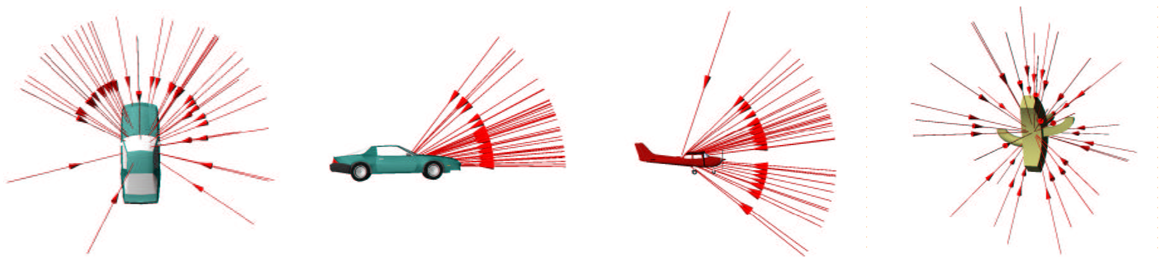


Figure 4-1: From [9]. Choice in canonical views selected during the first experiment in the paper for two familiar objects (car and airplane) and one unfamiliar structure.

The findings show that within the confines of individual experiments, the notion of canonical viewpoints is indeed a valid one for *familiar* objects. The canonical views preferred in the first experiment differ from those in the second, however. Whereas subjects predominantly chose off-axis views with many visible surfaces in the first experiment, the frequency of frontal or side views increased largely in the second experiment. The images shown in Figure 4-1 reproduce the canonical views selected during the first experiment for the car, airplane and for one unfamiliar object.

It is clear that familiarity with the objects greatly impacts the choice of characteristic view. In both experiments, views are chosen so that objects appear in their natural, upright position. As shown in the two middle images in Figure 4-1, familiarity with cars and airplanes also leads to varying choices in viewpoint based on the function of the object. For unfamiliar objects, such as the rightmost structure in the same Figure, there generally *do not exist* canonical views. Among those poses that avoid occlusion of parts, no particular pose is preferred suggesting individual differences in coding.

These results invalidate some of the earlier work in [34] that suggested universal canonical views based on inherent geometrical properties present in the view. Instead, the authors of [9] suggest that conveying as much information as possible about the object drives canonical view selection in the first experiment while mental images obtained in the second experiment underlie storage limitations resulting in more simplistic, planar views.

In order to arrive at an operable definition of a *characteristic view* that we can use to steer the camera to, we have to commit ourselves to one of the view selection criteria described above. Since it was established that there is no psychophysical evidence for intrinsic geometric qualities that we could exploit, we adopt the most sensible working definition based on classification accuracy. We use a concept from information theory – entropy – to evaluate the quality of each stored view for disambiguating the object class.

4.1.1 Entropy-based View Selection

Entropy measures the information content of a random variable x with associated probability distribution $P(x)$. It denotes the average information required to define the state of x and, for discrete distributions, is given as

$$H[x] = -\sum_{x} P(x) \log_2 P(x) \tag{4.1}$$

Higher entropy is associated with a more uniform distribution $P(x)$ and, conversely, low entropy indicates more "peaked" distributions.

We can use this measure to rank all known views in the training set based on their goodness of predicting the correct class. The rationale is that those views resulting in low-entropy distributions $P(\mathcal{C}_k|\mathbf{x}_{SC})$ have more predictive power than the ones with distributions of higher entropy.

Applying the Entropy measure:

For our problem at hand, we initially perform an offline operation on the whole set of training images. For every image, we perform shape context matching against all other images. We then compute $P_i(\mathcal{C}_k|\mathbf{x}_{SC})$ for every image i, based on the smallest observed matching costs between i and all object classes. After that we associate view i with the entropy of that distribution, yielding effectively an *entropy map* for each sweep. In our implementation we assume that areas of low entropy in a sweep yield more informative views and, consequently, lead to more effective object recognition.

It is noteworthy that our definition of the canonical view concept makes it both a *relative* and a *dynamic* term in the sense that entropy maps have to recomputed over the whole training set as new views are added to it.

To verify the performance of our entropy-based view selection criterion, we carried out an experiment with a limited set of objects. We hand picked the objects so that they would exhibit both informative and uninformative views depending on the camera's viewing direction. As demonstrated in our opening example of chapter 1, cylinders and spheres share a set of common, round views. We add to that list an egg and a cone object that produce similarly round views when observed from directly above. The goal of the experiment is to establish that our entropy measure would pick more distinctive views from each sweep.

In Figure 4-2 we show the recorded sweeps over each object. All of them share a similar, round view and more characteristic views of the respective class. For sake of space, we show the exhaustive shape context matching results for only three of these objects in a Hinton-like diagram in Figure 4-3. In the diagram, the size of each white patch reflects the magnitude of the matching cost between both views. Reading that diagram row-wise for views $1-12$ (*cylinder*), $13-34$ (*egg*) and $35-40$ (*sphere*), one can immediately observe that those views close to i (i.e., close to the diagonal) are assigned lower matching costs (grayer patches), as expected. We can also see that less characteristic views toward the end of the respective class boundaries increase the confusion with the other round views, such as all six images of the sphere.

Figure 4-4 shows, now for each of the four classes, the views i associated with the

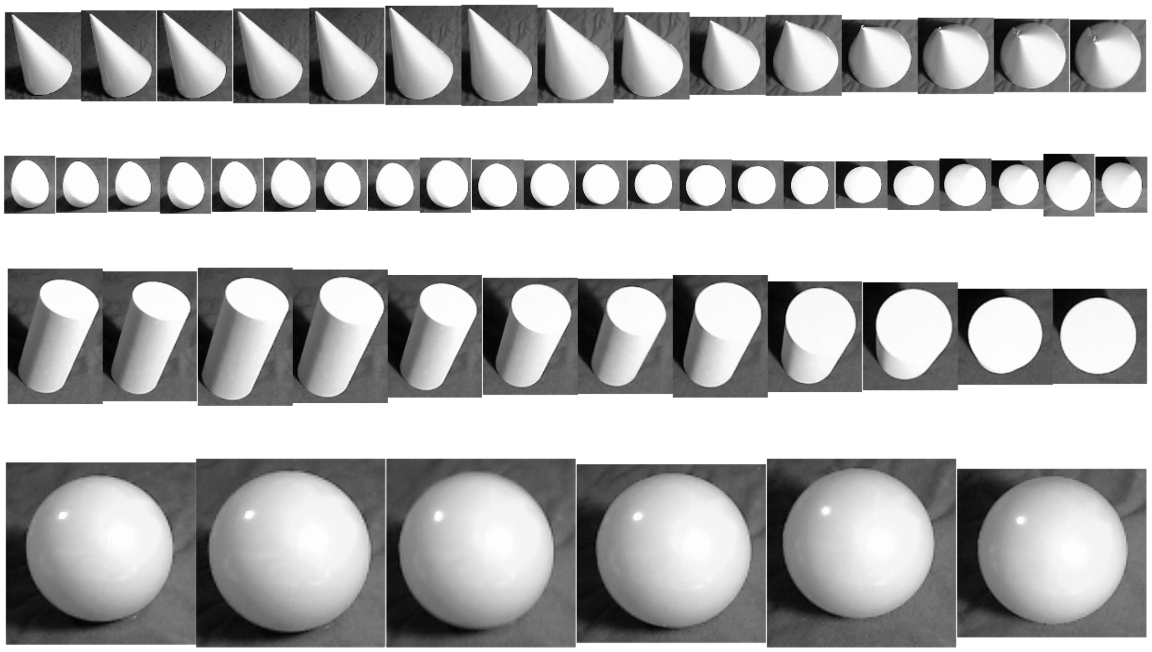


Figure 4-2: The sweeps over *cone*, *egg*, *cylinder* and *sphere* objects used in the characteristic view experiment.

lowest entropy distributions $P_i(\mathcal{C}_k|\mathbf{x}_{SC})^1$. This confirms our intuition that entropy is a valid measure for extracting *canonical views* leading to high disambiguation between classes.

In the following section we describe how we integrate these entropy measurements into our recognition system. The overall goal is to exert control over the camera in order to resolve category membership in a more timely fashion.

Finally, we would also like to note that we are by far not the first to adopt entropy as a measure for selecting salient points on the view sphere. In [2], for example, an active vision system is described that computes entropy distributions over the whole view sphere and steers the camera to points of lowest entropy, much like in our implementation.

[1]A more extensive break down of the entropies and the respective views can be found on a website accompanying this experiment at http://web.mit.edu/robbel/Public/exp/

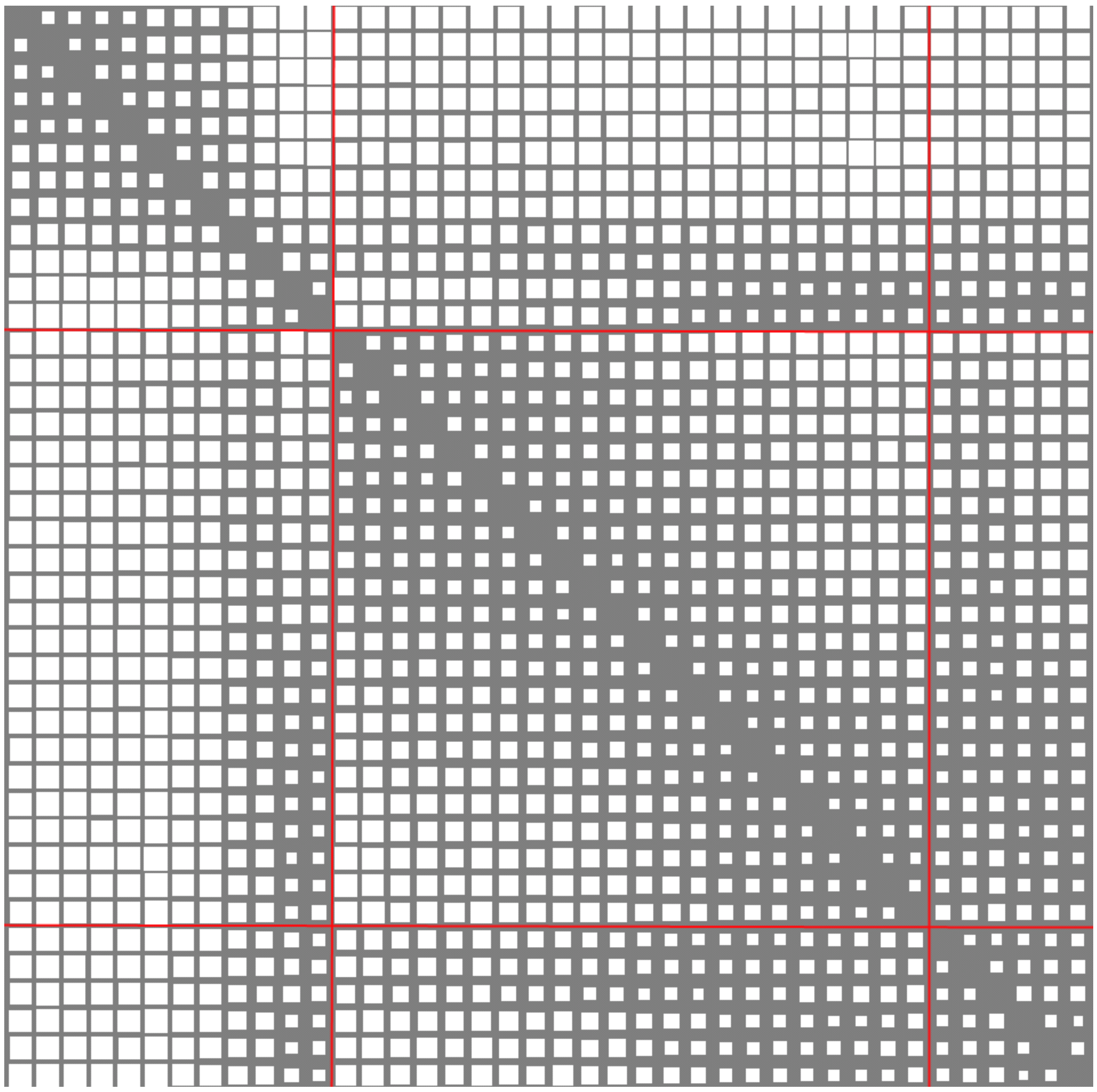


Figure 4-3: The exhaustive matching costs between *cylinder*, *egg*, and *sphere*. Red lines denote class boundaries. Grayer patches denote lower matching costs.

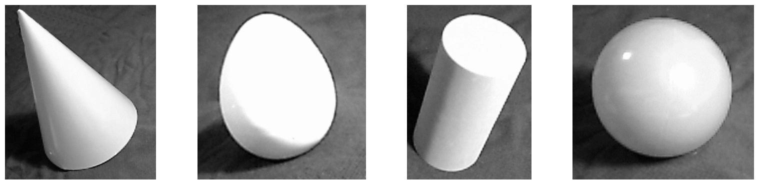


Figure 4-4: The canonical views associated with the lowest entropy distributions.

4.2 Integration with the Previous Model

In the implementation of our recognition system outlined in the previous chapter, we rely on the robot camera following identical trajectories during training and recognition to be able to exploit both information from *feature motion* $\mathbf{x}_M$ and *shape context matching* $\mathbf{x}_{SC}$. Our current motion repertoire consists of three distinct trajectories that we follow linearly until a stopping criterion (an entropy threshold θ_2 for $P(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M)$) is achieved.

In this chapter we developed an offline method to compute the ability of each view in the training set to make the correct class prediction. This metric introduces a natural *ordering* of the views in each sweep based on the entropy of the associated posterior class distribution. The relative positions of desirable low-entropy areas in a sweep are generally not the same for different objects, however. Depending on the object type and its pose in front of the camera, characteristic views may occur anywhere in the sweep. Before we can make a decision as to where to move the camera, we must have a certain degree of confidence about the object in order to choose the correct location in the sweep leading to maximum reduction in class ambiguity.

Based on this observation we suggest a simple method to merge our previous system with entropy-based view selection. As before, the system starts traversing one of the fixed trajectories. The entropy of the posterior distribution $P(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M)$ is monitored throughout the motion. Whenever we arrive at a certain confidence level about an object (based on an entropy threshold θ_1), we assume that we can speed up disambiguation by skipping ahead in the current sweep, specifically to the point where we found the posterior entropy to be lowest in our offline computation. After this greedy selection, we restart trajectory following until our initial acceptance threshold θ_2 is achieved. With this approach, we are still able to make use of the feature motion information $\mathbf{x}_M$ except for at the point that we jumped to.

4.3 Comments

In this chapter we presented a simple method to add active view selection to the models introduced in chapter 3. This method is based on a sound measure from information theory and allows – after a hypothesis about the object type has been made – the navigation to characteristic views on the recorded trajectories.

While we restrict ourselved to distinctive *views* here, one could similarly imagine skipping ahead to places where characteristic *feature motion* is expected (based on the entropy of $P(\mathcal{C}_k|\mathbf{x}_M)$).

Finally, in the current implementation we only skip ahead to views on the *same* trajectory. In the same way, however, it is possible to skip across trajectory boundaries by searching through the entropies of the posterior distributions associated with all other trajectory points.

Chapter 5

Experimental Evaluation

In this chapter we demonstrate the performance of our object recognition engine on two different datasets. The first is the *ETH-80 dataset* [4] which is specifically geared to evaluate category recognition methods, and the second consists of a number of objects collected with Trisk's camera under typical operating conditions.

In this evaluation we attempt to compare our current shape recognition algorithm as it is implemented on the robot, a novel baseline system, as well our two extensions suggested in the previous chapters. In detail, the approaches considered here are:

- **Single-frame Shape Context:** This implementation corresponds to the current shape recognition system used on Trisk. Given a single view of the object, shape context distances are computed to every object in the database and the one with the lowest overall cost yields the category label (*winner-takes-all*).
- **Multi-frame Shape Context (SC):** This establishes the new baseline system for this thesis. Object recognition relies on trajectory following and the Bayesian-updates of the posterior $P(\mathcal{C}_k|\mathbf{x}_{SC})$ as derived in section 3.3. As soon

as the entropy of the posterior reaches a threshold, the class $k = \arg\max_k P(\mathcal{C}_k|\mathbf{x}_{SC})$ is selected as the category.

- **Feature motion-based Classification (M):** This classifier consists solely of the feature motion information derived between frames. Naturally, object recognition requires traversal of a known trajectory to update $P(\mathcal{C}_k|\mathbf{x}_M)$. Similarly to the case above, an entropy threshold determines the stopping condition and the maximum a posteriori (MAP) solution determines the assigned class.

- **Multi-frame Shape Context with Feature Motion (SC+M):** This approach joins our baseline (SC) with the feature motion-based system (M), updating $P(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M)$ along a known trajectory. Similarly to the cases above, the MAP solution is selected after the stopping criterion has been achieved.

- **Active view selection-based Classification:** This method combines SC+M with the additional control layer detailed in chapter 4. In particular, as soon as the entropy of the posterior $P(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M)$ reaches a predetermined threshold θ_1, we skip ahead to the view deemed best for disambiguation on the remainder of the trajectory. As before, the MAP solution is computed after the stopping criterion θ_2 is achieved.

In summary, the original claim of this thesis that the spatial ordering of the incoming camera images reveals feature motion which in turn reveals information about the object structure is tested in this evaluation. Our new baseline, the multi-frame shape context classifier (SC), loses this information by treating all images as an unordered sequence. This classifier produces the same class predictions for different permutations of the same set of images. Still, the number of images required to arrive at a quality class prediction may vary largely depending on the chosen permutation. Our active view selection-based classifier attempts to exploit this fact by steering the camera to low-entropy regions as computed over the training set.

Before starting our review of these approaches, we give a brief overview of the datasets used in this evaluation.

5.1 Datasets

5.1.1 ETH-80

The original ETH-80 dataset consists of eight object categories with ten instances per category. Instances vary in shape and texture but otherwise appear on a uniform background and roughly share the same size. Each instance comes with 41 images that are distributed evenly over the upper view sphere at a resolution of 256×256 pixels.

The use-case suggested for this dataset in [4] is leave-one-out cross-validation. Since we operate on object *sweeps* instead of single images, we initially have to introduce an order over the included images to simulate camera trajectories for that dataset. Unfortunately, the images taken across the view sphere do not stem from smooth, curvilinear camera paths but instead from equally spaced points on an octahedron approximation to the sphere. In Figure 5-1 we show the approximate trajectories we adapted for the dataset, resulting in four trajectories overall (two as shown and two for the backside of the sphere).

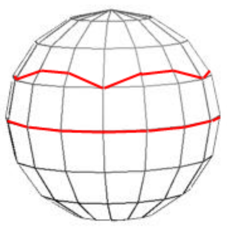
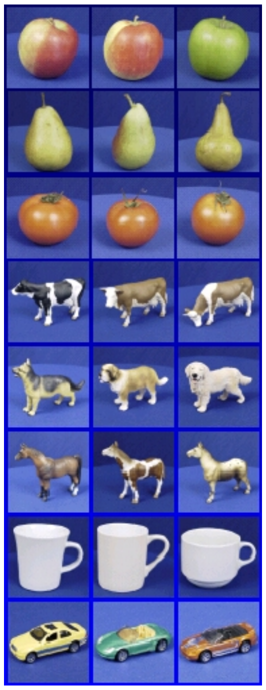


Figure 5-1: Two of the trajectories adapted from the ETH-80 dataset.



Categories	*Instances*
Apple	3
Pear	3
Tomatoe	3
Cow	3
Dog	3
Horse	3
Cup	3
Car	3

Figure 5-2: The three selected instances per category from the ETH-80 dataset [4]. For each such instance, we store four sweeps (cf. Figure 5-3), yielding 672 images.

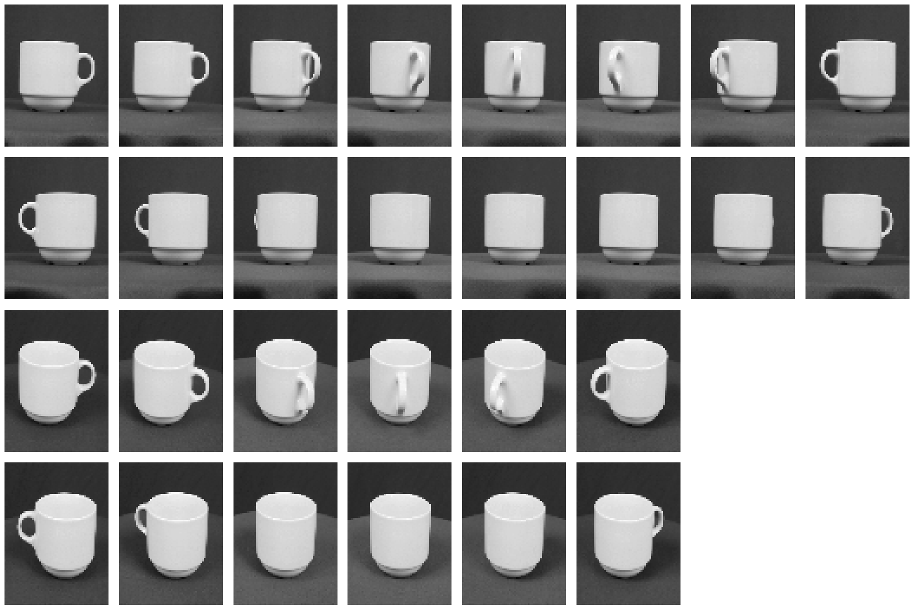


Figure 5-3: The four sweeps associated with the third cup instance.

The resulting image assortment for all four trajectories is visualized for the *cup* example in Figure 5-3. As shown, each trajectory results in a unique sweep of either eight or six images.

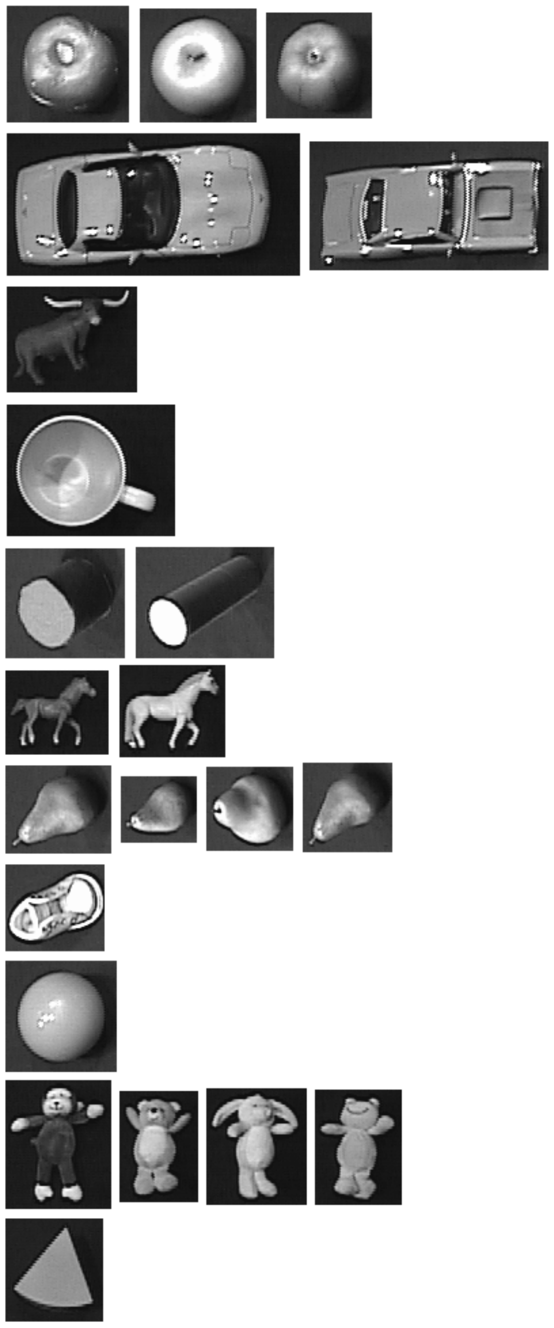
Our entire training set consists of the first three instances per object class from the ETH-80 data, resulting in an overall number of 672 images. Prototypical views for each instance are shown in Figure 5-2. Note, however, that being a shape-based method, we discard all color information from our dataset.

5.1.2 Trisk-22

The Trisk-22 dataset consists of a set of 22 real-world objects collected from the active vision head under real working conditions[1]. As shown in Figure 5-4, the dataset is divided into eleven categories with a variable number of instances per category. Images are generally low-resolution, of different sizes and taken under varying lighting conditions. All views are obtained during the traversal of three pre-determined head trajectories and are sampled at regular 1 second intervals. There is a total of 1010 object views contained in the Trisk-22 database.

In Figure 5-5 we show a typical set of sweeps obtained for three different instances in the dataset. At the bottom of the same Figure we also visualize a number of views with their extracted motion information overlaid. It is clear that due to imaging artifacts, such as from specular highlighting, the contour samples and their motion vectors can be expected to be much more noisy than their counterparts in the ETH-80 dataset.

[1]The Trisk-22 dataset can be found at http://web.mit.edu/robbel/Public/Trisk-22



Categories	*Instances*
Apple	3
Car	2
Cow	1
Cup	1
Cylinder	2
Horse	2
Pear	4
Shoe	1
Sphere	1
Teddy	4
Triangle	1

Figure 5-4: The 11 object categories and object instances in the Trisk-22 database.

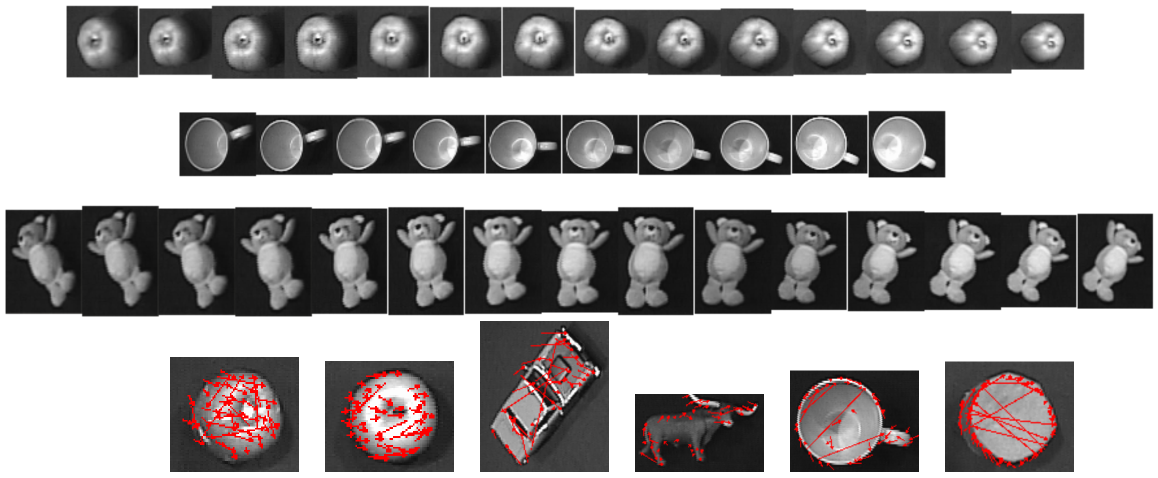


Figure 5-5: Three distinct object sweeps in Trisk-22 (*top*). Indicated feature motion on a set of views contained in the database (*below*).

5.2 Experimental Results

5.2.1 ETH-80

After having introduced the division of object views into sweeps, we are interested in the question how long the system has to trace a sweep in order to arrive at a quality prediction of the object class. In the following, we look at the *posterior class distributions*, the associated *entropies* and the *recognition performance* for the systems described in this thesis.

The general approach we take here is leave-one-sweep-out cross-validation, i.e., for each possible hold-out sweep we retain the remainder of the data for training purposes.

Parameter choices: Unless otherwise noted, the following experiments are executed with a sampling size of 50 along the Canny-extracted object contours. We use our rotation-invariant modification to the shape context algorithm and choose a discount factor of 0.3 for the bending energy in the overall cost term (in accordance with the original authors' settings). In our tests of the SC+M model, we weigh input from the feature motion and shape context model equally across the sweep.

Previous implementation: Figure 5-6 shows the leave-one-out error rate for the *single frame-based* shape context classifier on our ETH-80 subset. This classifier corresponds to the simplest possible model – and to our current implementation on the robot – and proceeds by computing the shape context distances to each of the stored models and outputting the MAP class estimate for every frame (essentially a nearest neighbor classifier with the shape context distance metric). Contrary to the proposed, *sweep-based* classifiers, this recognition system operates purely on a frame-by-frame basis and does not take the object history into account.

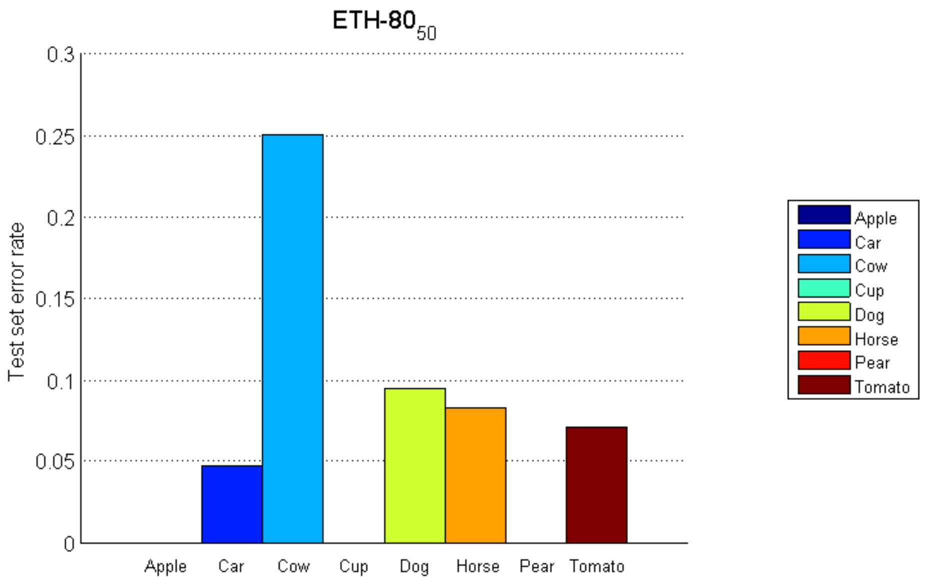


Figure 5-6: The test set error rates for one-shot shape context matching on our subset of the ETH-80 database.

The following experiments summarize the recognition performance of the different types of sweep-based classifiers suggested in this thesis.

Experiment 1: In this experiment, we compute the average posterior distributions

$P(\mathcal{C}_k|\mathbf{x}_{SC})$, $P(\mathcal{C}_k|\mathbf{x}_M)$, and $P(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M)$ obtained over a range of 1-6 impressions of the same object. These distributions respectively correspond to the output of the *shape only* (SC), the *feature motion only* (M) and the *combined* (SC+M) classifiers and are averaged over all twelve sweeps per object category (3 instances times 4 sweeps per instance).

The test set error rates reported in Figure 5-6 allow us to estimate a measure of difficulty for each of the object categories. For sake of space, we report the results of experiment 1 for an object type of high (*cow*), medium (*dog*), and low (*apple*) difficulty. We make all other results available in the supplementary Figures section in Appendix A.

Figures 5-7 to 5-9 convey the main results of this experiment. Shown are the mean posterior distributions together with the standard deviation error bars for the SC, M, and SC+M classifiers. We also show the reduction in entropy for each of the average posterior distributions throughout the sweep. For a color-coded legend for these and the following diagrams, please refer to the right side of Figure 5-6.

We can draw the following conclusions from this experiment:

- Using motion as an additional source of information is generally valid. For all but the cow data (more on this later) does the motion-based classifier produce results that boost the posterior probability of the correct category. This verifies the original claim of this thesis that local feature motion can be exploited for object recognition.

- Motion information does not replace traditional local features. As seen in Figures 5-7 to 5-9, the entropy of the posterior $P(\mathcal{C}_k|\mathbf{x}_M)$ is generally higher throughout the sweep. In the same set of diagrams we can also observe how $P(\mathcal{C}_k|\mathbf{x}_M)$ evolves from an uninformative (cf. first view where no motion data has been obtained yet) toward more informative distributions as more object

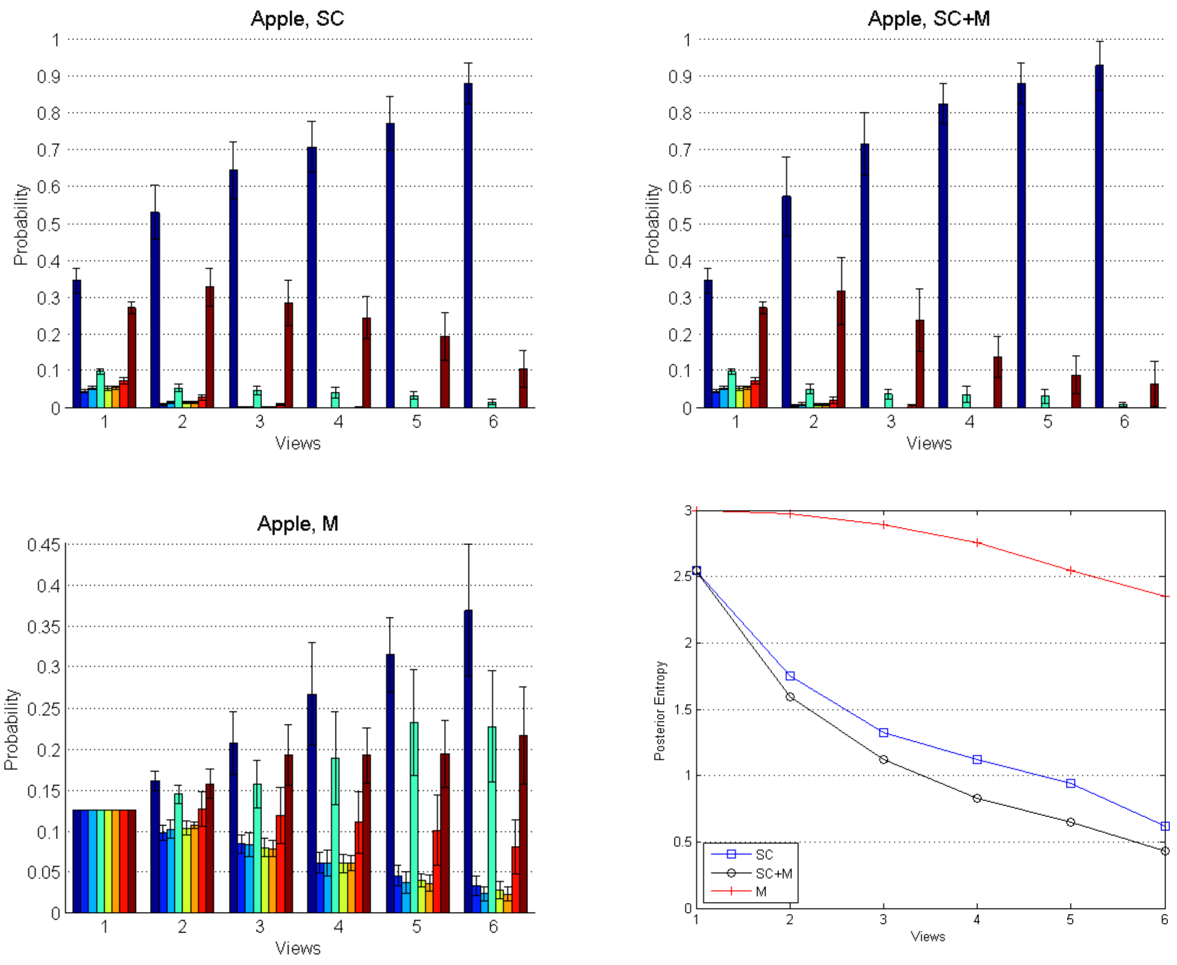


Figure 5-7: Mean posterior distributions after a specified number of *apple* views for SC, SC+M, and M models. The entropy of the distributions is shown on the bottom right.

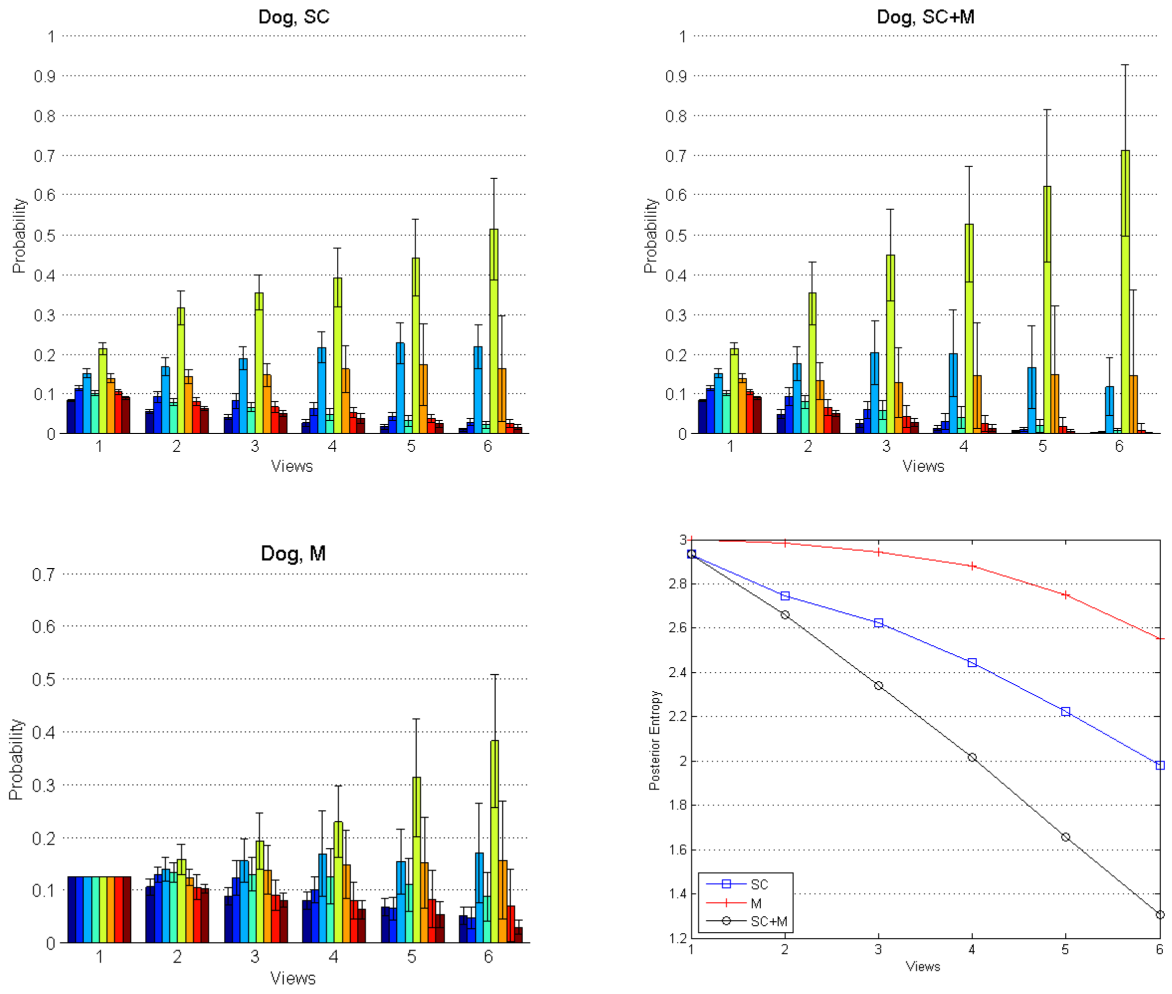


Figure 5-8: Mean posterior distributions after a specified number of *dog* views for SC, SC+M, and M models. The entropy of the distributions is shown on the bottom right.

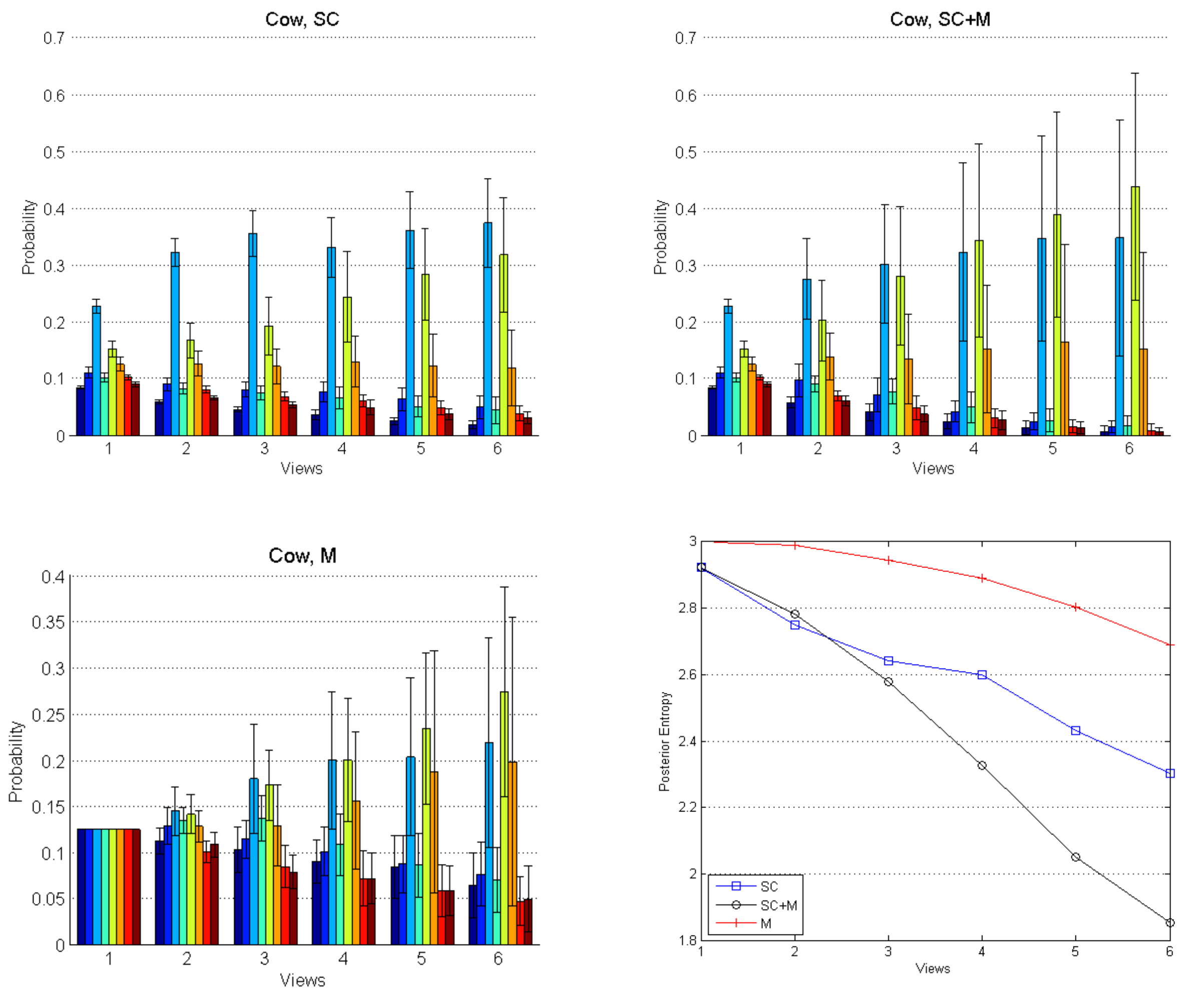


Figure 5-9: Mean posterior distributions after a specified number of *cow* views for SC, SC+M, and M models. The entropy of the distributions is shown on the bottom right.

views are obtained. It is clearly visible, however, that the contention between *similar* objects (such as apple and tomato in Figure 5-7) is maintained much longer for the feature motion-based (M) than for the shape only classifier (SC).

- Joining both classifiers (SC+M) generally leads to a desirable posterior distribution (as judged by both the correctness of the MAP estimate as well as the posterior entropy) quickest. The level of effectiveness of the motion information differs from class to class, however. We experience moderate gains in posterior probability over the baseline system for the apple category but significant gains for the dog class, for example. In the latter case, we achieve a mean posterior probability $P(\mathcal{C}_{dog}|\mathbf{x}_{SC}, \mathbf{x}_M)$ of 0.7 versus 0.5 for the shape only classifier after six views.

The example of the cow category deserves special attention as both SC and M classifiers (and, by design, SC+M) underperform for this class. We can make out two reasons for this behavior and attempt to fix the second in the following section. First we note that the cow category in our chosen dataset encompasses three substantially different cow instances as is visible in Figure 5-2 (head oriented left, right, and downward). We show in Figure 5-11 at the bottom of the next page that a purely shape-based local descriptor reaches its limits in this scenario as the inter-class distance between cows, horses and dogs, for example, can be smaller than the cow within-class distance for some views.

The second issue lies in the implementation of the classifier itself, specifically the incremental updates of the posterior $P_i(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M)$ at every time step i during run-time. In our original implementation, we maintain posterior distributions $P_i(\mathcal{C}_{k,s}|\mathbf{x}_{SC}, \mathbf{x}_M)$ for every recorded sweep s of all classes k in the database and choose

$$P_i(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M) \propto \max_s P_i(\mathcal{C}_{k,s}|\mathbf{x}_{SC}, \mathbf{x}_M) \tag{5.1}$$

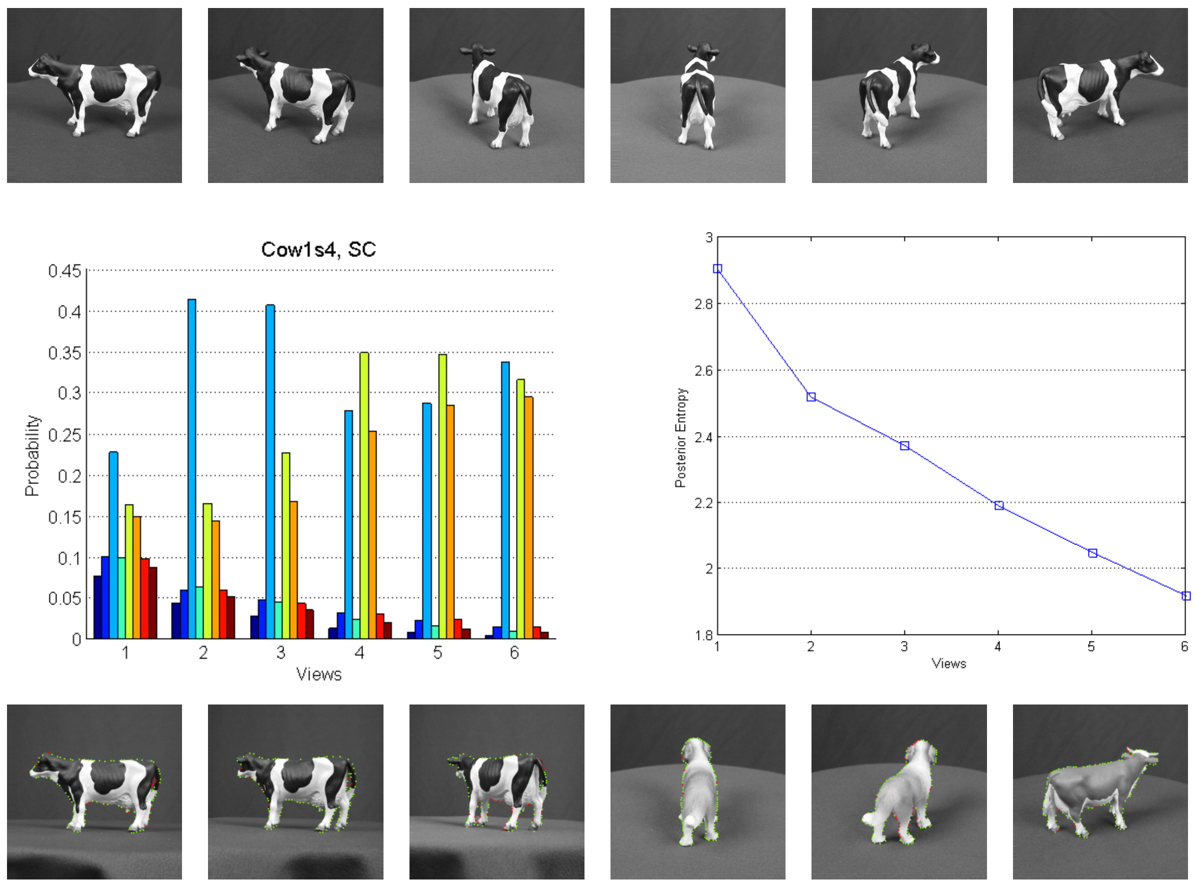


Figure 5-10: SC model results for a particular *cow* sweep. The MAP *sequence* estimate is shown at the bottom.

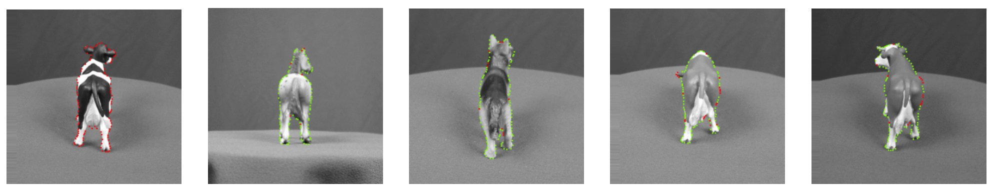


Figure 5-11: Point of failure of the shape context algorithm: shown are views closest to the first *cow* view in order of their relative matching costs. From left to right, *horse* (100%), *dog* (127%), *cow1* (142%) and *cow2* (155%).

for the posterior class distribution at time i.

This effectively leads to a contention between entire sweeps as it is shown for an exemplary cow sweep in Figure 5-10. In the Figure, besides the incoming object views (top) and the evolving posterior distribution and entropy over classes (center), we also display the maximally scoring sweeps at the bottom. It can be seen that a particular cow sweep s in the training database can be maintained throughout the first three views before a dog sweep becomes more probable. The reason that s loses out is related to the low matching score between the following *observed* image and the next frame in s. To work around this issue, we adapt our implementation as outlined in the following section.

Adapting the model:

For our second implementation (referred to as *SC2*, *M2*, and *SC+M2* in the following) we compute the posterior class distribution at time i slightly differently. Instead of choosing the *sweep* with the maximum posterior probability as shown in equation 5.1, we now perform updates to $P_i(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M)$ directly, doing away with the explicit posteriors over sweeps:

$$P_i(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M) \propto P_{i-1}(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M) \max_{v_i} P(\mathcal{C}_{k,v_i}|\mathbf{x}_{SC}, \mathbf{x}_M) \tag{5.2}$$

where v_i denotes all *views* at position i in *any* of the sweeps associated with object k.

This tweak to the model implementation essentially relaxes the requirement that both observed motion and shape patterns have to stem from a unique sweep in the training set. We report the results of the new SC2 model on the same exemplary cow sweep from the previous page in Figure 5-12. The introduced measure yields the correct MAP estimates throughout the entire sweep and also reduces the posterior entropy significantly.

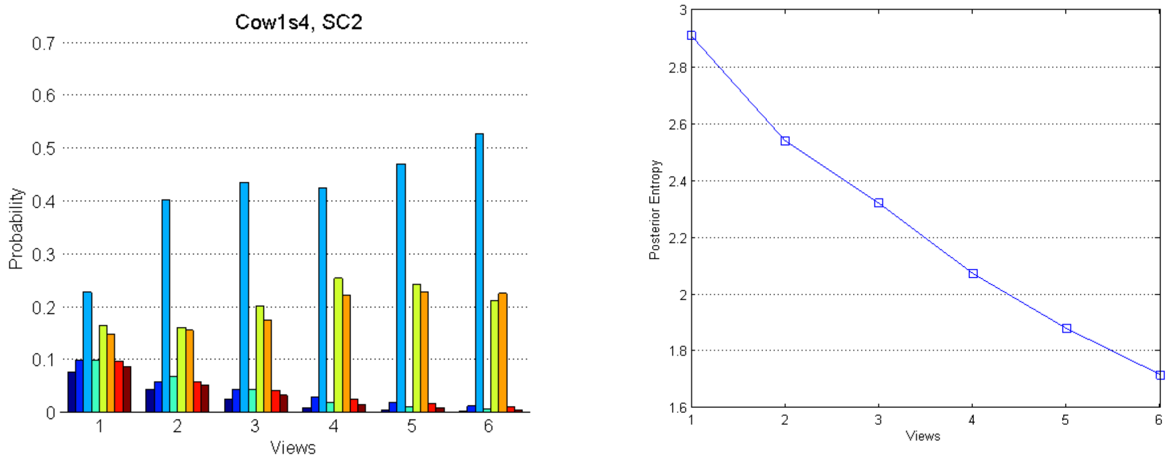


Figure 5-12: SC2: Matching over categories instead of entire sweeps.

The mean performance of the new SC2, M2, and SC+M2 classifiers on the entire cow category is shown in Figure 5-13 on the following page. This can be directly compared to the earlier result (Figure 5-9), revealing improvements to the average class posteriors of both feature motion-based and shape only models. We show the results for the remaining, simpler categories in the supplementary Figures section in Appendix A.

Active view selection:

We have seen from the cow example above that certain views of an object are more indicative of its category than others. Similarly, as shown in Figure 5-11, based on the nature of the training set, certain views may result in an incorrect MAP class estimate if the within-class variation is large. In order to deal with these issues, we suggested an active view selection scheme in chapter 4 that would exercise control over the camera position to skip ahead to locations where more informative views are expected.

In an initial computation over the training database, the entropies of the posterior class distributions associated with each image are computed exhaustively. The general

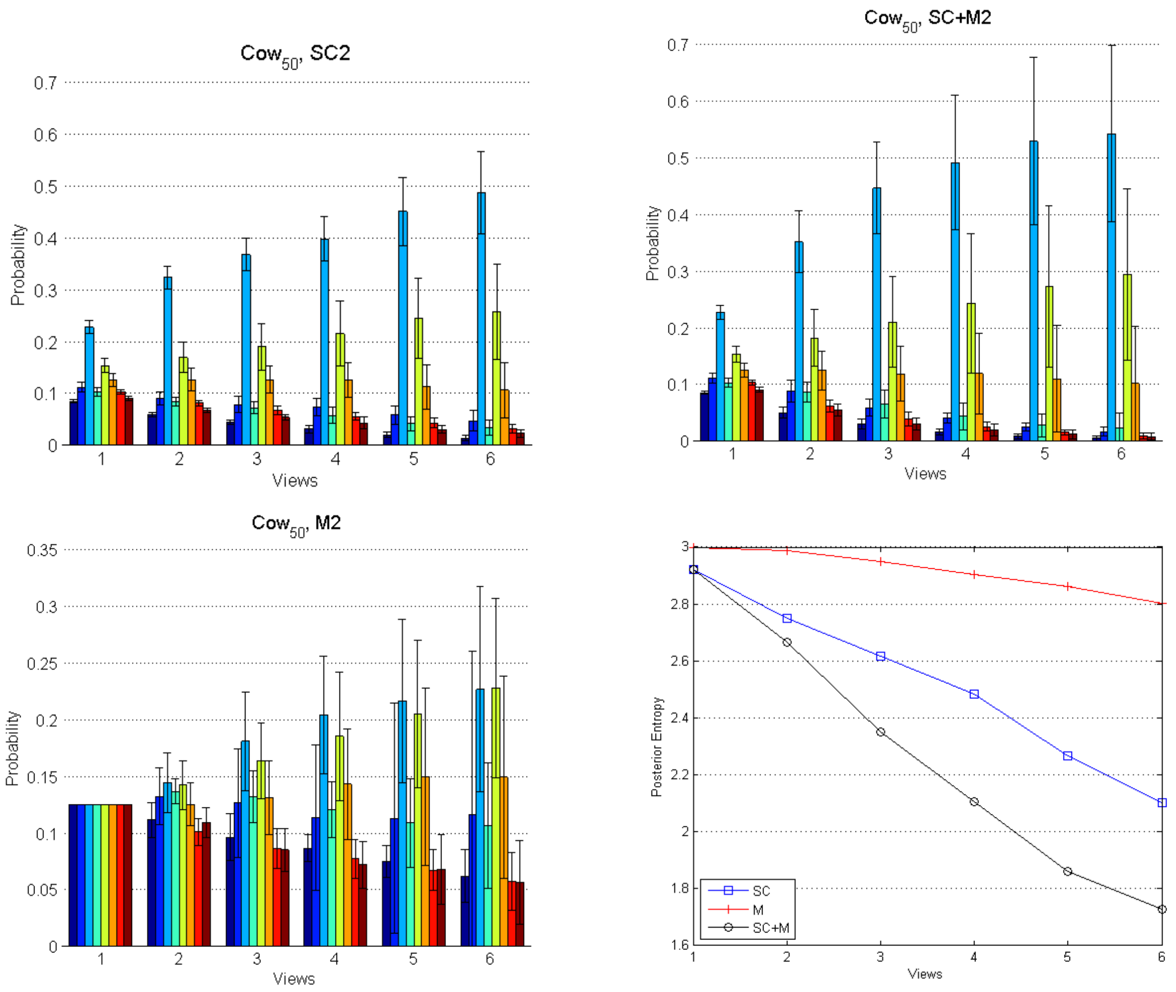


Figure 5-13: Mean posterior distributions after a specified number of *cow* views for SC2, SC+M2, and M2 models. The entropy of the distributions is shown on the bottom right.

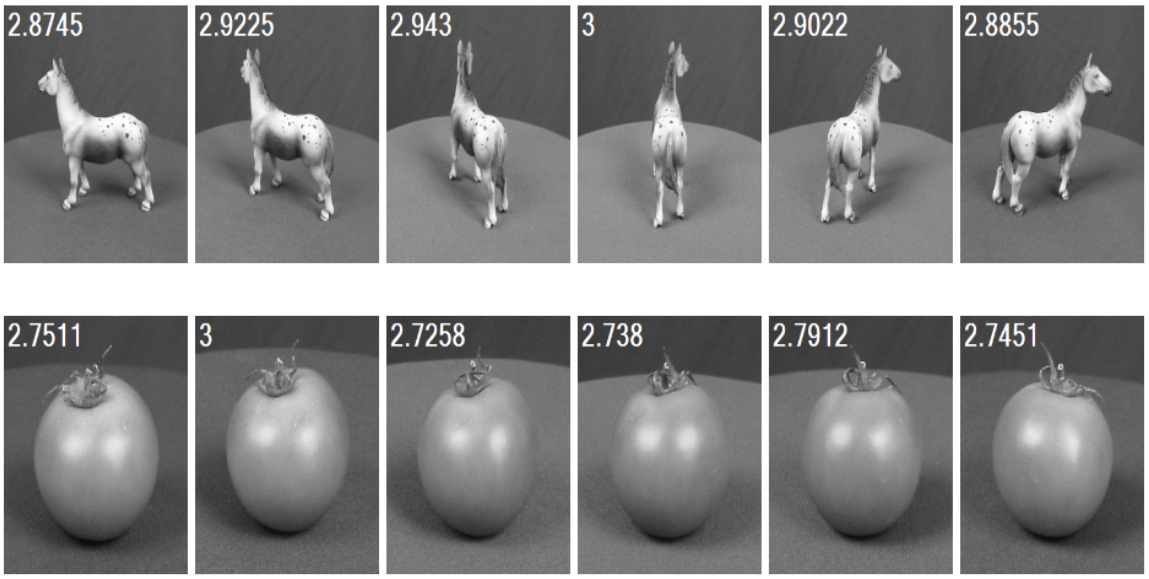


Figure 5-14: Two objects with the entropy of the associated posterior distributions overlaid. Values of 3 are associated with those views that result in incorrect MAP class estimates.

procedure is then to establish an initial category estimate and to greedily skip ahead to high quality views associated with the correct MAP estimate and low-entropy posterior distributions.

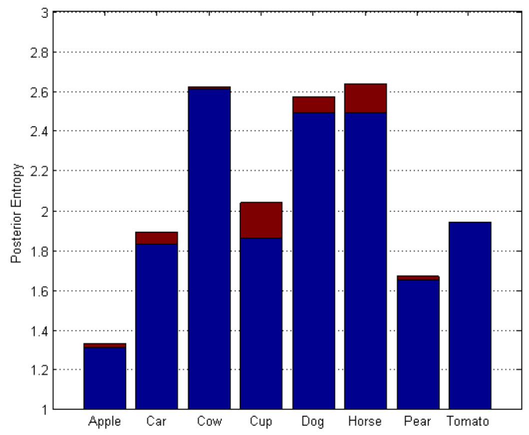
Figure 5-14 above shows the entropies associated with all views in two example object sweeps. Views with incorrect MAP class estimates were manually set to the maximum entropy of 3 (a uniform distribution over 8 categories). Throughout the dataset we note that frequently the back- or frontal views are less indicative of the class than the side views which is in agreement with our earlier observations about the cow sweep. In the following experiment we demonstrate the performance of the classifier with active view selection on the same subset of the ETH-80 database.

Experiment 2: For this experiment we augment the SC2 system with active view selection. After an entropy threshold $\theta_1 = 2.5$ of $P(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M)$ has been achieved (generally after two views), we determine the current best sweep estimate s^* and skip ahead to the next view on that same sweep with the lowest associated posterior entropy. Because of particularities of the dataset there exist sweeps with only three valid views (in the cow category) so that we report classification results that we obtain after observing only three views of each object.

Figure 5-15 reveals the overall reduction in average posterior entropy that we achieve with active selection over the earlier system after three views. On the right side of the same Figure we show the test set error rate for each of the object categories after the same three views. The error rate reported here refers to the leave-one-sweep-out cross-validation result and shows how many of the twelve hold-out sweeps per category were misclassified after three object views have been obtained. For the case of the tomato, one out of twelve sweeps was initially (after θ_1) classified as belonging to the apple category and the skip-ahead yielded another view that was uninformative with respect to the tomato class.

The reduction in posterior entropy for the active view selection method translates into more confident MAP classification decisions. In Figure 5-16 we show the average posterior distributions over object categories after three views of cups and dogs have been obtained. This directly compares to Figures A-3 and 5-8 where we played back the camera trajectory without active skipping, resulting in less indicative posteriors after the third view.

It should be noted, however, that the gains from active view selection presented here for our ETH-80 subset are moderate. This is largely due to our division of the dataset into sweeps where the initial two views are usually side views and indicative of the object category (cf. Figure 5-3).



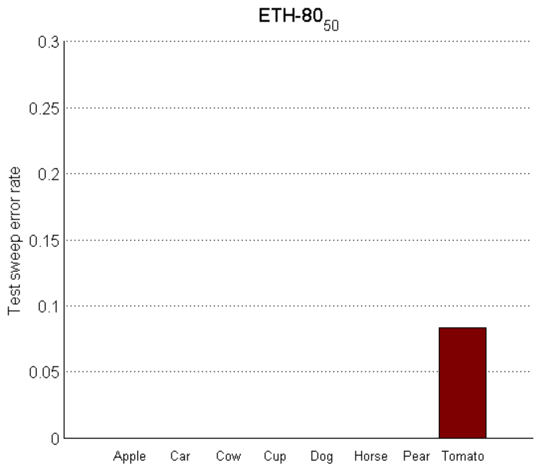
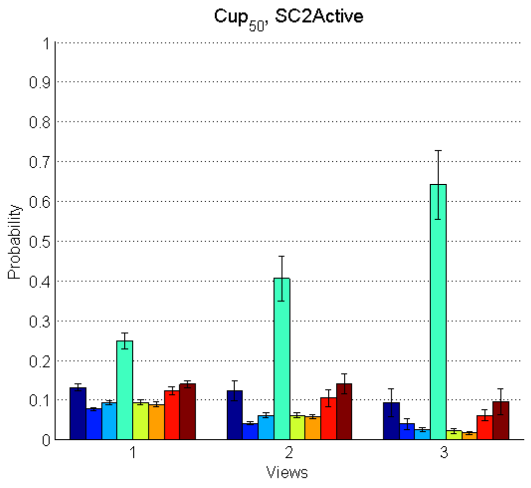


Figure 5-15: Reduction in posterior entropy after active view selection (*left*) and test set error rate after three actively selected views (*right*).



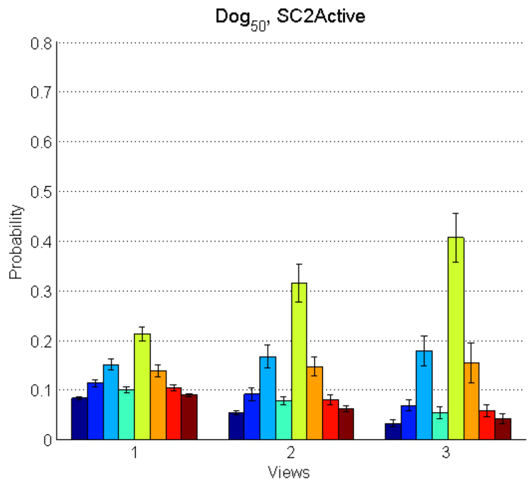


Figure 5-16: Mean posterior distributions after a specified number of *cup* and *dog* views for the active SC2 model.

5.2.2 Trisk-22

For the experiments in this section, we continue to use the same parameter choices for the shape context algorithm as before, most importantly a sampling size of 50 points along the internal and external contours of the objects. Because of the demonstrated performance on the ETH-80 dataset, we only evaluate the SC2 and SC+M2 models in a similar, leave-one-sweep-out cross-validation fashion.

Experiment 3: In this experiment, we look at the average *posterior class distributions* resulting from the classification of the hold-out sweeps. We also evaluate the number of views that are required on average to obtain a perfect *recognition performance* for each of the 66 sweeps in the dataset. The presentation is analogous to that of the ETH-80 results, except for the fact that we now evaluate the posterior distributions after up to 10 views (the smallest sweep in the dataset). Note as well that we do not explicitly list the results of the M2 model anymore since we established in the previous section that it consistently produced the highest-entropy distributions over classes. The color-coding of all Figures shown on the following pages corresponds to those displayed in the legend below.

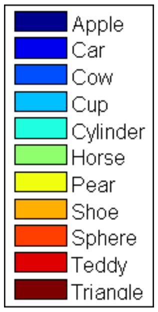


Figure 5-17: The color-coding for the objects in the Trisk-22 database.

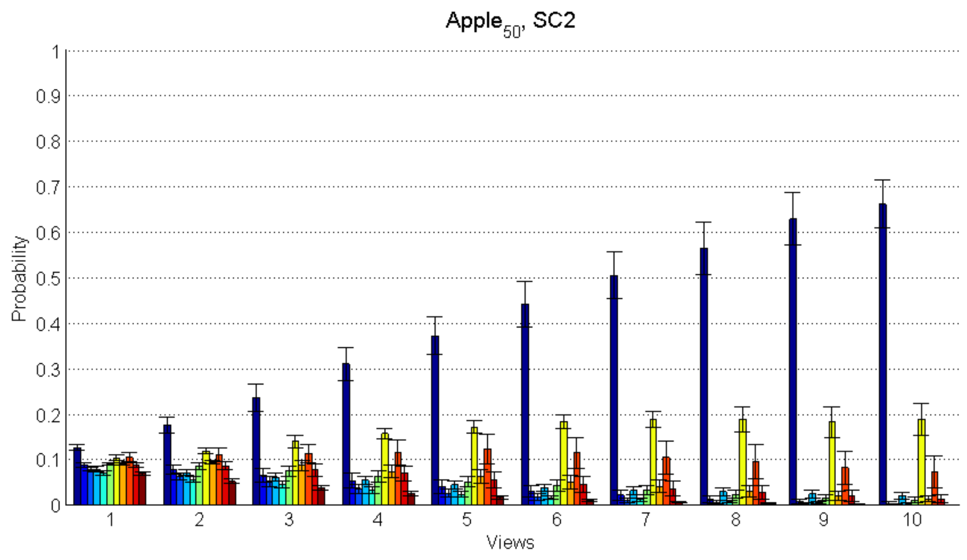
The main results of this experiment are summarized in Figures 5-18 to 5-21 for a number of classes in the Trisk-22 dataset. For each case, the lower of the two sets of distributions shows the improvement after including the explicit feature motion model as an additional source of information during classification. The following observations are possible about the results:

- Classification arising from the MAP estimate of the motion model $P(\mathcal{C}_k|\mathbf{x}_M)$ is generally *more discriminative* than for the ETH-80 dataset. One can assume that the reason for this lies with the denser sampling on the object contours due to the generally smaller image sizes. In addition to this *spatially* denser sampling, we also have a *temporally* much denser sampling of object views in every sweep which may add to the robustness of the motion model.
- As observed before, the joint SC+M2 model outperforms the SC2 baseline model significantly in terms of producing low-entropy posterior class distributions at an earlier point in time. This is more apparent with the Trisk-22 dataset than with the ETH-80 data: for many of the categories we achieve posterior distribution of similar quality already after 5 versus 8 object views.

For these and the remaining classes we summarize the important *classification* results in Figure 5-22. Here, our focus is on how many views are required to classify all sweeps in the Trisk-22 dataset correctly. Since the MAP class assignments do not differ between SC2 and SC+M2, we show the results for both classifiers in a single diagram.

5.3 Conclusion

In this chapter we presented an evaluation of all three classifier types suggested in this thesis. We established on two different datasets that feature motion (or "object dynamics") is a valid principle that can be exploited for object recognition of



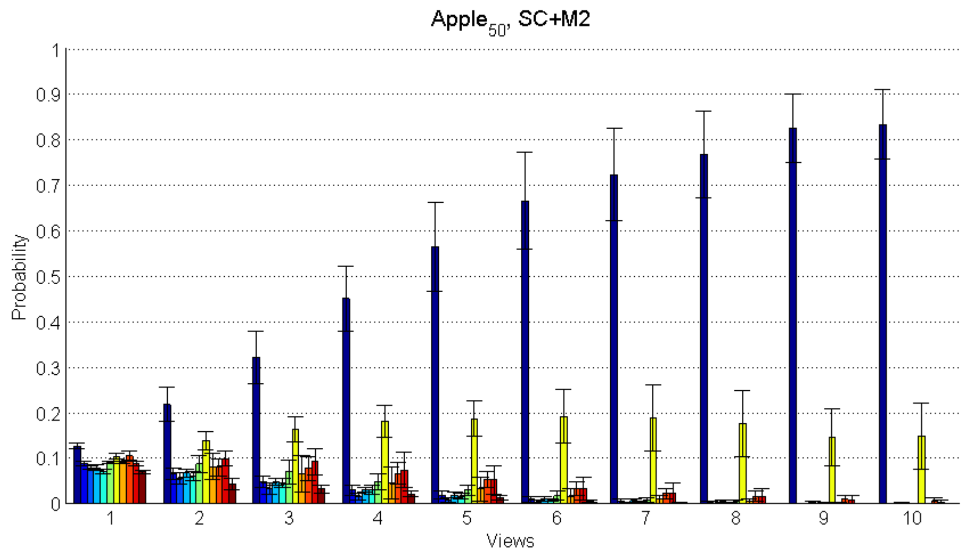
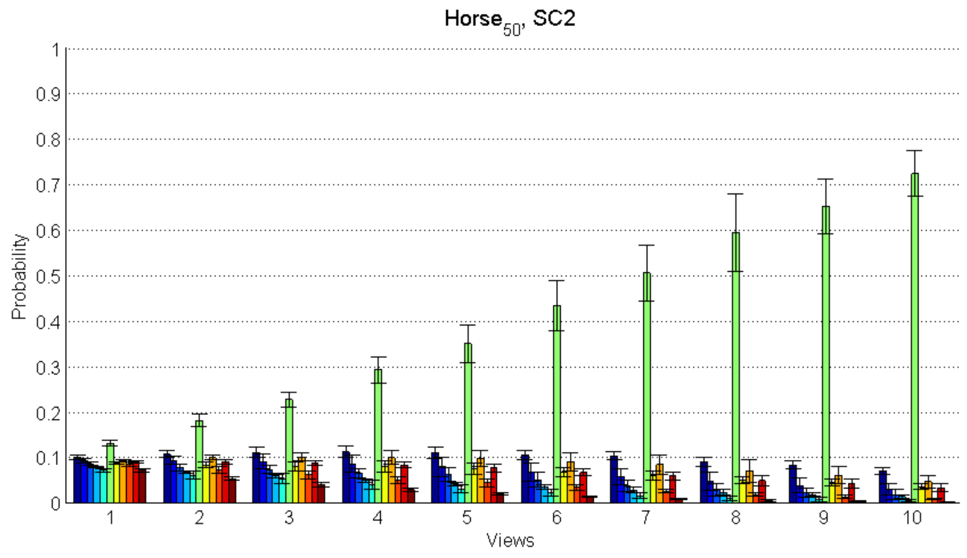


Figure 5-18: Mean posterior distributions after a specified number of *apple* views for SC2 and SC+M2 models.



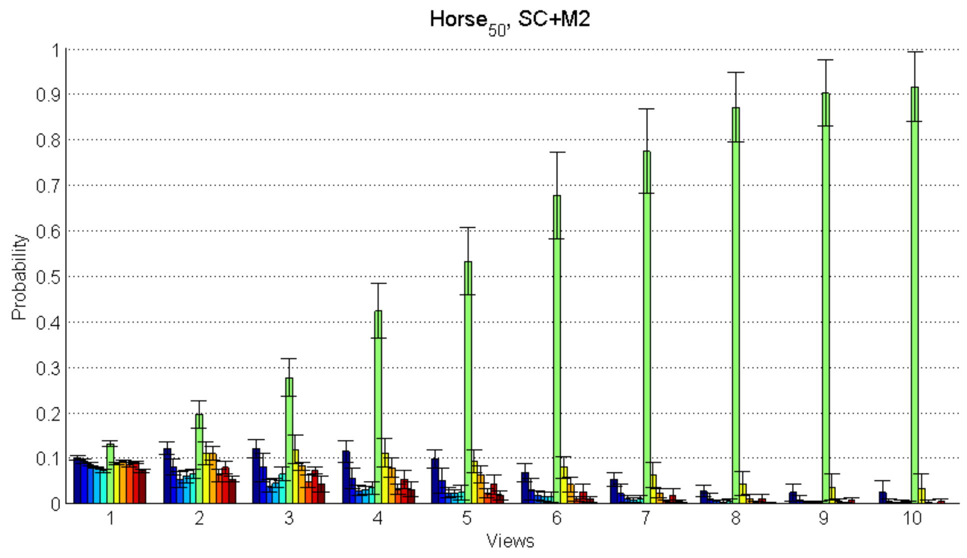
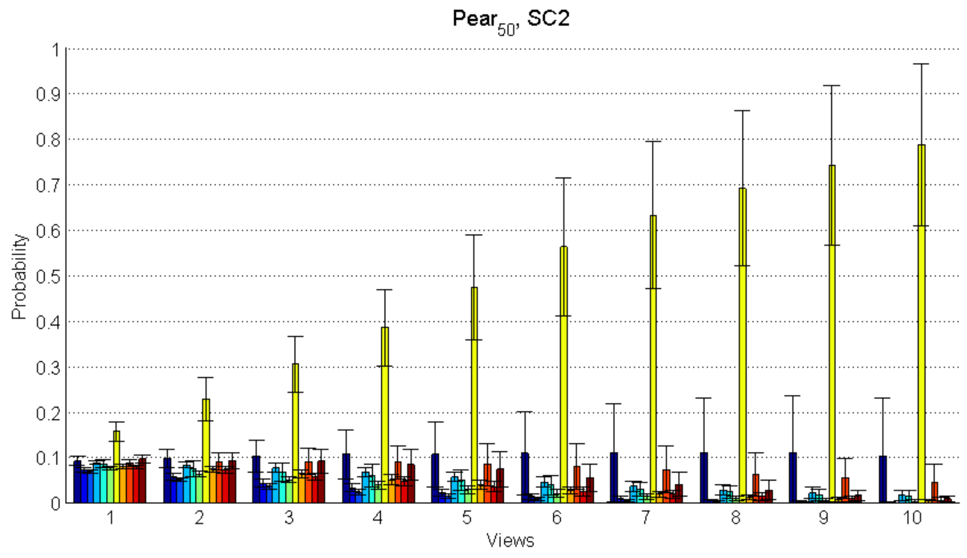


Figure 5-19: Mean posterior distributions after a specified number of *horse* views for SC2 and SC+M2 models.



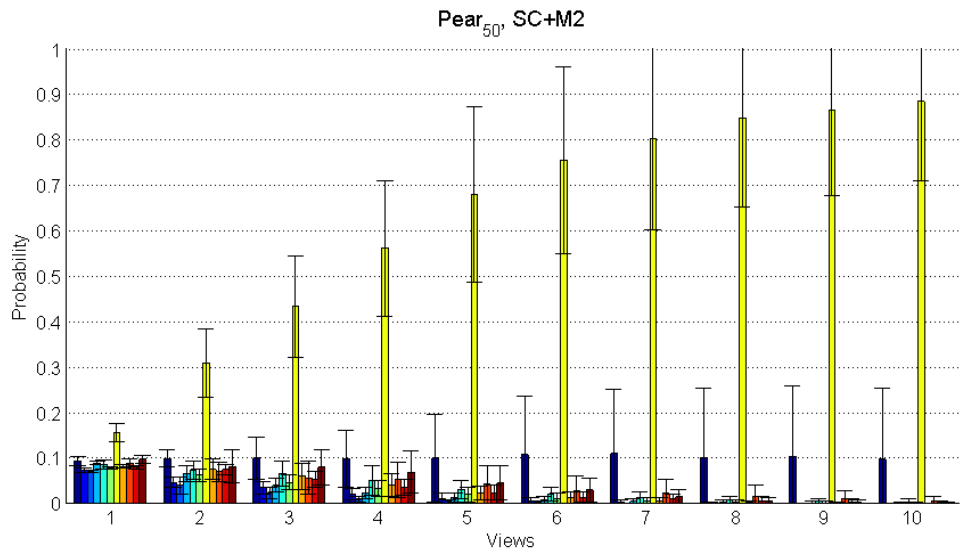
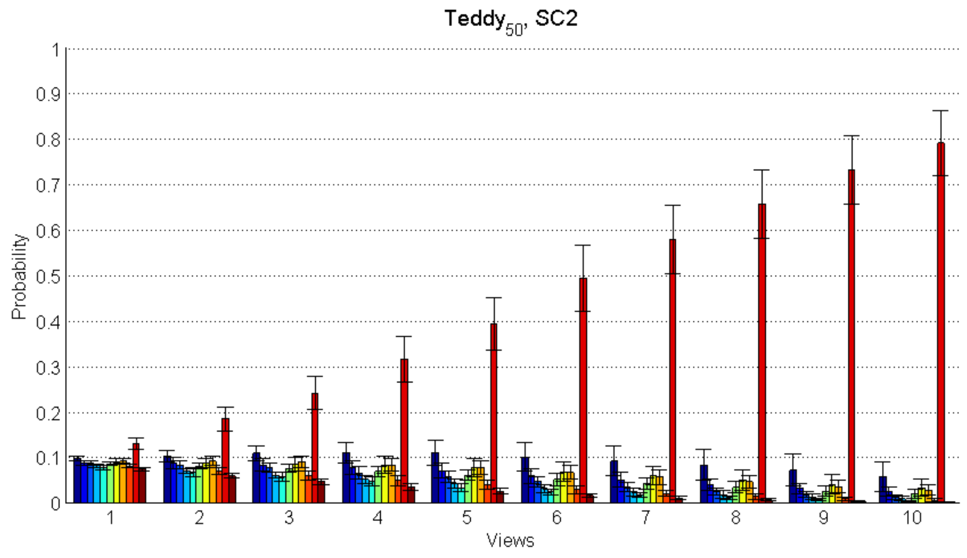


Figure 5-20: Mean posterior distributions after a specified number of *pear* views for SC2 and SC+M2 models.



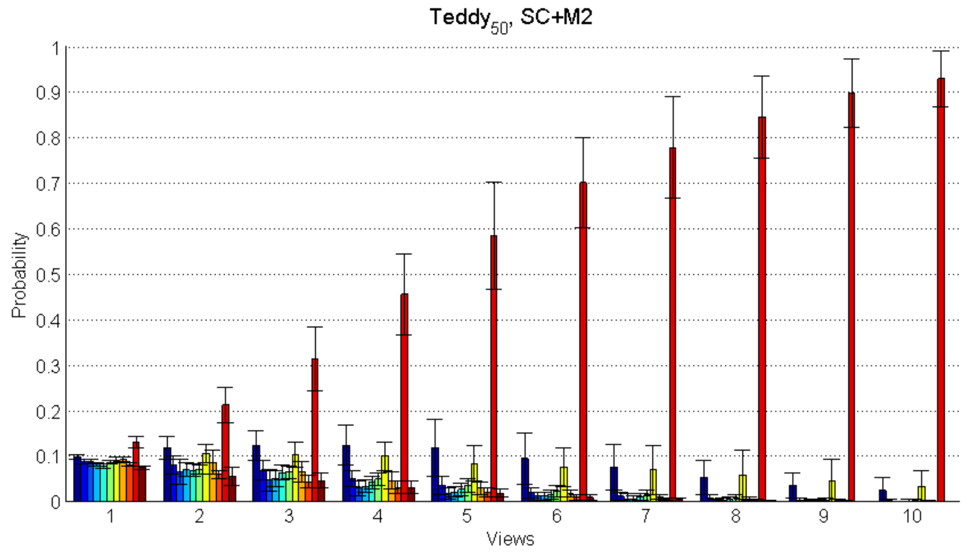


Figure 5-21: Mean posterior distributions after a specified number of *teddy* views for SC2 and SC+M2 models.

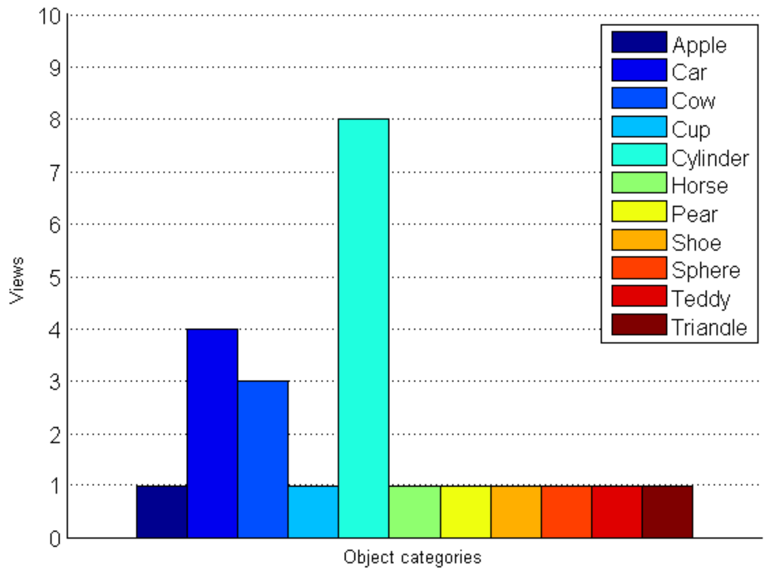


Figure 5-22: The number of required views to classify all sweeps in Trisk-22 correctly.

active vision systems. Different from our baseline model that essentially treats every image sequence as unordered, our joint model $P(\mathcal{C}_k|\mathbf{x}_{SC}, \mathbf{x}_M)$ makes use of the additional structural information revealed by the feature motion between frames about the object.

For the ETH-80 subset we observed that our revised model SC+M2 consistently outperformed the baseline SC2, resulting in posterior class distributions of lower entropy at earlier time steps. With our current choice of cost function (cf. section 3.2), the motion model was most successful at removing probability mass from those class hypotheses corresponding to clearly different objects. For similar objects, on the other hand, the motion model M2 alone did not always produce as discriminative results but was still clearly beneficial as part of the joint SC+M2 formulation (see, e.g., the cow case in Figure 5-13). Our evaluation of the active view selection strategy demonstrated the potential for further reduction in posterior entropy on the same dataset.

We found that the inclusion of a motion model yielded even more substantial increases in quality of the posterior class distribution for our real-world Trisk-22 dataset. We suggested that this may be in part due to the finer sampling along the object contours resulting from the smaller image sizes. To verify this point, we computed our motion model M2 on the same ETH-80 dataset with 100 contour samples instead of 50. The corresponding Figures are produced as A-9 to A-16 in Appendix A and show a general improvement of the motion model, as suggested.

In summary, feature motion-based classification appears to be a valid addition to an active recognition system built around invariant local features. While it certainly does not replace local feature-based matching in the presented form, we could demonstrate that for different target measures (such as a fixed goal entropy θ_2 or a target probability threshold for the winning class) our joint model SC+M2 improves on the baseline SC2 significantly in the majority of cases.

Chapter 6

Contributions and Future Work

In this thesis we looked at how state-of-the-art object recognition methods can benefit from feature motion information computed over successive object views. For active vision systems that additionally associate camera control parameters with each recorded view, we furthermore evaluated the effectiveness of a view selection strategy that steers the camera to distinctive points on the view sphere. The key rationale for our investigation was that our robot domain demands accurate object classification under stringent time constraints, usually posed by an interactivity requirement for interactions between human and robot. Our review of the object recognition literature showed that much of the current work is on recognizing object categories from fixed viewpoints in natural scenes under little or no such time constraints. The research presented in this thesis, on the other hand, addresses our specific requirements by introducing the novel concept of inter-frame feature motion as an additional source of information for disambiguating between objects.

In the course of this thesis we derived and implemented a number of probabilistic classifiers based on the shape context algorithm (SC, SC2), feature motion information (M, M2), both shape and feature motion (SC+M, SC+M2) and on an active view selection strategy. We then used a distinct set of camera trajectories to test our sweep-based recognition methods on two datasets, the first being a subset of the standard ETH-80 object database and the second a custom dataset recorded directly from our robot's camera. While the former of the presented recognition models trace a camera trajectory until an entropy threshold of the posterior class distribution is reached, the latter model parts with fixed camera trajectories and skips views that are deemed uninformative.

We could demonstrate for both datasets that incorporating feature motion or active view selection achieves a higher-quality hypothesis about the category in faster time. In particular for the real-world database did the joint model considerably improve on the individual models SC2 and M2.

During our experiments we also experienced the limits of contour-only based classifiers. As noted in the thesis, the feature motion principle is compatible with all local features that allow to establish correspondence between successive frames. Besides this possible future extension, one could also imagine other additions to our work. First, there currently exists the drawback that training and test trajectories have to start at the same point. If, as is the case for the ETH-80 data, we record "closed" sweeps, one could instead maintain multiple hypotheses about the starting point and the associated sweep probabilities. This requires an initial match into the entire object database instead of only the views associated with the first time step.

Second, it is feasible to explore whether clustering in the feature evolution space would reveal part structures. The driving motivation of this is for the robot to answer whether a particular object has, for example, a handle after having executed a sweep over the object.

Even in its current form, however, we believe to have contributed a reliable object recognition system that may particularly be useful for robotic or otherwise time constrained vision systems just as Trisk.

Appendix A

Supplementary Figures

A.1 ETH-80

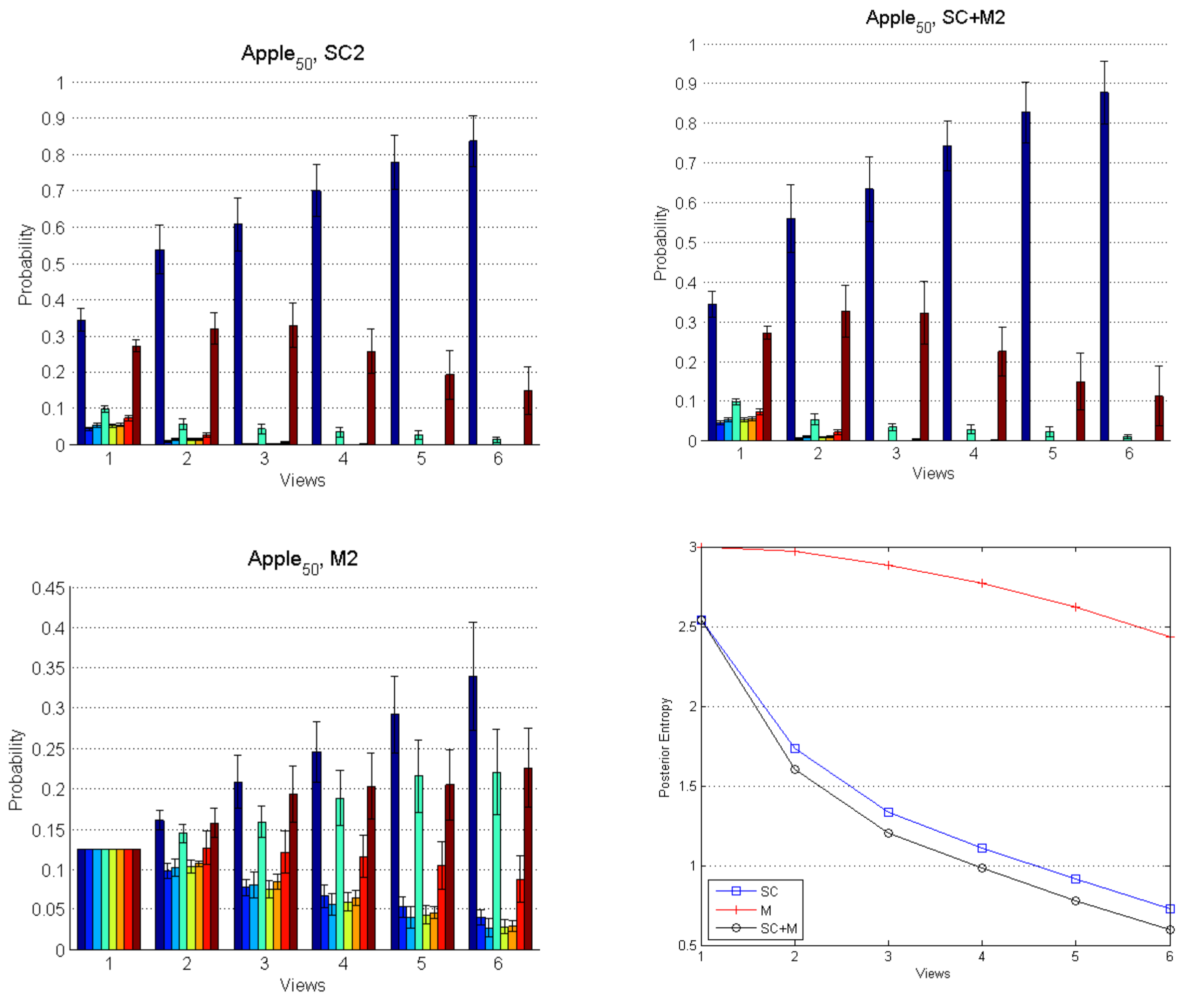


Figure A-1: Mean posterior distributions after a specified number of *apple* views for SC2, SC+M2, and M2 models (50 contour samples). The entropy of the distributions is shown on the bottom right.

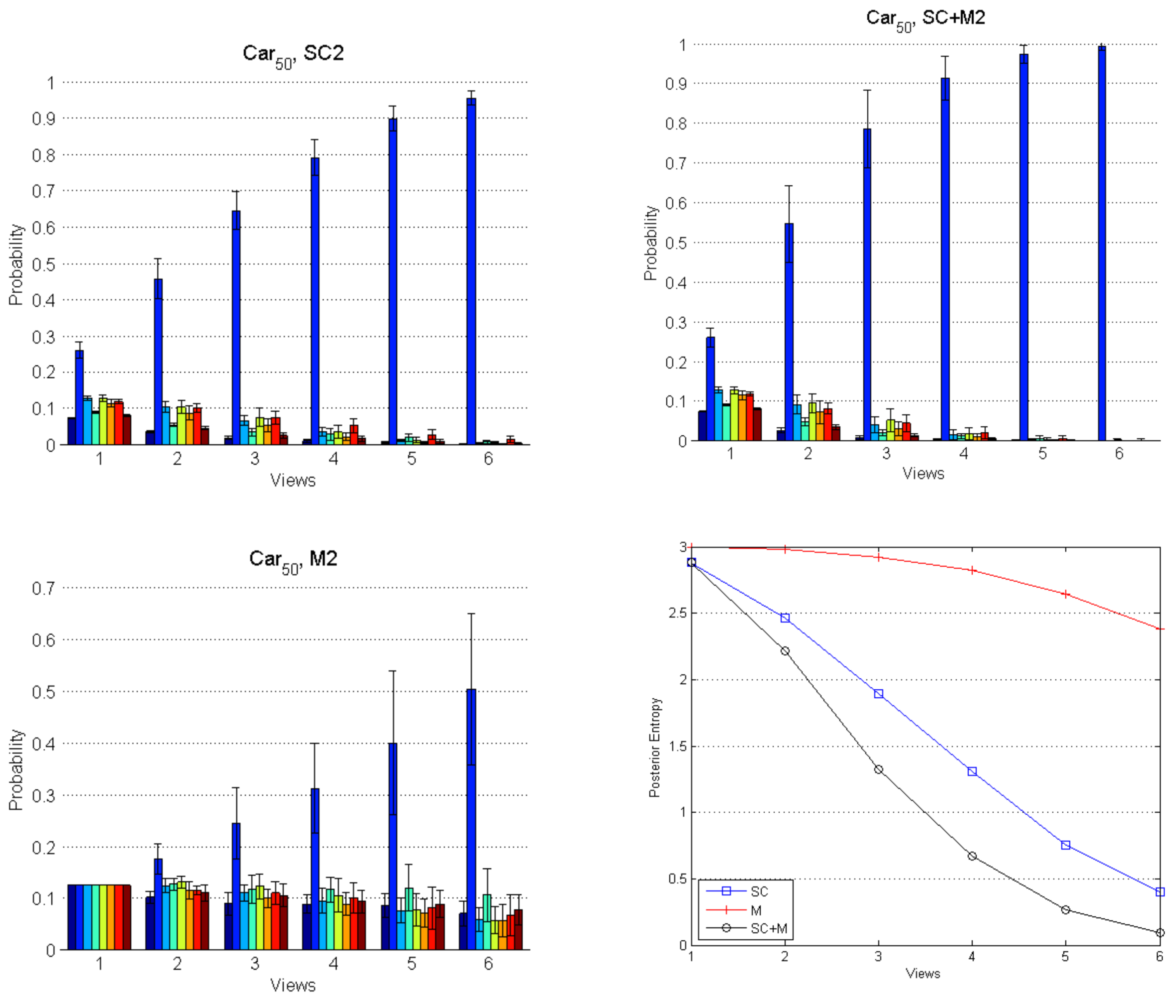


Figure A-2: Mean posterior distributions after a specified number of *car* views for SC2, SC+M2, and M2 models (50 contour samples). The entropy of the distributions is shown on the bottom right.

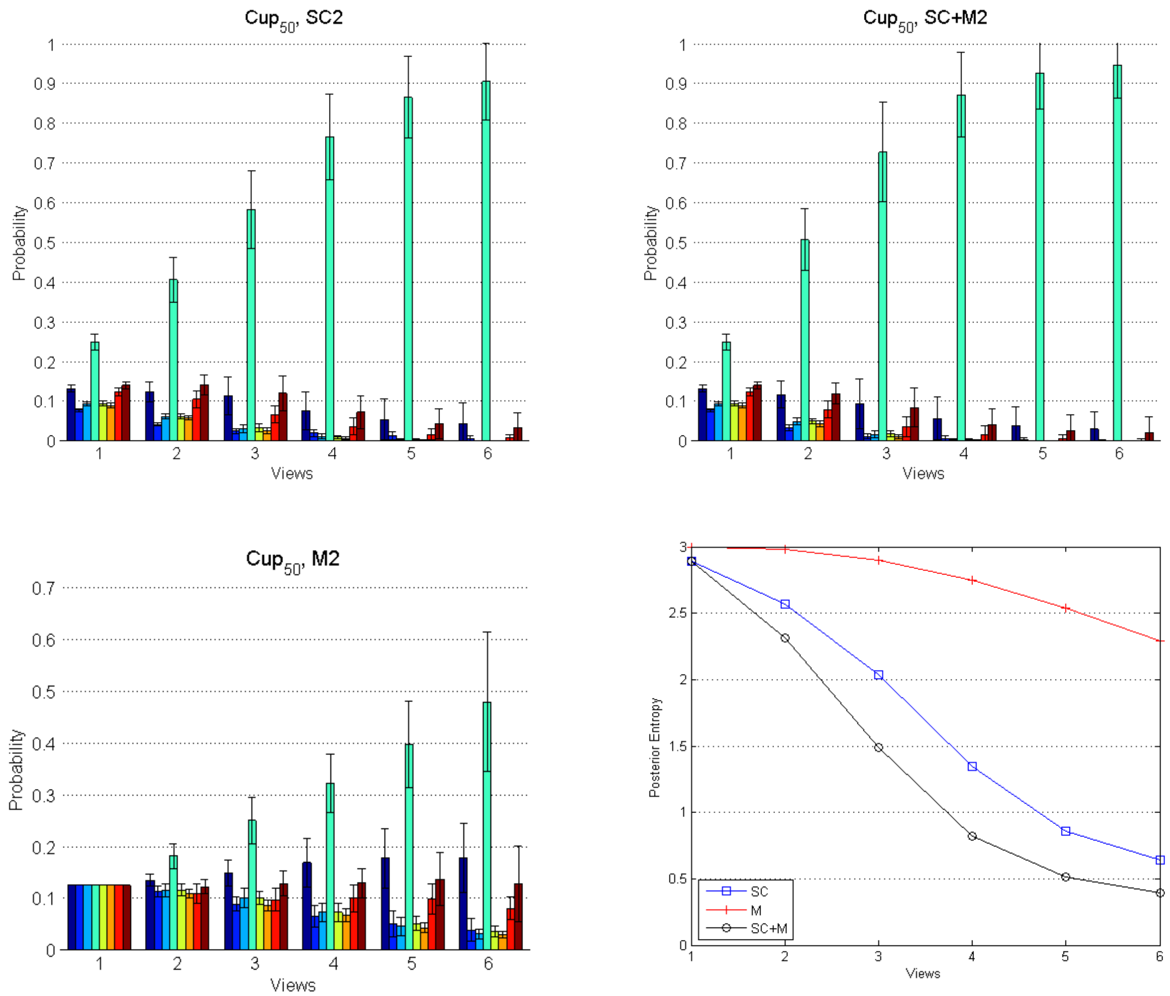


Figure A-3: Mean posterior distributions after a specified number of *cup* views for SC2, SC+M2, and M2 models (50 contour samples). The entropy of the distributions is shown on the bottom right.

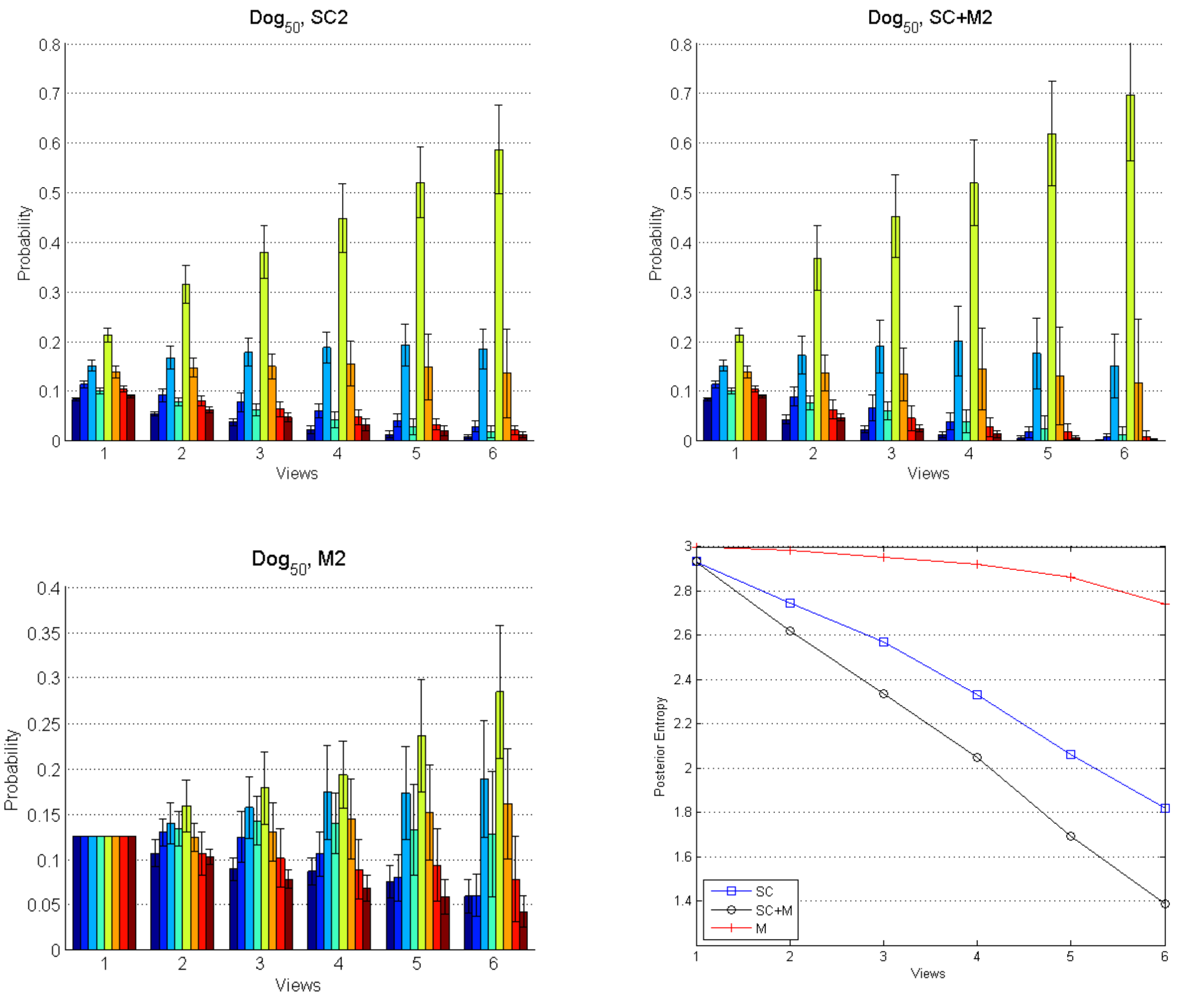


Figure A-4: Mean posterior distributions after a specified number of *dog* views for SC2, SC+M2, and M2 models (50 contour samples). The entropy of the distributions is shown on the bottom right.

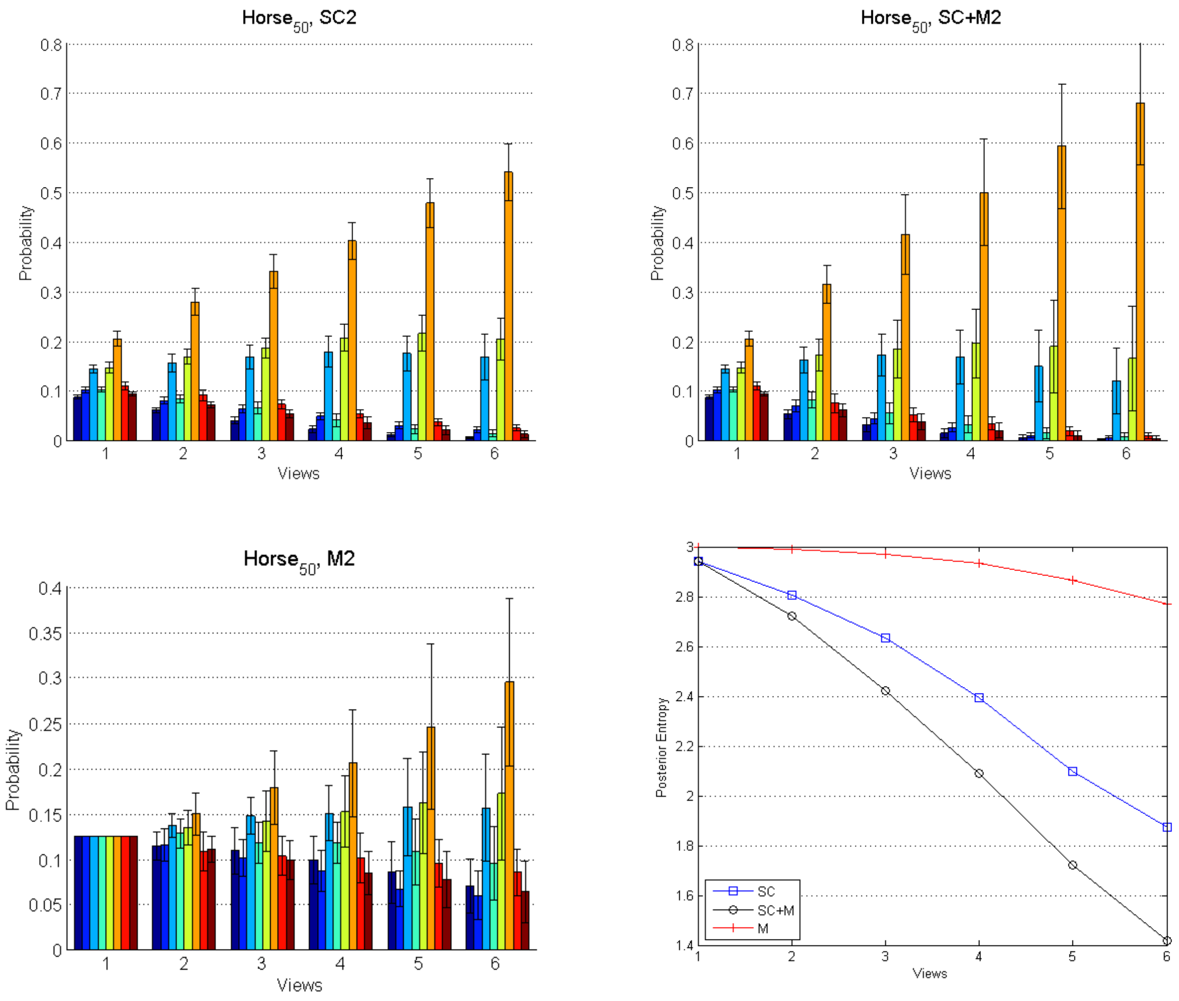


Figure A-5: Mean posterior distributions after a specified number of *horse* views for SC2, SC+M2, and M2 models (50 contour samples). The entropy of the distributions is shown on the bottom right.

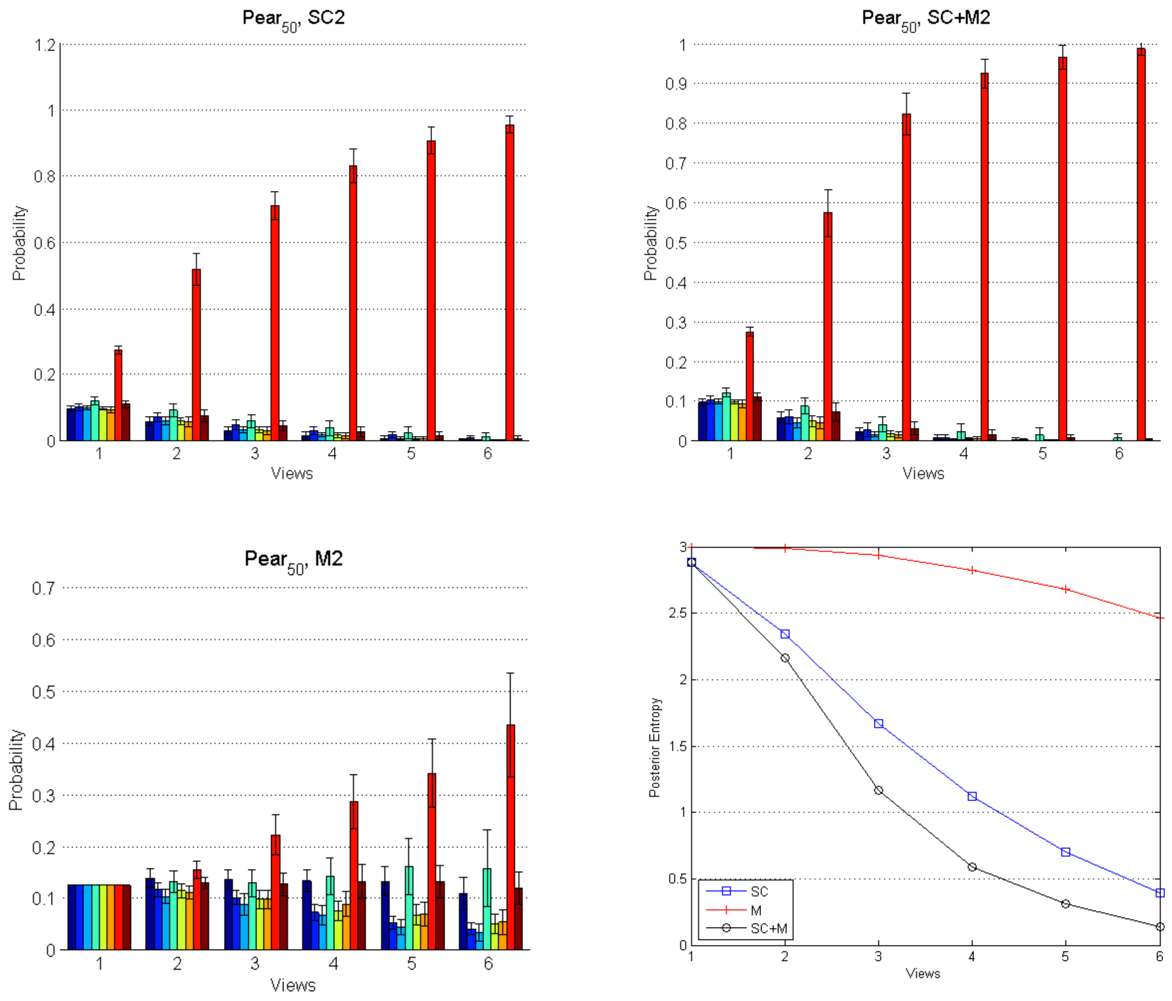


Figure A-6: Mean posterior distributions after a specified number of *pear* views for SC2, SC+M2, and M2 models (50 contour samples). The entropy of the distributions is shown on the bottom right.

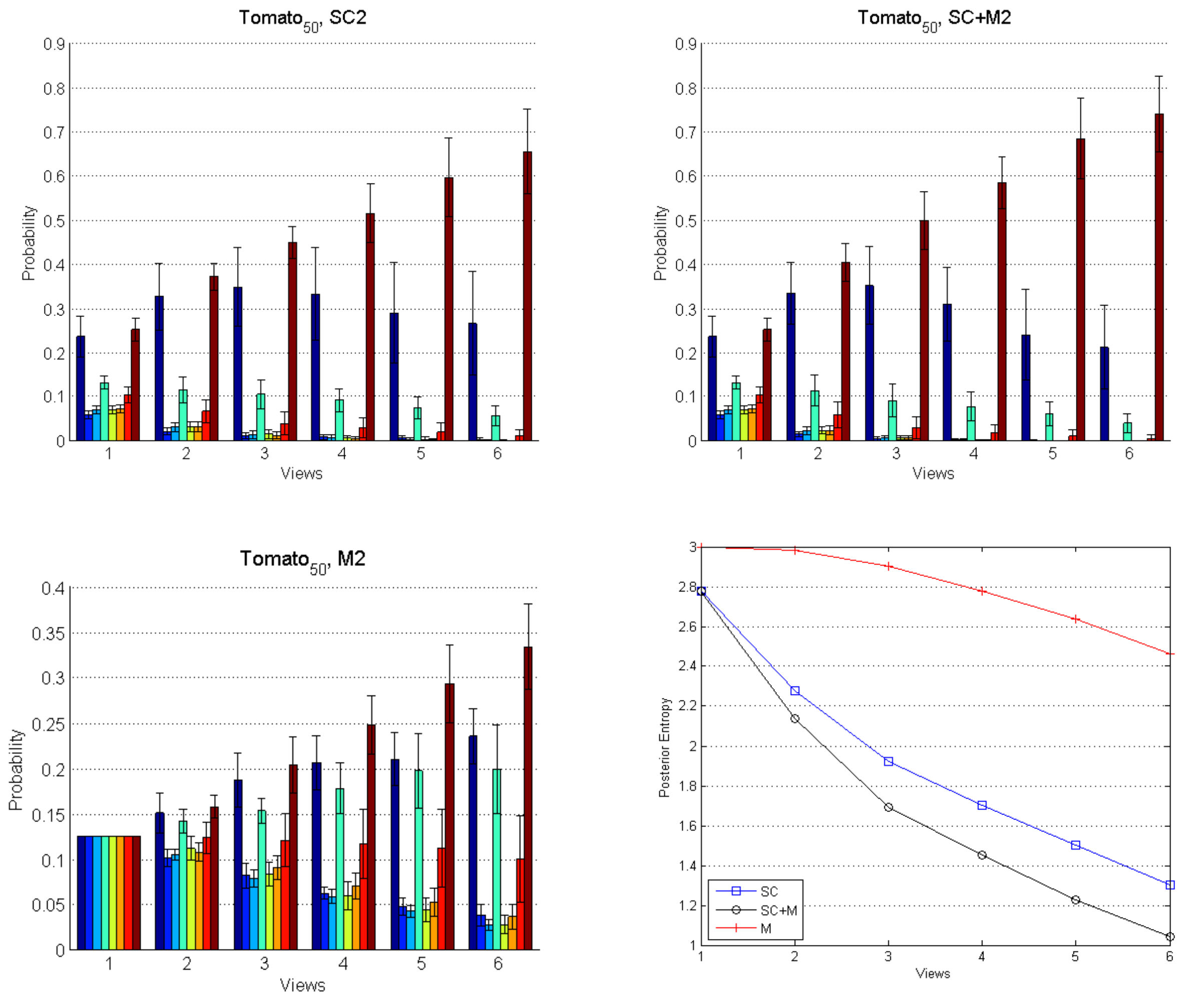


Figure A-7: Mean posterior distributions after a specified number of *tomato* views for SC2, SC+M2, and M2 models (50 contour samples). The entropy of the distributions is shown on the bottom right.

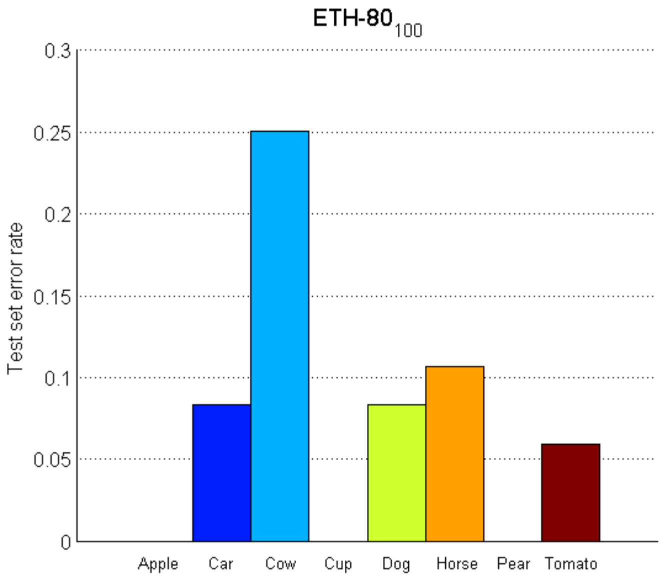


Figure A-8: The test set error rate for one-shot shape context matching (100 point samples along the contour).

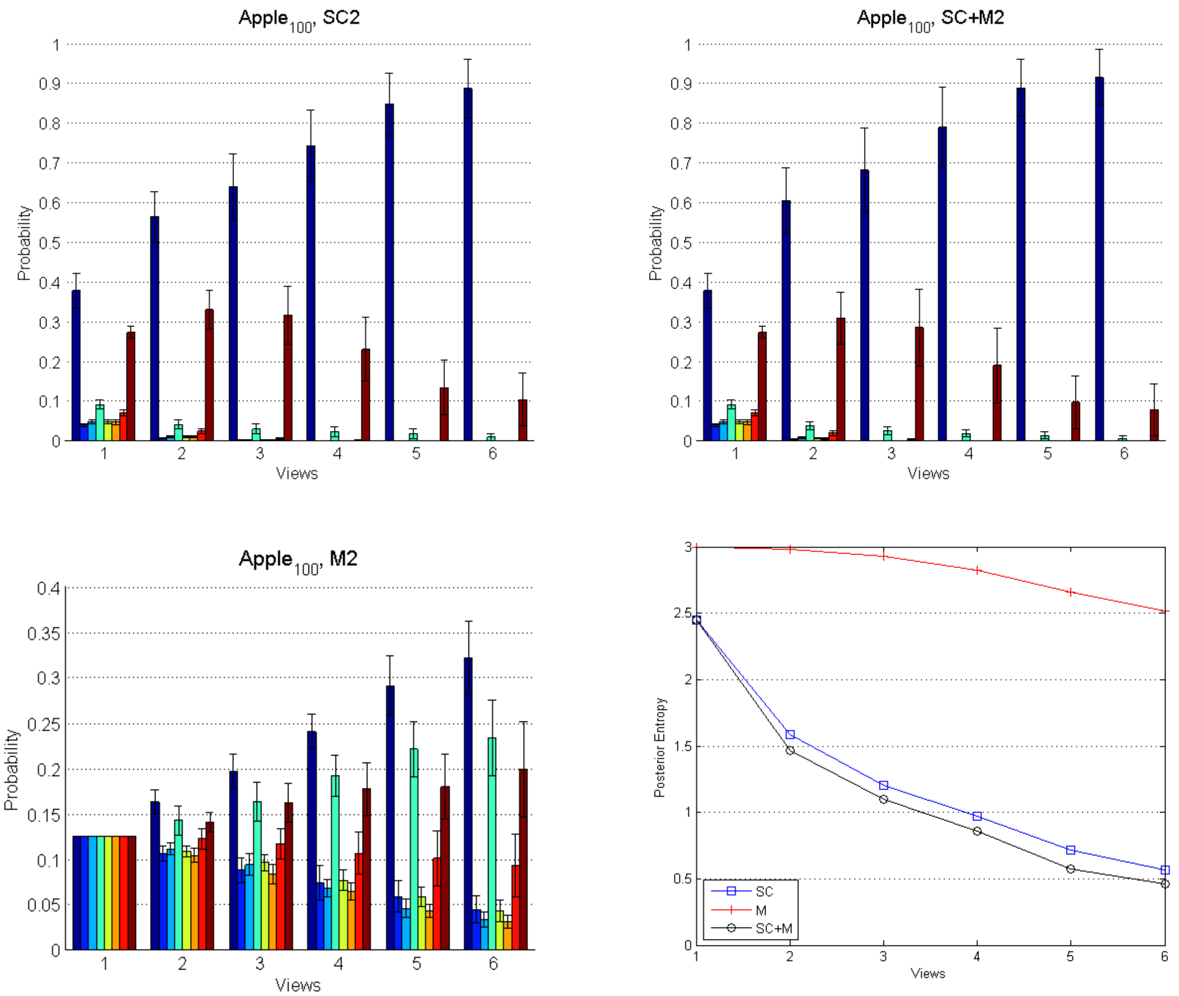


Figure A-9: Mean posterior distributions after a specified number of *apple* views for SC2, SC+M2, and M2 models (100 contour samples). The entropy of the distributions is shown on the bottom right.

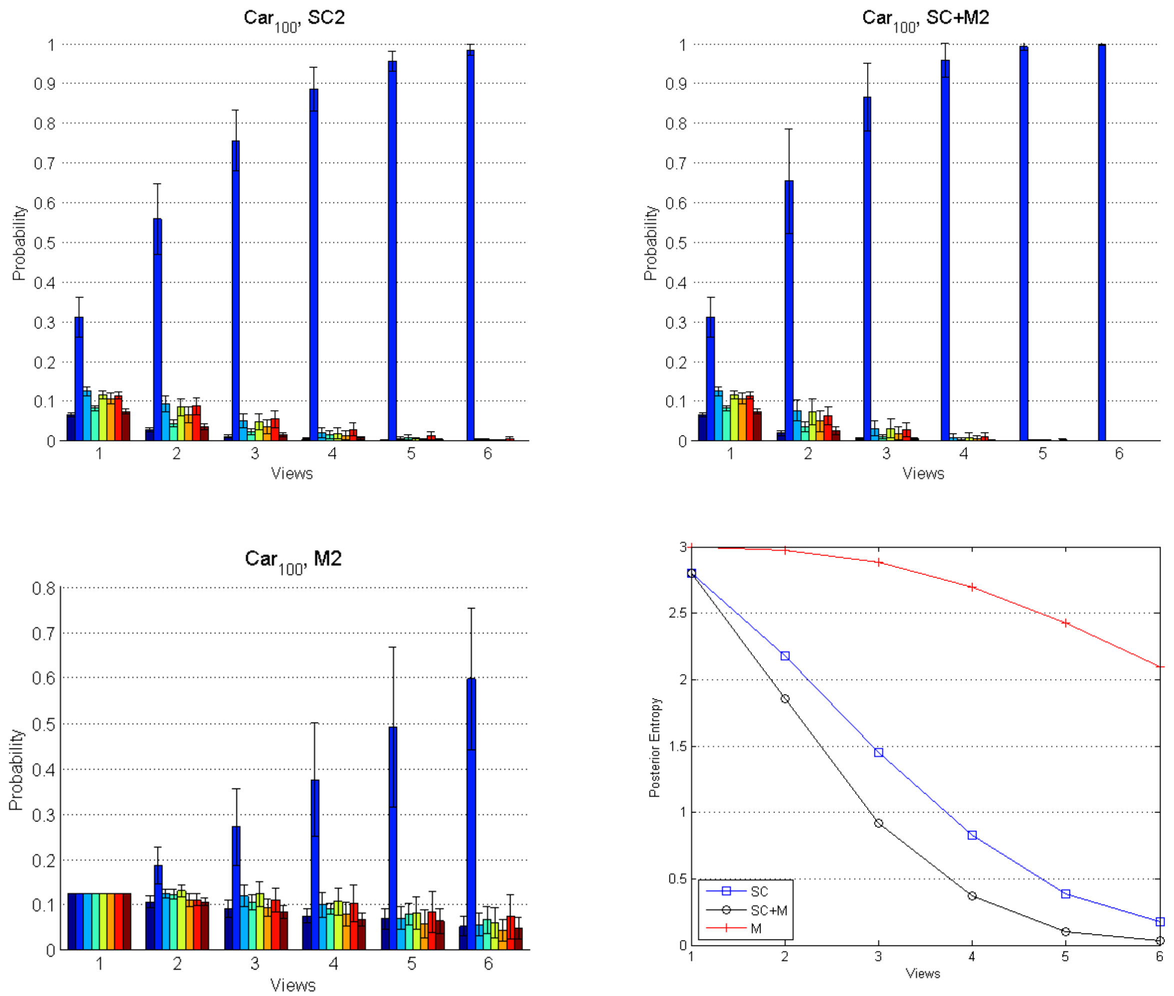


Figure A-10: Mean posterior distributions after a specified number of *car* views for SC2, SC+M2, and M2 models (100 contour samples). The entropy of the distributions is shown on the bottom right.

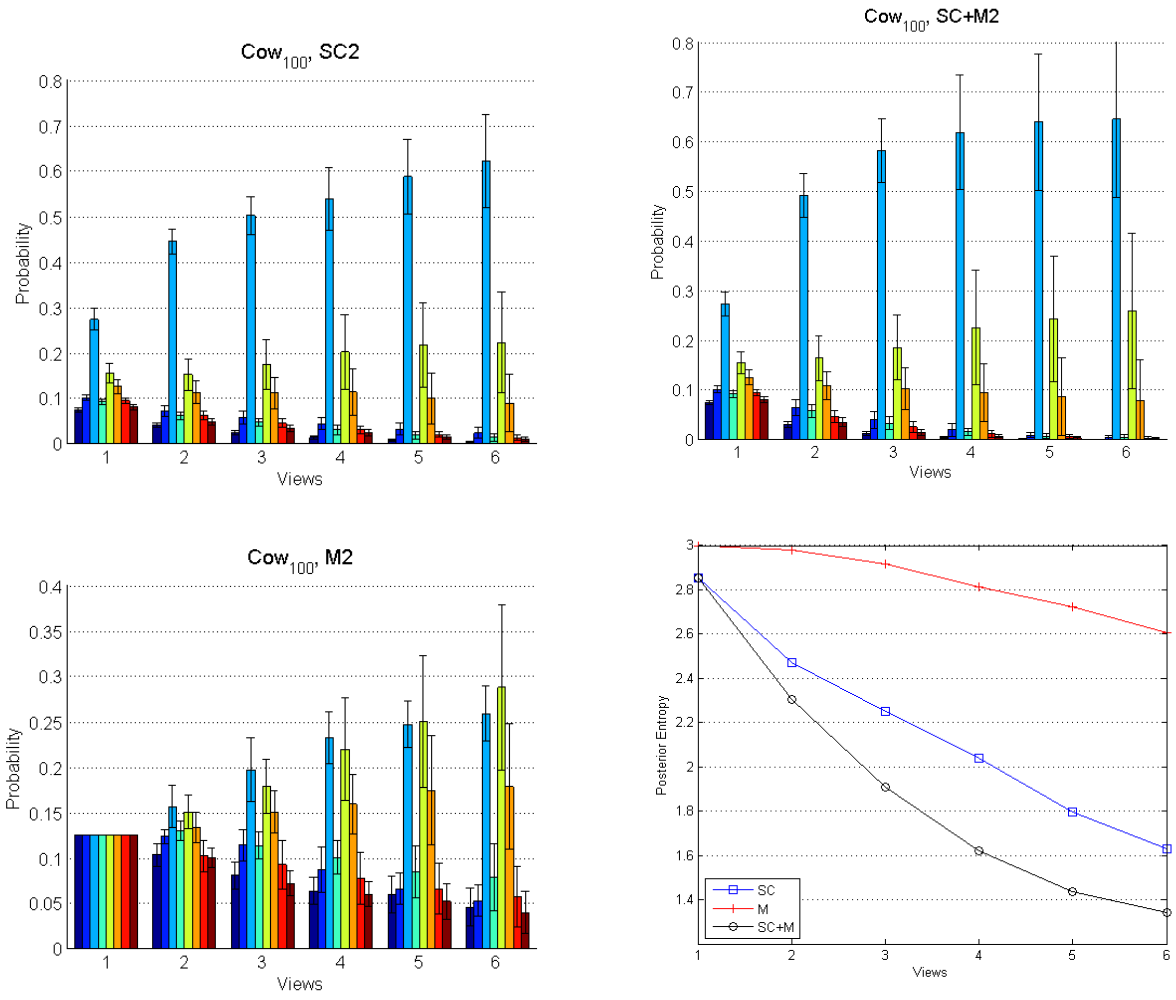


Figure A-11: Mean posterior distributions after a specified number of *cow* views for SC2, SC+M2, and M2 models (100 contour samples). The entropy of the distributions is shown on the bottom right.

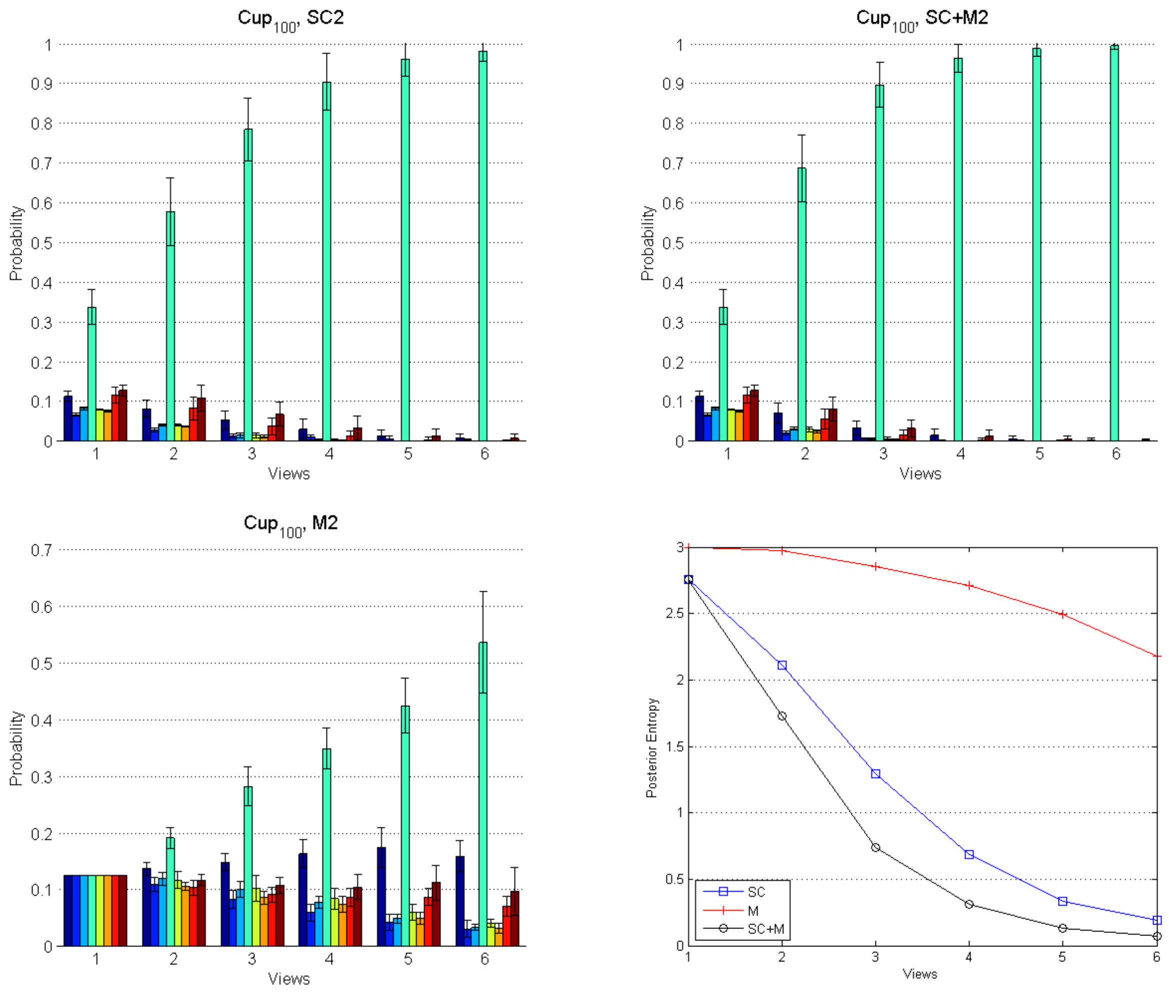


Figure A-12: Mean posterior distributions after a specified number of *cup* views for SC2, SC+M2, and M2 models (100 contour samples). The entropy of the distributions is shown on the bottom right.

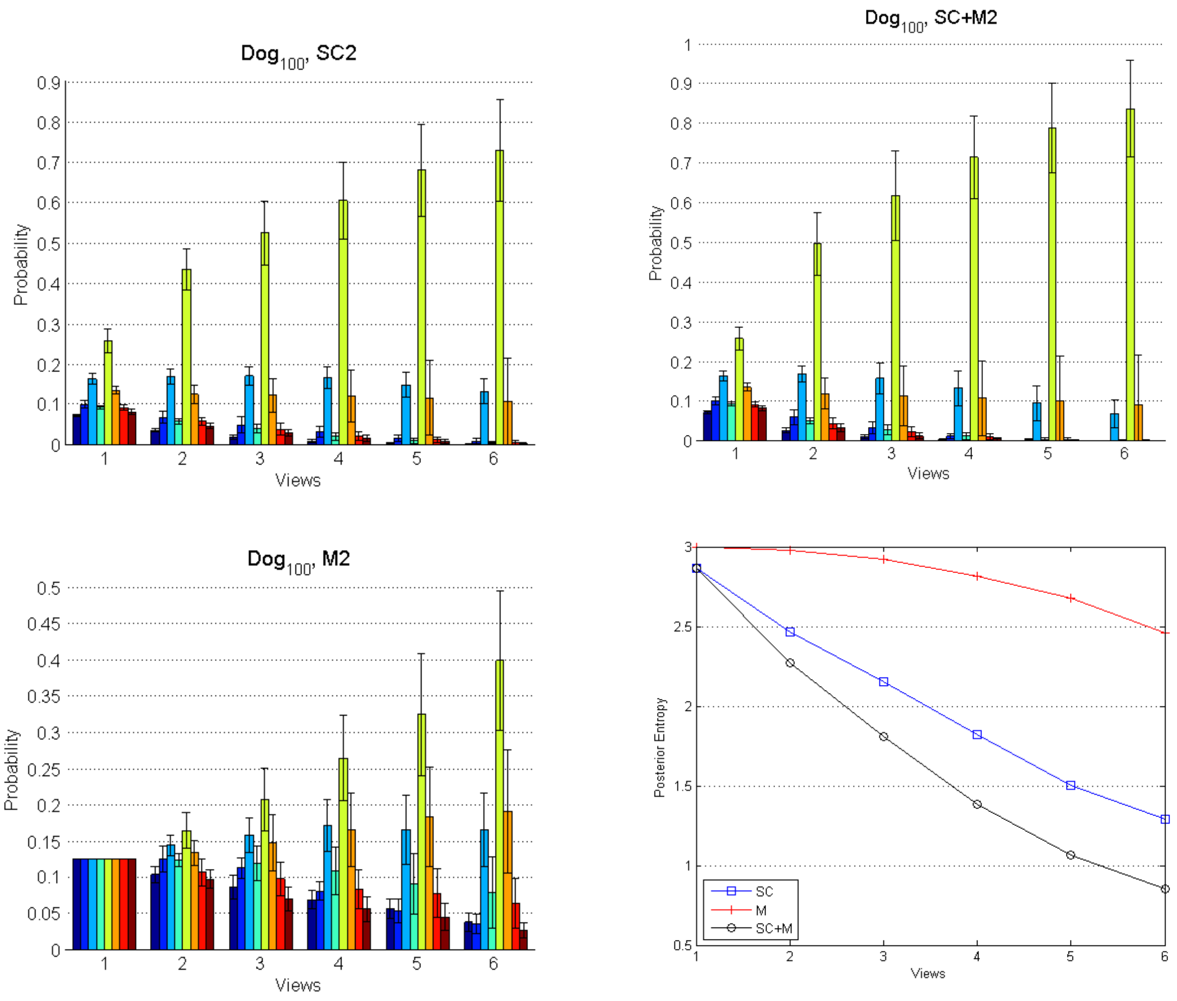


Figure A-13: Mean posterior distributions after a specified number of *dog* views for SC2, SC+M2, and M2 models (100 contour samples). The entropy of the distributions is shown on the bottom right.

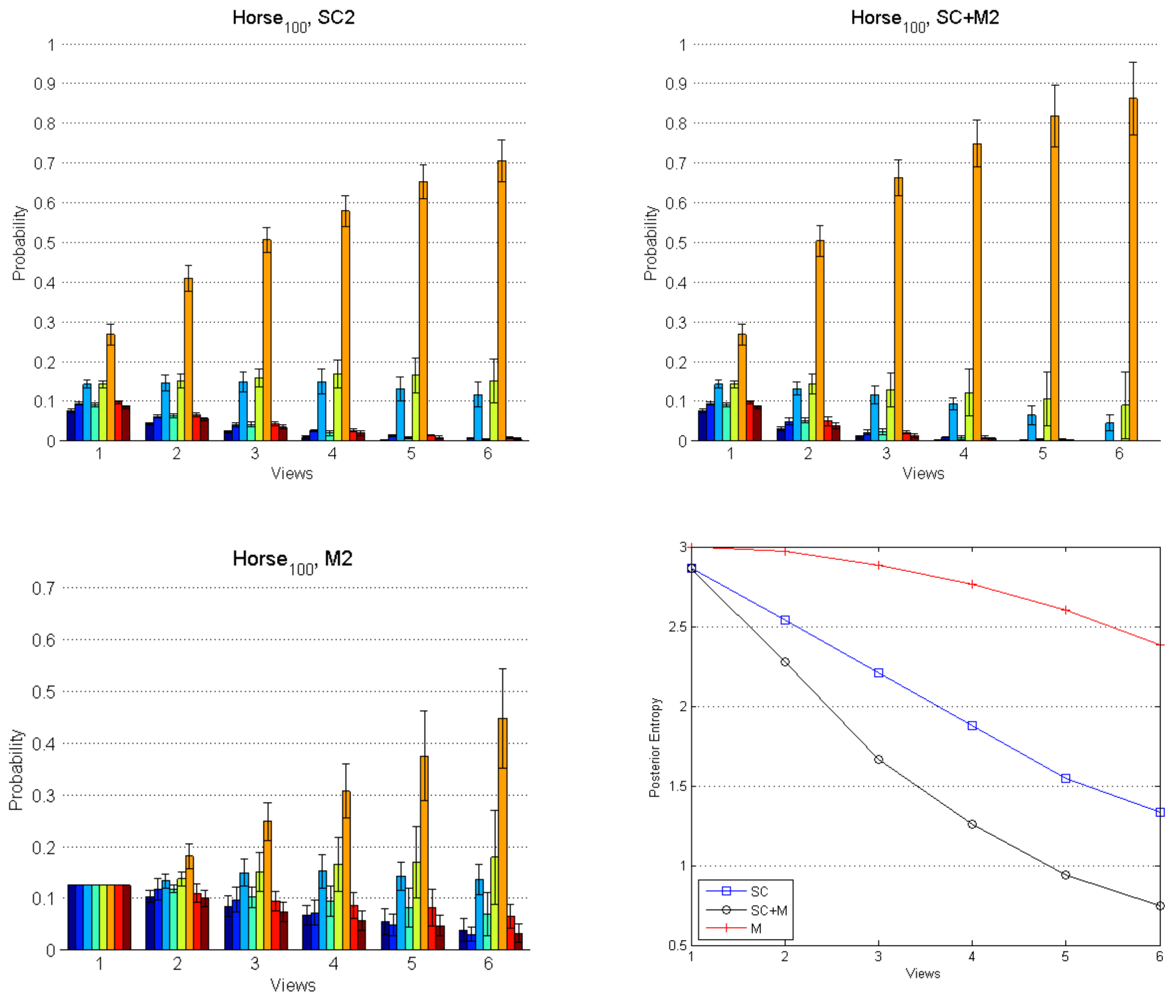


Figure A-14: Mean posterior distributions after a specified number of *horse* views for SC2, SC+M2, and M2 models (100 contour samples). The entropy of the distributions is shown on the bottom right.

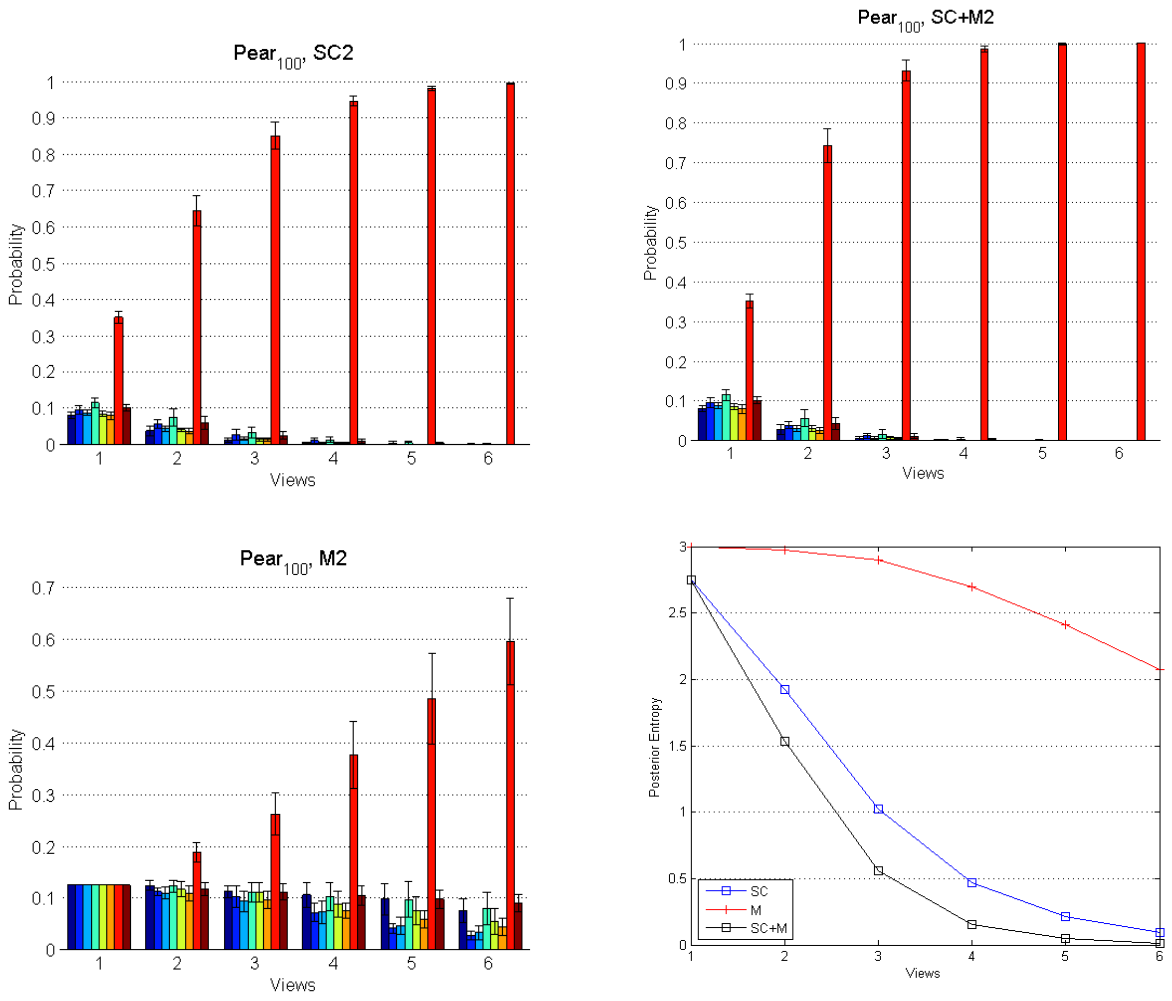


Figure A-15: Mean posterior distributions after a specified number of *pear* views for SC2, SC+M2, and M2 models (100 contour samples). The entropy of the distributions is shown on the bottom right.

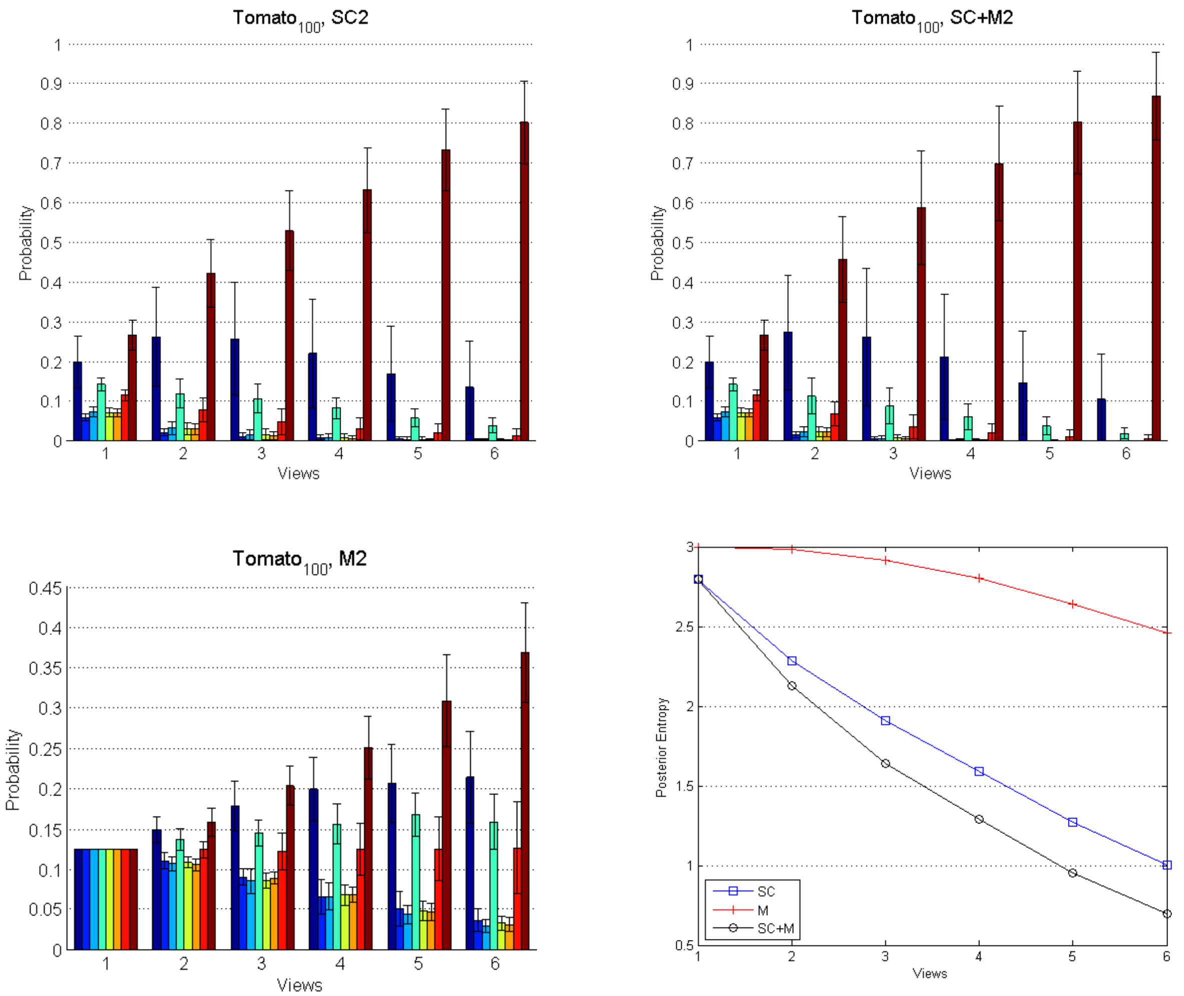


Figure A-16: Mean posterior distributions after a specified number of *tomato* views for SC2, SC+M2, and M2 models (100 contour samples). The entropy of the distributions is shown on the bottom right.

Bibliography

[1] Shivani Agarwal and Dan Roth. Learning a sparse representation for object detection. In *ECCV*, pages 113–130, London, UK, 2002. Springer-Verlag.

[2] T. Arbel and F. P. Ferrie. Entropy-based gaze planning. *IVC*, 19(11):779–786, September 2001.

[3] T. Arbel, F. P. Ferrie, and M. Mitran. Recognizing objects from curvilinear motion. In *BMVC*, 2000.

[4] B. Leibe B. and Schiele. Analyzing appearance and contour based methods for object categorization. In *CVPR*, pages II: 409–415, 2003.

[5] S. Belongie, J. Malik, and J. Puzicha. Shape context: A new descriptor for shape matching and object recognition. In *NIPS*, pages 831–837, 2000.

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

[7] Irving Bierderman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94(2):115–147, 1987.

[8] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, August 2006.

[9] V. Blanz, M.J. Tarr, and H.H. Blthoff. What object attributes determine canonical views? *Perception*, 28:575–599, 1999.

[10] Robert Bolles and Radu P. Horaud. 3dpo: A three dimensional part orientation system. *International Journal of Robotics Research*, 5(3):3–26, Fall 1986.

[11] F. L. Bookstein. Principal warps: thin-plate splines and the decomposition of deformations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(6):567–585, 1989.

[12] R. A. Brooks. Symbolic reasoning among 3-d models and 2-d images. *AI*, 17(1-3):285–348, August 1981.

[13] C. Cédras and M. A. Shah. Motion based recognition: A survey. *Image and Vision Computing*, 13(2):129–155, March 1995.

[14] F. Cutzu and S. Edelman. Canonical views in object representation and recognition. *Vision Research*, 34:3037–3056, 1994.

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

[16] J. W. Davis and A. F. Bobick. The representation and recognition of action using temporal templates. In *CVPR*, pages 928–934, 1997.

[17] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *Int. J. Comput. Vision*, 61(1):55–79, 2005.

[18] R. Fergus. *Visual Object Category Recognition.* PhD thesis, University of Oxford, 2005.

[19] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 264–271, June 2003.

[20] M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structures. *Transactions on Computers*, C-22(1):67–92, 1973.

[21] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach.* Prentice Hall Series in Artificial Intelligence. Prentice Hall, 2003.

[22] W. E. L. Grimson and T. Lozano Perez. Localizing overlapping parts by searching the interpretation tree. *PAMI*, 9(4):469–482, July 1987.

[23] K. Ikeuchi and T. Kanade. Automatic generation of object recognition programs. *Proceedings of the IEEE*, 76(8):1016–1035, August 1988.

[24] S. Ioffe and D. Forsyth. Human tracking with mixtures of trees. In *International Conference on Computer Vision*, volume 1, pages 690–695, July 2001.

[25] T. Jebara, A. Azarbayejani, and A. P. Pentland. 3d structure from 2d motion. *SPMag*, 16(3):66–84, May 1999.

[26] J. J. Koenderink and A. J. van Doorn. The internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32:211–216, 1979.

[27] B. Leibe and B. Schiele. Interleaved object categorization and segmentation. In *British Machine Vision Conference (BMVC'03)*, pages 759–768, Norwich, UK, Sept. 2003.

[28] D. G. Lowe. Local feature view clustering for 3d object recognition. *CVPR*, 1:682–688, 2001.

[29] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[30] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. *Eighth IEEE International Conference on Computer Vision (ICCV)*, 1:525–531, 2001.

[31] J. L. Mundy. Object recognition in the geometric era: A retrospective. In J. Ponce, M. Hebert, C. Schmid, and A. Zisserman, editors, *Toward Category-Level Object Recognition*, volume 4170 of *Lecture Notes in Computer Science*, pages 3–29. Springer, 2006.

[32] H. Murase and S. K. Nayar. Visual learning and recognition of 3-d objects from appearance. *Int. J. Comput. Vision*, 14(1):5–24, 1995.

[33] R. Nag, K. H. Wong, and F. Fallside. Script recognition using hidden markov models. In *ICASSP*, pages 2071–2074, 1986.

[34] S. E. Palmer, E. Rosch, and P. Chase. Canonical perspective and the perception of objects. In J. Long and A. Baddeley, editors, *Attention and Performance*, volume 9, pages 135–151. Erlbaum, Hillsdale, NJ, 1981.

[35] J. Ponce. Toward true 3d object recognition. In *CVPR Workshop on Generic Object Recognition and Categorization*. IEEE Computer Society, 2004.

[36] J. Ponce, T. L. Berg, M. Everingham, D. Forsyth, M. Hebert, S. Lazebnik, M. Marszałek, C. Schmid, C. Russell, A. Torralba, C. Williams, J. Zhang, and A. Zisserman. Dataset issues in object recognition. In *Towards Category-Level Object Recognition*. Springer, 2006.

[37] M. Pontil and A. Verri. Support vector machines for 3d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(6):637–646, 1998.

[38] M. Riesenhuber and T. Poggio. Models of object recognition. *Nature Neuroscience*, 3:1199 – 1204, 2000.

[39] L. G. Roberts. *Machine perception of three-dimensional solids.* PhD thesis, Massachusetts Institute of Technology. Dept. of Electrical Engineering, 1963.

[40] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3d object modeling and recognition using affine-invariant patches and multi-view spatial constraints. *CVPR*, 2:272–277, 2003.

[41] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *Int. J. Comput. Vision*, 66(3):231–259, 2006.

[42] C. A. Rothwell, A. Zisserman, J. L. Mundy, and D. A. Forsyth. Efficient model library access by projectively invariant indexing functions. In *Proceedings 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 109–114, Champaign, Illinois, 15–18 June 1993. IEEE Computer Society Press.

[43] L. K. Saul and S. T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *J. Mach. Learn. Res.*, 4:119–155, 2003.

[44] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *Int. J. Comput. Vision*, 37(2):151–172, 2000.

[45] J. Shi and C. Tomasi. Good features to track. In *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 593 – 600, 1994.

[46] T. E. Starner, J. Weaver, and A. P. Pentland. Real-time american sign language recognition using desk and wearable computer based video. *PAMI*, 20(12):1371–1375, December 1998.

[47] D. W. Thompson and J. L. Mundy. Three dimensional model matching from an unconstrained viewpoint. In *CRA*, pages 208–220, 1987.

[48] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[49] D. P. Huttenlocher S. Ullman. Object recognition using alignment. In *ICCV*, pages 102–111, 1987.

[50] M. Weber, M. Welling, and P. Perona. Unsupervised learning of models for recognition. In *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part I*, pages 18–32, London, UK, 2000. Springer-Verlag.

[51] P. H. Winston. The MIT robot. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 7, pages 431–463. Edinburgh University Press, 1972.

[52] H. J. Wolfson and I. Rigoutsos. Geometric hashing: an overview. *Computational Science and Engineering, IEEE* [*see also Computing in Science & Engineering*], 4(4):10–21, 1997.

[53] H. Zhang and J. Malik. Learning a discriminative classifier using shape context distances. In *CVPR*, pages 242–247, 2003.

[54] A. Zisserman. Trainable visual models for object class recognition. In *Pascal Pattern Recognition and Machine Learning in Computer Vision Workshop*, Grenoble, France, 2004.