Synthetic 3D Model-Based Object Class Detection and Pose Estimation

Joerg Liebelt

UNIVERSITE DE GRENOBLE

TECHNISCHE UNIVERSITÄT MÜNCHEN

N° attribué par la bibliothèque

THÈSE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITE DE GRENOBLE

Spécialité : Mathématiques et Informatique

préparée dans le cadre d'une cotutelle entre

L'UNIVERSITE DE GRENOBLE et la TECHNISCHE UNIVERSITÄT MÜNCHEN

Arrêtés ministériels : 6 janvier 2005 - 7 août 2006

présentée et soutenue publiquement par

Jörg LIEBELT

le 18 octobre 2010

Détection de Classes d'Objets et Estimation de leurs Poses à partir de Modèles 3D Synthétiques

---

Thèse dirigée par Cordelia SCHMID et codirigée par Pr. Rüdiger WESTERMANN

JURY

Pr. Augustin LUX	Président
DdR Michel DHOME	Rapporteur
Pr. Rainer LIENHART	Rapporteur
Pr. Daniel CREMERS	Examinateur
DdR Cordelia SCHMID	Directeur de thèse
Pr. Ruediger WESTERMANN	Co-Directeur de thèse

Thèse préparée au sein du Laboratoire Jean Kuntzmann et
de l'École Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique.

Détection de Classes d'Objets et Estimation de leurs Poses à partir de Modèles 3D Synthétiques

RÉSUMÉ : Cette thèse porte sur la détection de classes d'objets et l'estimation de leur poses à partir d'une seule image en utilisant des étapes d'apprentissage, de détection et d'estimation adaptées aux données synthétiques. Nous proposons de créer des représentations en 3D de classes d'objets permettant de gérer simultanément des points de vue différents et la variabilité intra-classe. Deux méthodes différentes sont proposées : La première utilise des données d'entraînement purement synthétiques alors que la seconde approche est basée sur un modèle de parties combinant des images d'entraînement réelles avec des données géométriques synthétiques. Pour l'entraînement de la méthode purement synthétique, nous proposons une procédure non-supervisée de filtrage de descripteurs locaux afin de rendre les descripteurs discriminatifs pour leur pose et leur classe d'objet. Dans le cadre du modèle de parties, l'apparence d'une classe d'objets est apprise de manière discriminative à partir d'une base de données annotée et la géométrie en 3D est apprise de manière générative à partir d'une base de modèles CAO. Pendant la détection, nous introduisons d'abord une méthode de vote en 3D qui renforce la cohérence géométrique en se servant d'une estimation robuste de la pose. Ensuite, nous décrivons une deuxième méthode d'estimation de pose qui permet d'évaluer la probabilité de constellations de parties détectées en 2D en utilisant une géométrie 3D entière. Les estimations approximatives sont ensuite améliorées en se servant d'un alignement de modèles 3D CAO avec des images en 2D ce qui permet de résoudre des ambiguïtés et de gérer des occultations.

MOTS CLÉS : vision par ordinateur, détection de classes d'objets, estimation de pose

Synthetic 3D Model-Based Object Class Detection and Pose Estimation

ABSTRACT : This dissertation aims at extending object class detection and pose estimation tasks on single 2D images by a 3D model-based approach. The work describes learning, detection and estimation steps adapted to the use of synthetically rendered data with known 3D geometry. Most existing approaches recognize object classes for a particular viewpoint or combine classifiers for a few discrete views. By using existing CAD models and rendering techniques from the domain of computer graphics which are parameterized to reproduce some variations commonly found in real images, we propose instead to build 3D representations of object classes which allow to handle viewpoint changes and intra-class variability. These 3D representations are derived in two different ways : either as an unsupervised filtering process of pose and class discriminant local features on purely synthetic training data, or as a part model which discriminatively learns the object class appearance from an annotated database of real images and builds a generative representation of 3D geometry from a database of synthetic CAD models. During detection, we introduce a 3D voting scheme which reinforces geometric coherence by means of a robust pose estimation, and we propose an alternative probabilistic pose estimation method which evaluates the likelihood of groups of 2D part detections with respect to a full 3D geometry. Both detection methods yield approximate 3D bounding boxes in addition to 2D localizations ; these initializations are subsequently improved by a registration scheme aligning arbitrary 3D models to optical and Synthetic Aperture Radar (SAR) images in order to disambiguate and prune 2D detections and to handle occlusions. The work is evaluated on several standard benchmark datasets and it is shown to achieve state-of-the-art performance for 2D detection in addition to providing 3D pose estimations from single images.

KEYWORDS : computer vision, object class detection, pose estimation

Thèse préparée au sein du
Laboratoire Jean Kuntzmann
51 rue des Mathématiques
Campus de Saint Martin d'Hères
BP 53
38041 Grenoble cedex 09

Abstract

The present thesis describes 3D model-based approaches to object class detection and pose estimation on single 2D images. We introduce learning, detection and estimation steps adapted to the use of synthetically rendered training data with known 3D geometry.

Most existing approaches recognize object classes for a particular viewpoint or combine classifiers for a few discrete views. By using CAD models and rendering techniques from the domain of computer graphics, we propose instead to build 3D representations of object classes which allow to handle viewpoint changes and intra-class variability.

We outline an unsupervised filtering process of pose and class discriminant local features on purely synthetic training data, and we derive a part model which discriminatively learns the object class appearance from an annotated database of real images and builds a generative representation of its 3D geometry from a database of synthetic CAD models.

During detection, we introduce a 3D voting scheme to reinforce geometric coherence by means of a robust pose estimation, and we propose an alternative probabilistic method which evaluates the likelihood of groups of 2D part detections with respect to a full 3D geometry. Both approaches yield approximate 3D bounding boxes in addition to 2D localizations; these initializations are subsequently improved and disambiguated by a registration scheme aligning arbitrary 3D models to 2D images.

The work is evaluated on several standard benchmark datasets and achieves state-of-the-art performance for 2D detection in addition to providing 3D pose estimations from single images.

Résumé

Cette thèse apporte des contributions à la détection d'objets en utilisant des modèles 3D. Plus exactement, elle porte sur la détection de classes d'objets et l'estimation de leurs poses à partir d'une seule image. Nous décrivons des étapes d'apprentissage, de détection et d'estimation adaptées à l'utilisation de données synthétiques pour lesquelles la géométrie est connue.

La plupart des approches existantes détectent des classes d'objets à partir de points de vue discrets en combinant des classifieurs. En utilisant des modèles CAO existants et des méthodes de rendu issues du domaine de la synthèse d'images, nous proposons de créer des représentations en 3D de classes d'objets permettant de gérer simultanément des points de vue différents et la variabilité intra-classe.

Pour obtenir ces représentations en 3D, deux méthodes différentes sont proposées. La première utilise des données d'entraînement purement synthétiques alors que la seconde approche est basée sur un modèle de parties combinant des images d'entraînement réelles avec des données géométriques synthétiques. Pour l'entraînement de la méthode purement synthétique, nous proposons une procédure non-supervisée de filtrage de descripteurs locaux afin de rendre les descripteurs discriminatifs pour leur pose et leur classe d'objet. Dans le cadre du modèle de parties, l'apparence d'une classe d'objets est apprise de manière discriminative à partir d'une base de données annotée et la géométrie en 3D est apprise de manière générative à partir d'une base de modèles CAO.

Pendant la détection, nous introduisons tout d'abord une méthode de vote en 3D qui renforce la cohérence géométrique en se servant d'une estimation robuste de la pose. Ensuite, nous décrivons une deuxième méthode d'estimation de pose qui permet d'évaluer la probabilité de constellations de parties détectées en 2D en utilisant une géométrie 3D entière. Les deux méthodes génèrent des détections en 2D ainsi que des estimations approximatives de poses en 3D ; ces estimations approximatives sont ensuite améliorées en se servant d'un alignement de modèles 3D CAO avec des images en 2D, ce qui permet de résoudre des ambiguïtés, d'effectuer un filtrage des détections en 2D et de gérer des occultations.

L'approche est évaluée sur plusieurs bases d'images de référence et nous montrons qu'elle est capable de fournir des estimations de pose en 3D à partir d'images 2D tout en générant des détections en 2D comparables à l'état de l'art.

Zusammenfassung

Ziel dieser Arbeit ist die Erkennung und Positionsschätzung von Objektklassen in einzelnen zweidimensionalen Bildern mittels eines dreidimensionalen modellbasierten Ansatzes. Die Arbeit untersucht Lern-, Detektions- und Positionsschätzungsverfahren, die auf die Verwendung synthetisch generierter Trainingsdaten mit bekannter dreidimensionaler Geometrie abgestimmt sind.

Der überwiegende Anteil existierender Verfahren erkennt Objektklassen aus einzelnen Kameraperspektiven oder kombiniert Klassifikatoren für einzelne diskrete Ansichten. Demgegenüber beschäftigt sich diese Arbeit mit dem Erlernen von dreidimensionalen Repräsentationen mittels CAD-Modellen und Bilderstellungsverfahren der Computergrafik, um mit Veränderungen des Erscheinungsbildes von Objekten einer Klasse aufgrund von wechselnder Kameraperspektive und ausgeprägter Intra-Klassen-Varianz umgehen zu können.

Es werden zwei verschieden Methoden zur Erstellung solcher dreidimensionaler Repräsentationen untersucht: ein unüberwachtes Lernverfahren zur Bestimmung von lokalen Bildmerkmalen aus rein synthetisch erstellten Trainingsdaten, welche für bestimmte Ansichten und Objektklassen charakteristisch sind, und ein überwachtes Lernverfahren, welches einerseits diskriminativ das Erscheinungsbild von Objektteilen aus annotierten realen Trainingsbildern bestimmt und andererseits unter Verwendung von synthetischen CAD-Daten ein generatives Modell der dreidimensionalen Geometrie dieser Objektteile erstellt.

Zur Detektion von Objektklasseninstanzen in einzelnen zweidimensionalen Bildern wird sowohl die geometrische Konsistenz von Gruppen lokaler Bildmerkmale mittels eines robusten dreidimensionalen Positionsschätzungsverfahrens bewertet als auch die Wahrscheinlichkeit bestimmt, dass Teile eines Objektes nach Projektion in den Bildraum in einer zur erlernten dreidimensionalen Modellgeometrie konsistenten Anordnung auftreten. In beiden Fällen kann zusätzlich zur zweidimensionalen Lokalisierung einer Objektinstanz im Einzelbild eine Schätzung ihrer dreidimensionalen Position und Ausrichtung erreicht werden. Diese Schätzung dient anschliessend als Initialisierung eines Registrierungsverfahrens, welches CAD-Modelle von Objektinstanzen auf Einzelbilder registriert und die Positionsschätzung weiter verbessert.

Die Genauigkeit der in dieser Arbeit untersuchten Verfahren zur Detektion und Positionsschätzung verschiedener Objektklassen wird auf mehreren standardisierten Testdatensätzen evaluiert.

Acknowledgements

I am indebted to my thesis supervisor Cordelia Schmid for her constant support and her invaluable scientific guidance. I am extremely grateful to Prof. Rüdiger Westermann for accepting to venture on the French-German co-supervision experiment beyond traditional Computer Graphics. This work would have been impossible without my colleagues at EADS, notably Klaus Schertler, Falk Schubert and Johannes Schels, whom I thank for their cooperation, countless creative discussions and their encouragement. As a consequence of the French-German cooperation, I have been admitted to and benefitted from two outstanding academic research teams at once, LEAR in Grenoble and tum.3D in Munich; I would like to thank their current and former members, particularly Adrien Gaidon and Alexander Kläser. This work was part of an industry cooperation with EADS Innovation Works in Munich; I would like to thank my superiors, Olaf Heinzinger and Dr. Richard Arning, for creating a reliable and inspiring work environment. Many people have helped establish and maintain the French-German co-supervision against all odds, notably Augustin Lux, Alba Castiglione and Petra Marzin; the mobility grant of the Deutsch-Französische Hochschule / Université Franco-Allemande has created the framework for this cooperation. I am deeply grateful to my parents for their continuous support and encouragement.

Contents

List of Figures

List of Tables

1

Introduction

Figure 1.1: Illustration of Plato's Allegory of the Cave (Jan Saenredam 1604).

In Part VII of The Republic, Plato introduces the "Allegory of the Cave": prisoners in a cave perceive the outside, real (3D and colored) world exclusively as the (2D and colorless) shadows of objects cast through an inaccessible pinhole into their confined prison world; see figure 1.1. Lacking divergent experience, they are led to equate the shadows with the real world and remain forever unaware of the true nature underlying their perceived shadow universe. Philosophy typically interprets this allegory as an illustration of the limits of human knowledge, the deceitful nature of human perception and the striving of the individual after the *true* form of things [126]. Interestingly, the functioning of human vision exhibits parallels to the above allegory, albeit on a lower, less philosophical level. Numerous studies suggest the ability of the human and even of the animal brain to infer complete (perfect) 3D form from few, even flawed and incomplete 2D structural clues (such as in figure 1.2), seemingly possessing a hidden model of the perfect 3D form of

things and the capacity of establishing correspondences between weak 2D clues and idealized 3D models. Sophisticated brain imaging methods (functional magnetic resonance imaging (fMRI), see for example [128]) have been able to locate the regions actively involved in the process, but the full understanding of the process itself is as yet beyond the reach of science.

Figure 1.2: Optical illusion: the human eye can distinguish a dalmatian dog by inferring 3D shape and shadows from minimal optical clues; illustration in [33].

Computer vision research, in particular in the domain of object (class) detection, has been striving after a comparable level of perfection. It aims at building algorithmic methods to automatically detect and identify classes of objects in single images and determine their location in 2D image space and, if possible, their 3D pose. Even though significant progress has been made in recent years, both in image representation and in learning theory, the results achieved do not yet come close to the human vision system. Still, given the ubiquity of consumer image recordings and the need to organize and commercially exploit the subsequent tremendous amount of visual data, research in computer vision can provide innovative solutions to these problems even without matching the performance of nature.

Recent findings in medical brain research [128] allow to conclude that the human brain makes use of some form of 3D representation of objects learnt during early childhood. While the human brain can rely on a natural stereo sensor as data source, many computer vision tasks rely exclusively on 2D images. Still, the advantages of a 3D representation, notably for identification, pose estimation and navigation tasks, are obvious: ambiguities can be avoided, appearance variations due to multiple viewpoints or geometry-dependent external factors such as shadows can be implicitly resolved and additional information on pose and scene geometry can be derived. The predominant amount of work in the field of object detection has focussed on 2D, the *shadow world* of Plato's allegory, both for knowledge representation as well as for result visualization. The present thesis takes up the idea of a preexisting, idealized 3D model-based representation in form of CAD computer graphics models and explores approaches of how this added knowledge can contribute to improving the computer vision tasks of object class detection and pose estimation from single 2D images. It benefits from the significant technological improvements which have recently taken

place in the domain of computer graphics, providing readily available tools to handle 3D model representations and to imitate the natural process of deriving 2D images from 3D representations.

1.1 Problem Statement

This thesis aims at developing algorithmic approaches to detect object classes in images. In the following, we provide a detailed definition of the task and summarize the main difficulties. Given the 2D output of an imaging sensor, including but not necessarily limited to optical cameras, we would like to detect all instances of a specific object category which are present in the sensor output. For simplification, we assume in the following that the sensor is an optical camera and the sensor reading is given as a 2D color image; it will be shown later (see for example section 2.2.5) that the extension to different sensors is possible.

In this scenario, the term *object class*, in opposition to a *specific object instance*, denotes a group of objects which share sufficiently many properties to justify assigning them to a *semantic equivalence group*; the *object class* can therefore be considered as a set of *specific object instances* which are similar in terms of some similarity measure. Imprecisions in natural language and interpretational freedom make it difficult to decide upon a universal definition of *semantic equivalence*; fortunately, theoretical work on the structuring of natural language, such as the WordNet approach of [69], can serve as a framework to address the problem. Starting with a set of what can be considered *atomic*, non-divisible instances, a bottom-up hierarchy can be build to reflect sets of cognitive synonyms called *synsets*, each expressing a distinct concept, which are further interlinked by means of conceptual-semantic and lexical relation. Given the large number of levels of such a semantic hierarchy, in the present work we focus on a small set of relatively vast semantic groups which stand exemplary for potentially relevant *object classes*: we resort to the three man-made object classes of cars, motorbikes and bicycles as exemplary *object classes* on which the algorithmic concepts will be demonstrated. Figures 1.3, 1.4, 1.5 show examples for these classes as well as a typical negative test image (1.4, right).

Figure 1.3: Some images from the PASCAL VOC2006 CAR dataset [18].

The term *object class detection* denotes the identification of all, potentially overlapping, locations in 2D image space which show an object belonging to a given class. An identification can be conveniently represented by a *bounding box*, i.e. a rectangle in 2D image space which contains the entire visible object area, alongside the *classification* into one of the *object classes* in question. Due to imaging conditions, a number of special cases exist; for simplification, we only consider the following two cases: objects can be *truncated* (figure 1.3, third and fourth image from the left),

Figure 1.4: Some images from the motorbike and bicycle classes (left, center) of the PASCAL VOC2006 dataset [18] and an example of a negative test image for those classes (right).

Figure 1.5: Some images from the car and bicycle classes of the 3D Object Category dataset [102].

where we assume the *accurate* bounding box to contain only the part of the object which is visible inside the image boundaries; and objects can be *occluded* (figure 1.3, right), where the accurate bounding box is defined as the largest rectangle circumscribing the visible parts of the occluded object; note that in this case, the rectangle might also contain the occluding objects, regardless of their class membership. Other cases, such as objects indirectly visible as reflections or in symbolic form as drawings, will not be considered.

In order to assess the performance of an approach to *object class detection* and benchmark against previous work, common evaluation datasets and evaluation criteria are required and a meaningful visualization of the performance has to be chosen. Due to the nature of learning-based approaches, these datasets have to provide training and testing data in order to guarantee the comparability of the results achieved. In the present work, we evaluate on three state-of-the-art benchmark datasets, the 3D Object Category [102] and the PASCAL VOC 2006 [18] and PASCAL VOC 2007 [17] datasets; some example images for the PASCAL 2006 dataset are shown in figures 1.3, 1.4, some examples for the 3D Object Category datset are shown in figure 1.5. Both datasets provide training examples containing 2D bounding box and class annotations, and testing examples with groundtruth annotations which allow to evaluate detection performance. The 3D Object Category dataset [102] emphasizes a systematic evaluation of multi-view conditions and provides weak pose annotations; the evaluation sections of chapters 3 and 4 provide details on the annotation format. In contrast, the PASCAL VOC 2006 and 2007 datasets focus on difficult and realistic image conditions, a large number of training images to be representative of a large-scale application and significantly more negative (i.e. not containing an object of a given class) than positive images. Training and groundtruth annotations follow the definitions given in the previous paragraphs. In order to compare 2D detections, a distance metric has to be used which takes into consideration that manual annotations may display imprecisions and different approaches to the detection task may generate bounding boxes which do not exactly match the annotations. Con-

sequently, the relative overlap between annotation and detection is suggested in [18] to measure detection precision; it assumes a positive detection when the area of overlap between a detection and an annotation exceeds a given percentage of the joint area of detection and annotation. This so-called *overlap criterion* has become the standard. In section 3.4.1, the evaluation procedure is described in detail.

A number of inherent difficulties render the task of object class detection particularly challenging.

- **Intra-Class Variation:** Objects within a class share certain semantic properties. However, they can differ in numerous ways without influencing these common properties; their differences make up the intra-class variation which a detector for this class has to cope with. When considering the example class *car*, typical intra-class variations arise from deviations in body geometry (notchback, hatchback etc.) as well as texture (shape of components, coloring etc.). In the present work, this obstacle is mainly addressed by using robust local image statistics, generalizing classifiers and the systematic generation of variations from synthetic training data. In principle, non-rigid deformations of the objects can be considered intra-class variation. In the present work, we focus exclusively on rigid object geometries; consequently, the variation induced by non-rigid deformations cannot be covered. See the section 6.2 on future work for possible approaches to the problem.

- **Background:** When evaluating a detector on real images, objects of a class can occur in many different environments. Each surrounding scenery introduces new and potentially unseen image components considered as background. Moreover, due to the choice of rectangular bounding boxes as training annotations, training data will not be perfectly segmented from the background which may contaminate the positive training samples. In the present work, we generate training samples with systematically varying background to explicitly guide the training process towards the common object (foreground) properties.

- **Imaging Noise:** Every imaging process is influenced by global illumination conditions, sensor noise and environmental conditions which impact the appearance of objects in the scene. By resorting to a synthetic training process in conjunction with robust image statistics, we can simulate some of the most dominant factors to account for their impact on real test images.

- **Object Pose:** During the image formation, different 3D scene and object configurations induce variations in object appearance due to changing viewpoints and occlusions. Here, we account for these changes in appearance by building object class representations during training which explicitly include 3D geometric clues. During detection, a pose estimation then allows to recover information on the scene structure from the image and assess the plausibility of a 2D detection in terms of a full 3D geometry.

- **Multi-Class:** The presence of multiple object classes in images constitutes an additional challenge, since possible ambiguities between classes have to be resolved and the computational effort is increased. In the present work, a strategy to multi-class detection is employed which is commonly refered to as *one-vs-all*: each object class is treated independently of the other classes in a training and detection process which aims at separating the relevant object class from a joint background class assembled from non-object background and all other

classes. While this strategy effectively ignores potential contextual dependencies between classes, it favors a flexible and extensible evaluation.

1.2 Context

Historically, research on learning-based multi-view object class detection and pose estimation developed from initial work in the 1990s; it can be separated into different approaches which we will briefly summarize and describe how they relate to our own contributions. A detailed survey of more recent related work is given in chapters 3 and 5, respectively.

1.2.1 Global Approaches

A global approach to object class detection attempts to describe the appearance of the entire object under a set of discrete viewpoints. The choice of the appearance representation varies; traditionally, the detection of the object class of human faces received the most attention. Turk and Pentland [120] projected image patches of faces for a few discrete viewpoints separately into the dimensionality-reduced eigenspace. Subsequently, several authors [72, 74] merged geometric and shape clues with the appearance representation to achieve approximate viewpoint estimations. In order to achieve a higher generalization of the appearance description, more invariant descriptions are suggested such as in [26] and more sophisticated classifiers are employed, see for example [123] and [91]. During detection, patches from test images were selected either exhaustively or based on some simple entropy- or color-based predetection step. An efficient strategy for evaluating a global detection method on an entire image is known as a *sliding window classifier* [84, 115]; it builds a pyramid of images on different scales and applies the detector exhaustively to all possible image subregions. Still, identifying reliable globally invariant descriptions for entire objects is difficult and these approaches are often limited to idealized conditions of homogeneous backgrounds or pre-segmented regions without occlusions and object classes with low appearance variation. In the present work, we combine global and local components to flexibly describe object class appearance and geometry: in chapter 3, groups of viewpoint-consistent local descriptors are matched to optimize a global distance measure, and in chapter 5 we rely on global appearance similarity to derive improved 3D pose estimations.

1.2.2 Part-Based Models

As a remedy to the shortcomings of global approaches, objects were represented by multiple layers of parts [107] or spatially subdivided into regions. The appearance of each part was described by applying the global techniques to more localized image patches and the co-occurrence and the spatial relationship between these parts were modeled, for example with spring-like forces as in [24]. By allowing more internal variation and distributing the detection of a full object over a set of part classifiers, robustness towards appearance variation and occlusions was increased. However, the fundamental dependency on the per-patch appearance classification remained. In chapter 4, we propose a part-based model for appearance and geometry which builds on a fixed grid of object parts and a discriminatively learnt appearance classifier.

1.2.3 Local Descriptors

Consequently, methods to repeatably select characteristic image regions [38, 70], termed *feature detectors*, and to discriminatively represent them by more invariant local descriptions [60, 65, 106], termed *feature descriptors*, were introduced. These methods usually exploit gradients and generalized filter responses to detect and encode characteristic structures such as corners and edge combinations in the image and cope with scale, translation or affine invariance by suitable normalization [67, 68]. Several authors consequently proposed robust distance measures [32] to compare the resulting high-dimensional feature descriptors. More recent work [5] has focussed mainly on optimization. As an alternative to the use of detectors, descriptors can also be computed densely over the entire image [118]. The advent of local features extended the feasibility of object class detection significantly and gave rise to numerous approaches exploiting their invariance for this task, for example [105]. Local features are used in chapter 3 to capture object class appearance; we propose a novel filtering process to determine those features which are robust towards small local variations and discriminative for their class and viewpoint. In chapter 4, local features form the building blocks of our appearance representation.

1.2.4 Vocabulary Codebooks

In order to more efficiently extract common characteristics from a set of local features, the use of unordered co-occurrence relationships of local features has been proposed [11], termed *bag-of-features*. Based on vector quantization of a set of local descriptors harvested from training images, a set of prototypical descriptors is generated which can be interpreted as a base vocabulary or codebook. Local features computed on test images are then assigned to their closest codebook and their occurrence counts are stored in a histogram. By varying the region of influence for a histogram, localization can be achieved [63]. The original *bag-of-features* approach represents unordered, spatially unaware co-occurrence counts. Due to symmetries and self-similarities of objects, such unordered sets may have a reduced discriminatory power. In [49], based on the idea of [32] a spatial layout of co-occurrence histograms was introduced which allows to consider feature constellations on varying levels of locality. Both object class detection approaches outlined in this work rely on codebooks; in chapter 3, local descriptors are clustered into a codebook of fixed size where each codebook entry is annotated with additional information on the 3D geometry of the object class. In chapter 4, dense spatial pyramids are used to capture the global object and local part appearance.

1.2.5 Pose Estimation

Pose estimation methods can build on different underlying concepts. In the present work, rigid 3D models and single 2D images are used to derive point sets which are optimally aligned onto each other to estimate the camera pose; furthermore, a calibrated camera is assumed. Alternative strategies use higher-order components such as line [76, 85] or free-form objects [51] or combine calibration and estimation steps; these will not be discussed here; for a detailed survey, see [96]. Work on point-based pose estimation can rely on an iterative solution as in [36, 57] and [81], or

resort to a closed-form solution such as the perspective three-point (P3P) method [23, 37, 129], which in turn relies on a 3D-to-3D point set alignment suggested by [44] and [121]. Optimization strategies range from gradient descent [57] to simulated annealing [30]. If the initial point correspondences are not known or unstable, robust correspondences have to be determined as part of the pose estimation process. Frequently, robustness is derived from randomly sampling minimal subsets of point correspondences and evaluating inlier counts or accumulated errors; other authors [30] perform iterative reweighing of all point pairs. In chapter 3, we compare three pose estimation approaches for the task of computing approximate poses based on clustered local features with 3D annotations; in chapter 4, a probabilistic approach is introduced which determines the most likely pose given the constellation of detected parts. For a given initialization of object class and approximate pose, we describe in chapter 5 how the initial pose estimate can be improved by optimizing a global appearance similarity measure.

1.3 Contributions

In this thesis, we aim at extending object class detection and pose estimation on single 2D images by a 3D model-based approach, building on existing CAD models and rendering techniques from the domain of computer graphics.

The first contribution is a 3D approach to multi-view object class detection. Most existing approaches recognize object classes for a particular viewpoint or combine classifiers for a few discrete views. We propose instead to build 3D representations of object classes which allow to handle viewpoint changes *and* intra-class variability. Our approach extracts a set of pose and class discriminant features from synthetic 3D object models using a filtering procedure, evaluates their suitability for matching to real image data and represents them by their appearance and 3D position. We term these representations *3D Feature Maps*. For recognizing an object class in an image, we match the synthetic descriptors to the real ones in a 3D voting scheme. Geometric coherence is reinforced by means of a robust pose estimation which yields a 3D bounding box in addition to the 2D localization. This work was published in [55].

The second contribution extends the previous approach to multi-view object class detection by introducing discriminative part classifiers and a probabilistic pose estimation method; it further allows to combine training data from synthetic as well as from real images. Appearance and geometry are treated as separate learning tasks with different training data. A part model is used which discriminatively learns the object appearance with spatial pyramids from a database of real images, and encodes the 3D geometry of the object class with a generative representation built from a database of synthetic models. The geometric information is linked to the 2D training data and allows to perform an approximate 3D pose estimation for generic object classes. The pose estimation provides an efficient method to evaluate the likelihood of groups of 2D part detections with respect to a full 3D geometry model in order to disambiguate and prune 2D detections and to handle occlusions. This work was published in [54].

The third contribution addresses the limitation of the previous methods which provide only approximate 3D pose estimations. Building on initializations of object category and approximate 3D pose as provided for example by the previous methods, a registration scheme is described to

align arbitrary standard 3D models to optical and *Synthetic Aperture Radar* (SAR) images in order to recover the full 6 degrees of freedom of the object. We propose a novel similarity measure which combines perspective contour matching and an appearance-based *Mutual Information* (MI) measure; it is optimized using an evolutionary Particle Swarming strategy, parallelized to exploit the hardware acceleration potential of current generation graphics processors (GPUs). We show that our approach leads to precise registration results, even for significant image noise, small object dimensions and partial occlusion where other methods would fail. This work was published in [52].

1.4 Thesis Overview

In chapter 2, we provide details on the synthetic training data which will be used in all subsequent chapters and we outline the camera models and parameterizations that define the rendering process.

In chapters 3 and 4, we present two different approaches to object class detection, adapted to the use of synthetic training data. A survey of recent related work on this topic is given in section 3.2.

The first approach, described in chapter 3, is based on an unsupervised training process (section 3.3.1) to robustly derive appearance and 3D geometry representations from synthetic CAD models. In section 3.3.2, we provide details on the corresponding detection process which builds on a probabilistic voting scheme and a robust 3D pose estimation; in section 3.4, the different contributions of this initial approach are evaluated on several benchmark databases.

Based on the analysis of the first approach, we outline a second method in chapter 4 which combines a part-based appearance representation learnt discriminatively from real training images and a 3D geometry representation learnt generatively from synthetic CAD models. In section 4.1, we relate this approach to the method from chapter 3, we describe the training and detection steps in section 4.2 and provide results of the experimental evaluations in section 4.3.

In chapters 3 and 4, we have focused on detecting generic object classes in images and on obtaining an approximate estimation of their 3D pose in addition to a 2D localization in image space. In chapter 5, we describe a method which builds on the generic 3D detection results of the previous chapters as initializations to more precisely align a single 3D CAD model to a single image and to perform a fine-grained object instance selection. We summarize previous work on CAD model registration in section 5.2 and derive a similarity measure in section 5.3. In section 5.4, we analyze the impact of a suitable optimization scheme on various test cases. In section 5.4.4, we demonstrate on a calibrated test set that the proposed method offers a suitable tradeoff as regards precision, speed and universality in order to complete the generic object class detection approaches from chapters 3 and 4 towards a more precise 3D pose estimation and the identification of specific object instances.

Chapter 6 concludes the thesis and provides an outlook on promising future lines of work.

2

Synthetic Training Data

"The lines will continue to blur between computer graphics and photography until the idea of what is real becomes meaningless. Whether you capture something with a lens or use virtual photons, the rules of lighting are the same." James Cameron

In this chapter, we will provide details on the synthetic CAD models used in the subsequent chapters and we will outline the camera models and parameterizations which define the rendering process to generate training data. We will start by describing the properties of these CAD models and the main elements governing the generation of synthetic images. We will then provide a short outline of the camera model typically used in the domain of computer vision and discuss how it relates to a computer graphics rendering pipeline.

2.1 Introduction: In Defence of Synthetic Data

One of the ideas of the present work is the use of synthetic data for computer vision tasks, notably for object class detection and pose estimation. Synthetic data is modeled after reality and both the process of creating CAD models and the generation of images from these models are designed to provide the most realistic representation possible. Still, both the model creation as well as the rendering suffer from imperfections: the model creation process can be limited in accuracy by the design tools used or the artistic freedom of the human designer, while the rendering process depends on the quality of the physical model chosen to represent surface shading and lighting in addition to potential hardware limitations in resolution and numerical precision. Consequently, the employed methods have to be adapted in order to account for these deficiencies. This is one of the main objectives of the present work, both for object class detection and for pose estimation.

Once the gap between synthetically generated data and real images has been bridged, however, the advantages of the synthetic world are numerous: synthetic data can be generated on demand, exhaustively including variations in lighting, background and viewpoints. Surface and material properties can be modified to span the range of possible object appearances, dynamic scene- and geometry-dependent noise can be introduced and configurations of these parameters which are rare in real scenarios can be generated at will. Full 3D scene and object geometry information is

available and semantic annotations and metadata are an automatic byproduct of the data generation; finally, the results of the training or validation process can be immediately fed back into the data generation.

The majority of object detection and pose estimation methods rely on databases of real images as training data or templates. For specific tasks or benchmark purposes, these real databases are manually chosen in the hope of being representative of the problem [89]. Consequently, it can be argued that this process of drawing samples from real data is as imperfect as the synthetic data generation; in neither case, the exhaustiveness of the drawn samples and the persistence of the underlying selection model can be guaranteed. Both approaches are based on assumptions and heuristics as to the expected deployment scenario, but real data is limited in scope since no real sample can represent all variations in appearance, geometry, environment conditions, viewpoint and noise. In this work, we argue that the advantages of using synthetic data outweigh its shortcomings.

Synthetic data has been used for computer vision tasks in the past. Among the first authors to suggest the use of CAD models for object detection tasks were [25, 40, 57, 79, 112, 130], mostly by building relational graphs of edge segments, by exploiting the link between surface shading and model normals or by flexibly matching and aligning contour segments. Through the use of edge-based features, the approaches are able to generalize up to a certain degree over classes of objects with similar outlines; however, they do not systematically take the object appearance into account. Consequently, they do not require appearance-based textured rendering and lighting. Approaches to detection, 3D pose estimation and tracking of specific objects have been proposed for example by [1] which resort to appearance-based rendering; they mostly rely on the availability of photorealistic textures which correspond exactly to the appearance of the object in an image and consequently do not generalize well to object classes.

In this work, in order to exploit the potential of synthetic data which provides both geometry and appearance information, we rely on standard computer graphics techniques which are available in most current hardware platforms, operating systems and programming toolsets. Recently, the flexibility of both hardware and programming frameworks has been extended to allow using graphics processors for general purpose tasks, which significantly facilitates their use in the computer vision domain. A detailed review of the techniques underlying the rendition of synthetic images from 3D models and the concept of general purpose programming on graphics hardware is beyond the scope of this work. However, we will provide a summary of selected aspects which impact the quality of the synthetic data generation and thus the performance of the methods which will be described in the subsequent chapters.

2.2 Data Representation and Rendering

In this section, we summarize the camera and rendering model and their parameterizations in order to generate synthetic training data to be used in the following chapters.

Figure 2.1: Selection of some 3D models used to represent the object classes *car*, *motorbike* and *bicycle*. See appendix B for a visualization of the complete database of 3D models used in the present work.

2.2.1 Data Representation

The CAD models used in this work stem from different free and commercial CAD model databases, notably turbosquid.com, 3d02.com and doschdesign.com. Each model consists of a set of surface points or vertices and the associated adjacency information describing its 3D surface. Each vertex is annotated with additional properties, a *material*, describing the appearance of the model surface at that point. The material contains a diffuse color value including the degree of transparency, a set of 2D texture coordinates and a factor describing the behavior of the local surface under specular reflection. Semantic grouping of vertices into parts is available in some models depending on their design process, although it is not used in the present work. Textures represented as images may be associated with the model to provide additional detailed appearance information; they are mapped onto the model surface using perspective correction based on the Besenham interpolation algorithm, the standard approach in computer graphics. Figure 2.1 shows a few examples of 3D models used to represent the three object classes bicycle, motorbike and car; also see appendix B, figures B.1, B.2, B.3, B.4. These models will serve as training data or registration templates in the following chapters. As a consequence of the lack of standardization in CAD model design and the artistic freedom of the model designers, the level of detail in terms of geometry, coloring and texture differs significantly among the 3D models in our database. Although methods exist to normalize these differences for example by surface reparameterization [43], equivalent differences are present in real image datasets, such as varying image sizes and resolutions and vastly

differing camera characteristics and quality. In this work, we assume that the quality differences in the CAD models can be considered beneficial to improving the generalization capacity of the computer vision tasks. In future work, methods will have to be developed to quantify these quality differences and to assess their impact on the training performance systematically; see section 6.2. In order to account for background variation, rendered models are overlayed on real images not containing any relevant object classes as shown in figure 2.2; these background images stem from the category *None* of the TU Graz-02 Database [80]. Note that for the experimental evaluations, we additionally resort to the pure negative training images provided as part of the benchmark data sets.

Figure 2.2: Examples of real background variation in synthetically rendered training data.

2.2.2 Camera Models in Computer Vision and Computer Graphics

In the following, homogeneous notations are used for 2D points $x_{2D} = (x, y, 1)$ and 3D points $x_{3D} = (x, y, z, 1)$ such that $(x, y, w) \rightarrow (x/w, y/w)$ and equivalently $(x, y, z, w) \rightarrow (x/w, y/w, z/w)$.

2.2.2.1 Real Camera Model

Assuming a set of homogeneous 3D coordinates x_{3D}^{wc} in some world coordinate system, a camera projects each point in world coordinates onto a point in homogeneous 2D screen coordinates x_{2D}^{sc} such that

$$x_{2D}^{sc} = K\, x_{3D}^{cc} = K[R|t]\, x_{3D}^{wc} = KP\, x_{3D}^{wc} \tag{2.1}$$

where x_{3D}^{cc} are the coordinates in a camera coordinate system after rotation R and translation t; all coordinate systems are assumed left-handed. The matrix K is usually called the calibration or intrinsic matrix and P is the extrinsic or camera matrix. K comprises the parameters describing

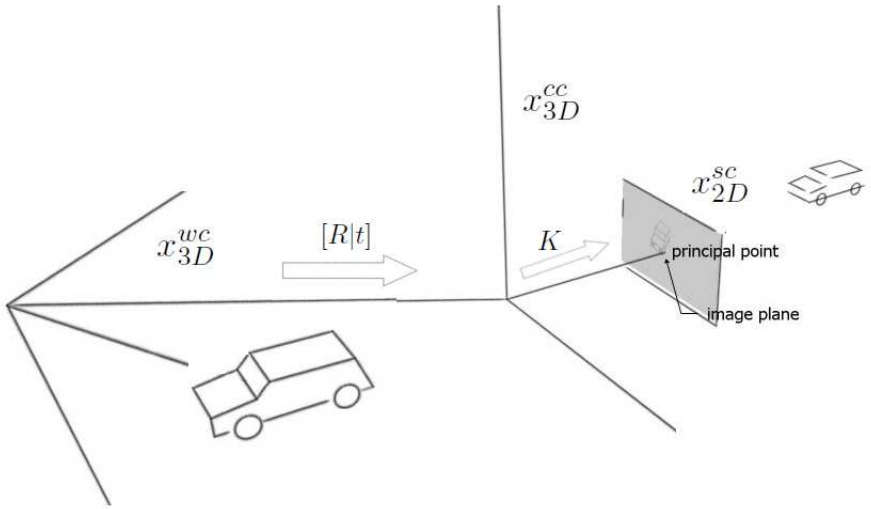

Figure 2.3: Camera projection: transformation from world to camera coordinates and projection into the image plane.

the internal camera properties, notably skew s, focal length f, screen pixel dimensions (m_x, m_y) and principal point $p = (p_x, p_y)$ such that

$$K = \begin{pmatrix} f\,m_x & s & p_x & 0 \\ 0 & f\,m_y & p_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{2.2}$$

Simplified models may assume $s = 0$ and $m_x = m_y$ where $(x, y, z)_{3D}^{cc} \rightarrow (fm_x x/z + p_x, fm_y y/z + p_y)$. The camera matrix

$$P = \begin{pmatrix} & \mathrm{R} & & \mathrm{t} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.3}$$

can be further decomposed into a sequence of translation and rotations. Figure 2.3 illustrates the transformation and projection steps.

2.2.2.2 Synthetic Camera Model

Different camera model notations exist in computer graphics, mostly varying with respect to the orientation of the coordinate systems used. In the present work, we follow the notation specified by the $DirectX^{©}$ pipeline since it has been used in the implementations. $DirectX^{©}$ relies on a left-handed coordinate system and a projection sequence

$$x_{2D}^{sc} = S \cdot C \cdot P \cdot M \; x_{3D}^{vc} \tag{2.4}$$

where x_{3D}^{vc} denotes a 3D point (also called *vertex*) in the object-centered coordinate system of the 3D CAD model which the point belongs to. The following equivalences between the synthetic and the above described real camera model exist:

$$K \equiv S \cdot C$$
$$x_{3D}^{wc} \equiv M\, x_{3D}^{vc}$$

where $M = [R_M|t_M]$ denotes the additional transformation between object and world coordinate system. The definition of S and C with an additional reference system is necessary for resolving occlusion and visibility of objects as shown in figure 2.4. C is defined by the camera field of view and the distance of the closest (z_{near}) and farthest (z_{far}) distance at which an object is still to be visible as illustrated in figure 2.5, left;

$$C = \begin{pmatrix} cot(fov/2) & 0 & 0 & 0 \\ 0 & cot(fov/2) & 0 & 0 \\ 0 & 0 & \frac{z_{far}}{z_{far}-z_{near}} & 1 \\ 0 & 0 & \frac{-z_{near}\cdot z_{far}}{z_{far}-z_{near}} & 0 \end{pmatrix}. \tag{2.5}$$

In order to perform this step efficiently, the view frustrum is converted into a normalized representation, the *clip space*, as shown in figure 2.5, left. The final screen transformation S maps the contents of the clip space from normalized coordinates in $[-1, 1]$ to screen coordinates with width w and height h.

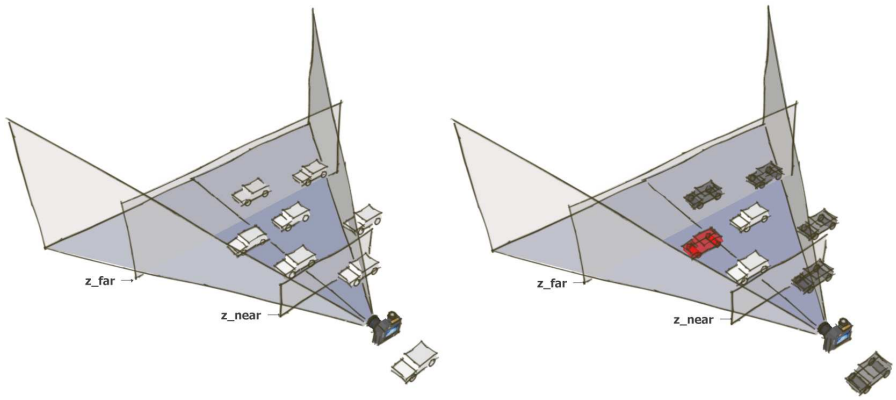

Figure 2.4: Synthetic rendering: the view frustrum is defined by the field of view and the z_{near}, z_{far} planes (left); objects are removed when they are outside of the view frustrum (right, in black) or occluded (right, in red).

2.2.3 Optical Rendering

In this work, the objective of creating synthetic renditions of CAD models to provide training data for computer vision tasks does not necessarily lie in improving the rendition quality to achieve

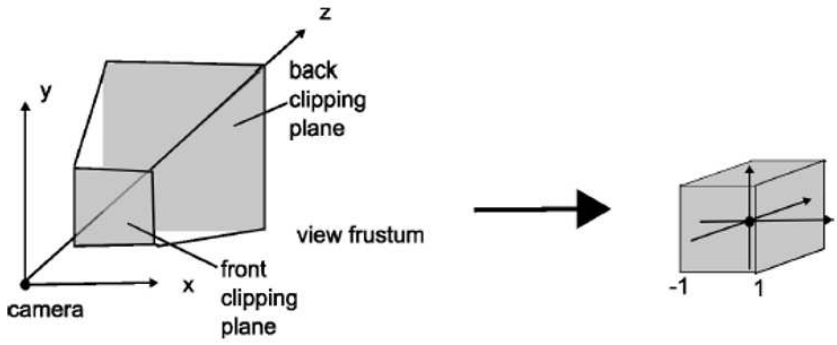

Figure 2.5: Synthetic rendering: efficient handling of occlusions by transforming the view frustrum to normalized clip space, illustration courtesy F. Schubert.

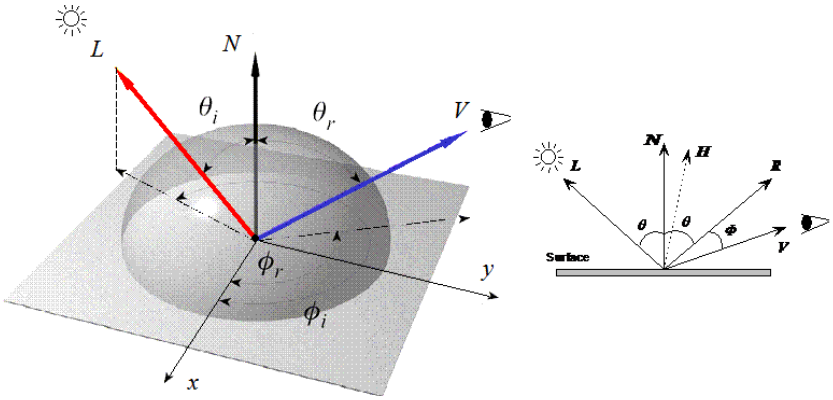

Figure 2.6: Lighting and reflectance modeling for synthetic rendering. Illustration in [131].

photorealism by fully exploiting the most advanced computer graphics technologies, driven by the entertainment industry and commercial visualization and simulation. We argue instead that it may not be necessary to increase rendering performance beyond what is currently considered as the basic functionality; unless the impact of dataset selection on computer vision systems is clearly understood [89], it may not be the most promising approach either. Instead, the implicit simplifications in the physics and lighting models which form the basis of current consumer-grade computer graphics methods can contribute to improving computer vision systems. Instead of increasing the rendering performance to match reality, some degree of natural idealization is introduced, thereby increasing robustness and generalization performance of the vision algorithms. In this work, we describe methods which are capable of dealing with differences between synthetic and real data up to a certain degree, consequently depending less on rendition. Figure 2.7 shows a real object (left) and its synthetic rendering (second from left) using the approach described in this section; the most notable difference is due to the opaque rendering of originally transparent surfaces.

Since the advent of computer graphics, numerous lighting models have been proposed in the literature, ranging from crude simplifications of local shading to complex global illumination and ray tracing, taking into account the interaction of CAD models with the ambient scene. These light models are closely connected to the surface interpolation strategies of the geometric models. A survey of the domain can be found for example in [110]. The contribution of the present work does not lie in the domain of computer graphics, but in applying it to computer vision tasks. Since the subsequent objectives are different, certain simplifications can be made as to the complexity of the used lighting model; notably, no interaction with the ambient scene such as indirect lighting or shadow generation is required. Instead, real background images represent the context variation which the algorithms are expected to deal with; see the previous section 2.2.1 for details. Moreover, we are not concerned with more complex material properties such as transparency or anisotropy in surface appearance, thus a Lambertian (directionally independent) model can be considered sufficient. As a result, we chose the Blinn-Phong lighting model [6] which has become the standard in most rendering pipelines; it has been experimentally shown [78] to offer a suitable tradeoff between realism and modeling complexity. In simplifying the contribution of ambient light, per-vertex and texture color and reflection properties of the surface, it is capable of creating renditions which are useable for the methods proposed in the following chapters. Its ability to generate reflection highlights of varying emphasis on the rendered surfaces is sufficient to introduce at least one of the important variations in the training data.

The empirical Blinn-Phong lighting model [6] is an approach to modeling the illumination intensity stemming from a light source with given wavelength and direction (L in figure 2.6). The light from the source is conveyed to a sensor V, depending on the orientation (normal N) and different additional properties of the illuminated surface, notably its diffuse and specular reflectance. We assume a single point light source at infinite distance and a locally planar surface patch. The resulting intensity at a single surface point follows as

$$I = I_a k_a + I_i k_d (L \cdot N) + I_i k_s (N \cdot (\frac{L+V}{2}))^n \tag{2.6}$$

where I_a is the intensity of the ambient light, k_a the coefficient of ambient reflection, I_i the intensity of the incident light, L its vector (see figure 2.6), k_d the coefficient of the diffuse reflection for the material, k_s the coefficient of specular reflection, and n an index to weigh the contribution of the specular component; the actual specular component depends on the angle ϕ between sensor V and reflection vector R (see figure 2.6, right) which in this formulation is approximated by $N \cdot H$ with $H = \frac{L+V}{2}$ to reduce computational complexity. The above equation can be extended to multiple light sources by summation over their respective contributions. In order to use this light model, the CAD models have to provide the local color values and reflection coefficients allowing to compute the above term.

A frequent problem in synthetic rendering stems from aliasing artifacts due to limited sampling rate and numerical precision. In the present work, the problem is circumvented by resorting to the standard anti-aliasing approach of resampling the image at multiple resolutions, adapted to the distance of the object surface to the camera; see [110] for details on anti-aliasing.

2.2.4 Linking Pixels to 3D Points

One key advantage of a modern rendering pipeline lies in its flexibility. In section 2.2.5, we show exemplarily how the same rendering procedure can be used to create simplified simulations of a radar sensor; in a similar way, infrared and polarimetric sensor simulations could be derived from the 3D CAD models and noise, distortion and environmental factors such as fog and motion blur could be added. In the following chapters, the association of image pixels with their originating 3D points on the model surfaces will be crucial in order to build 3D representations of the object classes in addition to describing their visual appearance. This association of pixels and 3D points is made possible by exploiting the flexibility of the rendering pipeline in generating arbitrary per-pixel outputs. Instead of only generating the four color values R,G,B,A per pixel, we can pass on all information contained in the original 3D model down to the pixel level and use additional output channels for each pixel to save material properties, arbitrary model metadata or the 3D coordinates, surface normals or derived gradients of the model surface point which is projected into the current pixel position. Since the rendering pipeline inherently resolves occlusion, only the data associated with visible pixels is written. Figure 2.7 shows some outputs generated from a 3D model (intensity, surface normals, color gradients, surface contour).

Figure 2.7: Real object (left) and some outputs generated from its corresponding 3D CAD model (left to right: intensity, surface normals, color gradients, surface contour).

2.2.5 Extension to Different Sensors: Synthetic Aperture Radar

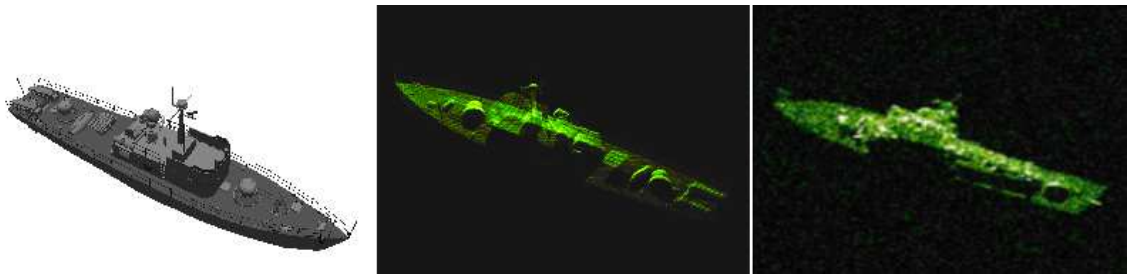

Figure 2.8: Simulated SAR rendering based on a 3D CAD model (left); our result (center) and a commercial grade simulator (right, [66]).

In chapter 5, we propose an approach capable of working on sensor inputs from other than optical devices; this will be demonstrated exemplarily on *Synthetic Aperture Radar* (SAR) data. SAR is a remote sensing technique which exploits post-processing of a sequence of radar signals over time to fusion the echos from points at different distances. As a consequence, an object remains longer

inside the beam of the radar and its resolution can be increased in the resulting radar image by merging the separate echos, given the sensor movement is known.

In the present work, SAR images are generated on the model side using a simple algorithm to simulate the SAR specific projection properties based on provided 3D CAD models. Although we do not simulate speckles and highlights which would require raytracing precision, the quality of the simulated SAR modality is sufficient to register 3D models to real SAR sensor input and to recover the 3D pose of the objects in the scene.

Assume that we are given a 3D CAD model and its 2D image rendered according to section 2.2.3. To simulate its SAR image, for each rendered pixel we save its original 3D coordinates (x, y, z) along with its intensity in a 2D image (see section 2.2.4 on rendering multiple per-pixel outputs). Using these 3D coordinates, we then place a pixel-wise regular grid geometry on the 2D image, assign each grid node the intensity value of the rendered image at that pixel position and deform each grid node such that it maps to the 2D screen coordinates (x, z). This deformation corresponds to a simplified representation of the two parameters azimuth and distance which govern the SAR imaging process. If several grid nodes map to the same screen coordinates, their intensity values are accumulated at this position.

Figures 2.8, 5.8 show results of the above algorithm as opposed to an industry-grade SAR simulation [66]. The synthetic rendering proves sufficient for the registration task described in chapter 5; it allows avoiding difficulties in obtaining rare and usually classified training and evaluation data.

2.3 Conclusion

In this chapter, we summarized the process of generating synthetic training data using methods from the domain of Computer Graphics. We outlined the choice of the 3D CAD model representation, the synthetic camera parameterization and the lighting model. In the following chapters, we propose approaches to generic object class detection, approximate 3D pose estimation, object instance selection and fine-registration which exploit the advantages of the synthetic 3D training data in order to extend state-of-the-art beyond pure 2D detections.

3

3D Feature Maps for Detection

In this chapter, we present a first approach to object class detection based on synthetic training data and describe the advantages of the CAD models for training and detection. We summarize related work in section 3.2 and describe the training process to robustly derive appearance and 3D geometry representations from synthetic CAD models in section 3.3.1. In section 3.3.2, we provide details on the detection process which builds on a voting scheme and a 3D pose estimation; in section 3.4, the different contributions of the approach are evaluated on various benchmark databases.

3.1 Introduction

Existing work on object detection based on local features can be roughly separated into two groups, i.e., detection of specific objects and of object classes. Numerous approaches propose solutions for viewpoint-independent detection of **specific objects** and significant progress has been made recently [1, 51, 59, 75, 98]. Some approaches build multiview representations of an object instance from real images; for example Lowe [59] harvests local features from real images, forms groups of local features in 2D image space belonging to same viewpoints and derives a probabilistic formulation to identify viewpoint-consistent groups of features in new images. In [98], a 3D model is build from a sequence of images of an object instance without requiring viewpoint annotations by describing the appearance of local image patches, successively matching them to other images in the sequence and identifying common geometric constraints. A hybrid approach is suggested in [51] who formulate the detection as a classification task and train a tree-based classifier by sampling local patches from real images and synthesizing small viewpoint distortions to increase robustness. Synthetic textured 3D models of specific object instances have been used in [1] to harvest local features from rendered views; by exploiting the available 3D geometry, the harvested features can be geometrically annotated and used for robust detection and pose estimation. Najafi et al. [75] combine a synthetic 3D model geometry with aligned real images which allows to learn appearance distributions of local features from real and synthesized views; during detection, starting with an initial feature match a set of viewpoint-consistent features and the object pose can be determined.

In contrast to the detection of specific object instances, methods for detecting **generic object classes** have to handle significant intra-class variations in addition to multiple viewpoints. In recent

years, multi-view generic object class detection has received increasing attention [10, 22, 102, 114, 134]. Most approaches address the task by extrapolating known strategies from 2D single-view object class detection, notably by combining classifiers for separate viewpoints [20, 48, 117]. Some authors have proposed to include weak geometric information into the learning process, mostly by applying locally deformable 2D models for discrete viewpoints [10, 22, 29]. Very recently, more sophisticated methods started to make use of a 3D model structure [101, 134]. Learning such a generic representation of the 3D geometry of an object class is challenging and remains an active research topic [2, 102, 114]. However, even these approaches only determine 2D regions of interest in the image plane as the localization output. Yet, in many cases a 3D pose estimation of the detected generic object would be useful in terms of specific applications, for example for the localization of robots, as well as to improve 2D detections, for example by resolving occlusions or exploiting geometric scene priors.

The advantages of a 3D representation for multi-view object class detection are obvious: 2D detections can be disambiguated and pruned with respect to their consistency with the object class geometry under full perspective projection, and detection confidence can be computed per-object instead of combining per-classifier scores. Furthermore, such a representation allows an approximate estimation of the pose.

In this chapter, we describe an approach to viewpoint-independent object class detection which does provide information on the 3D pose of the detected object. Unlike most recent approaches, we do not build a 3D model from 2D images and their geometric constraints, but resort to a database of existing, fully textured synthetic 3D models to compute a robust 3D representation for each object category, thereby facilitating viewpoint-independent recognition. The local features obtained from rendered synthetic objects have to be selected during training (section 3.3.1) in order to be suitable for a reliable matching to real image features. Figure 3.1 illustrates the detection steps (section 3.3.2). Local features from real images are matched to the synthetically trained ones. Each match casts votes to determine the most likely class and 3D pose of the detected generic object. The most promising votes are then evaluated and refined with respect to their geometric consistence with the 3D model using a robust pose estimation step (section 3.3.2.2). In section 3.4, we present experimental results on the 2006 PASCAL datasets for motorbike and car models [18] and the 3D Object Category dataset CAR [102] and analyze the precision of the 3D pose estimation using a calibrated scenario.

3.2 Related Work

A survey of related work on multi-view object class detection shows three predominant approaches which differ in their choice of the geometric representation. 2D detectors can be combined by linking them over multiple viewpoints in order to achieve some degree of viewpoint invariance [48, 101, 117] and modeling flexible spatial layouts of part detectors [10, 22, 24, 29]. Other methods have been proposed which build 3D representations of the object class from 2D training data based on initial viewpoint annotations [42, 2, 102, 114]. As a third approach, the use of existing 3D models has been suggested in the past [25, 57, 79, 112, 130] and more recently in [39, 134].

Dynamically built representations for viewpoint-independent object recognition have been proposed in the field of face detection, where several authors deal with multiple viewpoints by

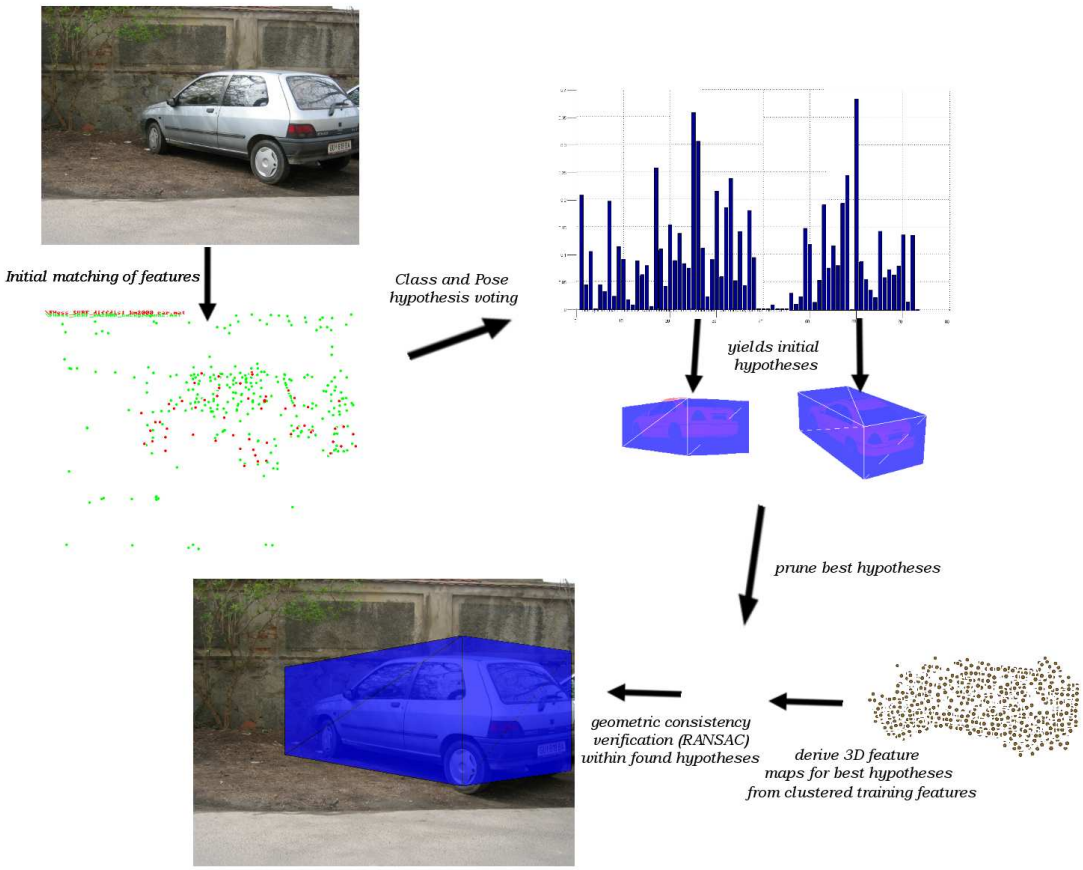

Figure 3.1: Overview of the proposed detection approach.

combining the results of separate single-view detectors ([20, 77, 127]). Fan and Lu [20] tackle multi-view detection by treating each discrete viewpoint as a class and discriminatively learning a multi-class SVM. To account for the fact that the amount of appearance variation differs for each section of the viewsphere, [77] adapt the training data of each discrete SVM to reduce the overall risk of misclassification over all viewpoints. For general object categories, the combination of 2D detectors to cover a multi-view sphere has also been the initial step towards a more comprehensive use of geometry for object class detection: Thomas et al. [117], for example, suggest linking Implicit Shape Models for specific viewpoints amongst each other, thereby achieving a detection over multiple viewpoints at the cost of an expensive training process on manually segmented viewpoint-specific examples.

In order to increase robustness towards pose changes, additional probabilistic layout models as well as local 2D geometric constraints have been introduced in combination with increasingly powerful object part representations and learning procedures. These probabilistic layout models were originally introduced in [24], who model a flexible geometry with interaction forces between different subparts. The idea is taken up by [10] who introduce a simplified layout which assumes a set of mutually independent branch parts which only depend on a few root parts instead of modeling all pairwise interactions. The approach is further extended in [22] with discriminatively

learnt part appearance, different heuristic layout models for the main viewpoints and a root part per viewpoint which covers the entire object, thereby increasing robustness. In [50], Implicit Shape Models are defined as clustered local features with annotations of object scale and centers; during detection, these features cast votes according to the annotation to allow identification of a set of consistent parts. Kushal et al. [48] enforce local geometric constraints between Partial Shape Models which are dense locally rigid assemblies of image features, thereby achieving robustness to viewpoint changes as well as better 2D localization performance. Hoiem et al. [42] suggest a Layout Conditional Random Field to model part interactions for a set of discrete viewpoints from the pixel level upwards. In [83], additional classifiers on selected levels of spatial pyramids are learnt to obtain orientation estimates. Instead of modeling sparse sets of parts, a fixed grid-based subdivision of object views has been suggested in [29] who propose a greedy algorithm to detect regions of parts which conform to the training part layout. However, all these approaches are inherently limited to a few discrete viewpoints as detection output.

Alternatively, viewpoint-annotated training data can be used to dynamically build 3D representations to better address the possible viewpoint variations of object classes. Savarese and Fei-Fei [101] determine homographies of groups of local features in order to map large 2D image regions onto a collection of near-planar parts to form a viewpoint-independent 3D model. In [102], homographic constraints between groups of object parts are combined to form a piecewise planar 3D approximation of object classes which also allows to interpolate unseen instances within the chosen parameterization. More recently, [114] introduced a probabilistic approach to learning affine constraints between object parts; during testing, they rely on a complex sequence of random forest classifiers, Hough voting and linear SVM classifiers. In [2], sparsely annotated 2D feature positions are factorized to obtain a 3D implicit shape model which extends the original implicit shape model to 3D transformations and occlusion issues. Although these methods perform well, their training process is elaborate and they rely on relatively sparse object part representations which may impact their robustness for unseen objects. Instead of building a 3D representation, the inverse approach has been suggested in [132] by using the identified geometric constraints between images of separate views to warp all views into a single 2D unfolded representation.

An alternative lies in using a database of existing 3D CAD models, given their ubiquitous availability and increasing realism in recent years. In the late 1980s and 1990s, this idea has already been advocated by several researchers. Some authors resorted to flexibly aligning groups of consistent edge segments by probabilistic matching [57], to building relational graphs from CAD models to identify characteristic object features [35] or extracting symmetry properties of geometric primitives from CAD models and describing them as permutation groups [25]. Others performed geometric indexing based on invariants, typically curvature or edges, which cast votes into a hash table of potential object poses [79, 112, 130]. Many approaches include an additional step to verify and refine the geometric consistency of the most likely hypotheses [34, 35]. The majority of these solutions rely on geometric primitives which are in general not sufficiently robust and discriminant for generic object categories. Recently, Yan et al. [134] addressed this issue by collecting patches from 2D images with 3D viewpoint annotations and mapping these patches onto an existing 3D CAD model. However, the suggested texture mapping of 2D patches onto a single 3D model is prone to cause artifacts in the appearance representation. Moreover, a single 3D model is in general not sufficient to capture the geometric variations within an object category. Heisele et al. [39] generate difficult training sets from synthetic 3D models for an active

learning algorithm. While being significantly simpler to train, these methods suffer from rendering artifacts and do not systematically exploit the 3D geometry available during training for a 3D pose estimation at detection time. The work of [119] differs from these approaches in relying on motion-based segmentation of object silhouettes from videos which are matched to silhouettes of synthetic models.

In this chapter, we build on the idea of using existing 3D CAD models. However, we do not rely on geometric features for matching, but use a vocabulary of photometric descriptors which are more discriminant and are shown to match to real images. In addition, the available 3D geometry of the object category models is systematically exploited to achieve viewpoint-independent state-of-the-art 2D object class detection results and an approximate 3D pose prediction.

3.3 Our Approach

The proposed approach relies on a database of existing, fully textured synthetic 3D models to compute a robust 3D representation for different object classes. During training, the local features obtained from rendered synthetic objects are filtered (section 3.3.1) in order to be discriminative for their respective object class and viewpoint and to be suitable for a reliable matching to real image features. A global 3D representation, consisting of a codebook of clustered features which were retained in the filtering step and their 3D locations on the training object surfaces, is build (see figure 3.4). Figure 3.1 illustrates the subsequent detection steps (section 3.3.2). Local features from real images are matched to the synthetically trained ones (figure 3.1, top left). Each match casts votes to determine the most likely class and 3D pose of the detected generic object (figure 3.1, top right). The most promising votes are then evaluated and refined with respect to their geometric consistence with the 3D model using a robust pose estimation step (figure 3.1, bottom).

3.3.1 Training

In this section, we describe the unsupervised training process which builds on synthetic 3D CAD models and suitably selected local features in order to build a data representation capturing 3D geometry and local appearance of an object class.

3.3.1.1 Training Data

In chapter 2, we discussed the use of synthetic 3D CAD models as a possible solution to generating training data for object class detection. In the present chapter, we outline an approach which uses such synthetic training data (as shown for example in appendix B); more specifically, we train on subsets of the CAD model databases for the classes car and motorbike; see section 3.4.1 for details on the chosen subsets. We follow the rendering method described in chapter 2 to generate training images with precise 3D viewpoint annotations for different camera parameterizations as specified in section 3.3.1.3. As discussed in chapter 2, by choosing to train our detector on rendered views of synthetic 3D models, we circumvent both instable training conditions and complex model-building at the cost of a possibly reduced descriptor similarity.

3.3.1.2 Local Features

Our approach is based on local features which are extracted from rendered views of synthetic 3D models. The performance of our object class detector, as is the case with any other feature-based detector, depends heavily on descriptor similarity. In our case, it depends on the ability to establish correspondences between descriptors extracted from synthetic images and from real images.

Here, we use the FastHessian feature detector in combination with the SURF descriptor [5]. Experimental results show that this image description allows to match synthetic with real images when combined with a discriminative filtering step, see section 3.4. Unlike the standard parameters suggested by [5], our detection uses a smaller sampling step of one pixel and the descriptor is used in the extended 128-dimensional upright (i.e., not rotation invariant) version.

3.3.1.3 Model Acquisition

A truly viewpoint-independent object detector would require training an object representation which continuously covers the entire camera view sphere. To reduce the complexity of the problem, we resort to a discrete representation where gaps in the view sphere are bridged by the invariance of the local features.

First, all our 3D models are scaled to fit into a unit bounding sphere and they are oriented along their dominant dimension. For each model, its minimum-volume enclosing rectangular bounding box is computed from its mesh geometry. To further simplify the problem, we determine the average ratio of the bounding box dimensions (length, width and height) within each object class, resulting in a single scale parameter to represent a class-specific bounding box. Each model is now rendered from a discrete number of viewpoints, characterized by the following conditions as depicted in figure 3.2:

1. The camera is positioned at $\lambda = (a, e, d)$ with a, e, d being azimuth, elevation and distance, looking at the world coordinate origin at (0,0,0).
2. The model is rendered with its bounding box centered at the origin, called the *0-pose*.
3. The distance between object and camera is varied at discrete steps in order to account for the fact that the scale invariance of the features is limited in practice.
4. The orientation parameters azimuth and elevation run through a set of discrete values.

The choice of the discretization has to take into consideration the degree of invariance offered by the chosen local feature descriptors as well as the viewpoints which are to be encountered by the detector when working on real images. See section 3.4 for the specific discretization used in our experiments and chapter 2 for details on the synthetic data generation.

A distinct bounding box pose for a given viewpoint can now be described by the three-dimensional parameter set $\lambda = (a, e, d)$ with a, e, d being azimuth, elevation and distance of the camera pose (directed per definition towards the world origin). The parameter set λ forms the *object hypothesis*. These hypotheses are all relative to the internal virtual camera calibration matrix K chosen for the rendering step. As a consequence, the 3D pose estimation provided by our method will be relative to the same internal camera parameters; our method only provides an estimation of the external camera matrix $V(\lambda)$.

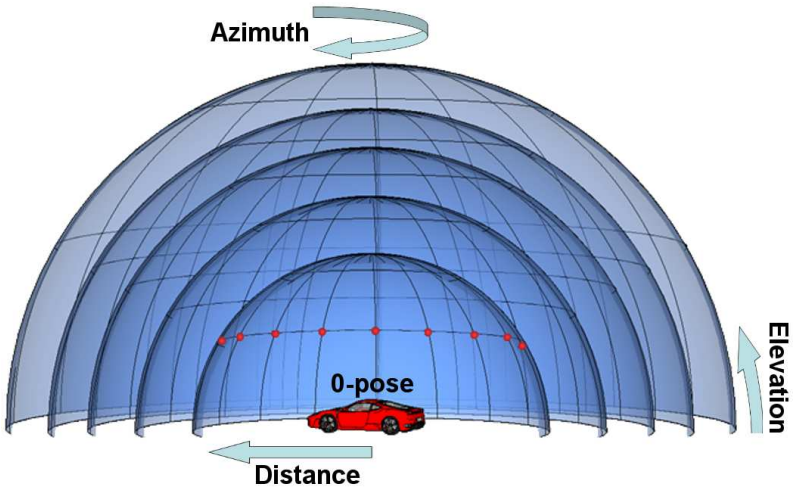

Figure 3.2: Discretization of the camera parameters azimuth, elevation and distance during training.

For each object hypothesis, we then collect local features. Each feature is annotated with the object hypothesis describing its originating viewpoint and bounding box along with a weight which corresponds to the inverse of the number of features found under this viewpoint. This weight is necessary to balance each viewpoint's contribution, since for example profile views of an object typically cover a larger image surface and therefore result in significantly more features than frontal views. In addition, each feature stores its 3D position relative to the model geometry in the normalized object coordinate system. This 3D position, which corresponds to the feature location in the image after backprojection onto the object geometry, later allows for a 3D pose estimation; see section 2.2.4 on how these 3D positions can be obtained from the synthetic data. We term these groups of 3D-annotated features *3D Feature Maps*, since they contain all the information necessary to roughly reconstruct in 3D the object hypothesis from which they originated.

3.3.1.4 Discriminative Filtering

The local features should be discriminant with respect to the object category and each discrete viewpoint. At the same time, the features have to be invariant towards small local pose variations in order to bridge the gaps between the discretely sampled training viewpoints; moreover, they have to be robust in the presence of background. This can be achieved by a discriminative filtering procedure similar in spirit to [1, 51]. In section 3.4, we show the importance of the discriminative filtering for the detection performance. Filtering consists of the following steps, see figure 3.3 for an illustration:

1. Each training object is rendered once with the exact viewpoint parameters in front of a white background; local features are collected for the rendered image, in the following identified as the *default feature set*.

2. The training object undergoes a sequence of slight variations of each of the three pose parameters, typically covering $\pm 1\%$ of its respective parameter space, leading to 3^3 evaluated parameter combinations.
3. For each pose variation, the object is rendered three times, in front of a white, a real and a synthetic background (see figure 3.3 for a visualization of the idea and 3.4 for an example.).
4. Each pose variation and background yields local features which are matched to the *default feature set* w.r.t. descriptor distance and 3D position distance after backprojection into 3D world coordinates. The features of the default set are weighted according to the number of matches for each pose and background variation, thus giving a higher importance to the more discriminant and robust ones.

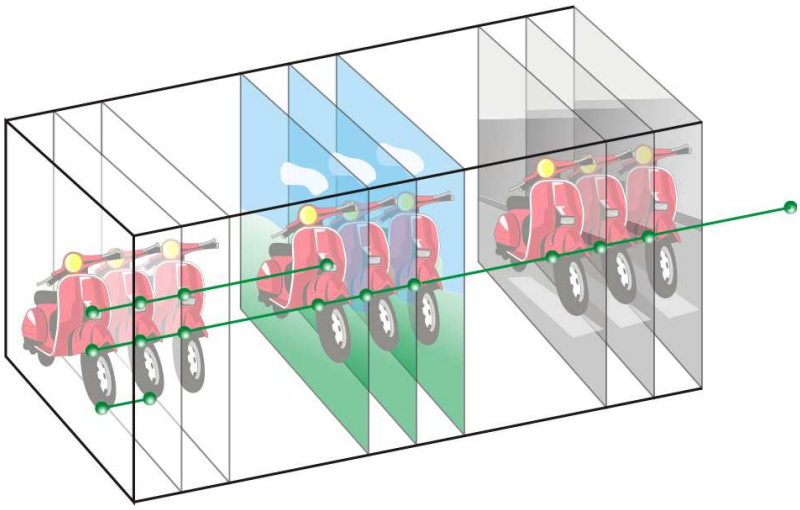

Figure 3.3: Discriminative filtering of the features during training. The features are weighted according to their stability w.r.t. different backgrounds and small local pose variations.

The discriminative filtering could in principle be extended in the same way to more rendering parameters, such as lighting and imaging conditions.

Figure 3.4: Different steps of discriminative filtering for one pose: initial features (left), intermediate step (center), final result (right).

To reduce the complexity of the approach during matching to real images, the number of features has to be reduced as early as possible in the processing chain. We, therefore, train a simple two-class *Support Vector Machine* (SVM) classifier on the synthetic object features harvested during training and a real background feature set in order to differentiate between relevant object and

irrelevant background descriptors. The SVM classification step further avoids having to compute pairwise distances between the model features and those image features which are unlikely to be located on an object; this is efficient since the number of support vectors necessary to classify background features is typically small compared to the number of model features with which we would otherwise have to exhaustively compute pairwise distances. *Support Vector Machines* are a standard machine learning approach; for details, see appendix C. For the above task of separating object class relevant features from those likely to belong to the background, we rely on a *radial basis* kernel defined by

$$k(x_i, x_j) = exp(-\gamma \|x_i - x_j\|^2) \tag{3.1}$$

and determine the parameters C, β, γ by an exhaustive grid search with fixed step size.

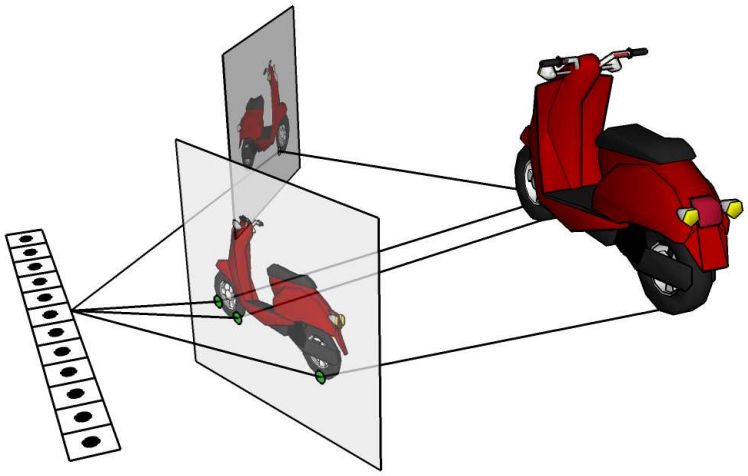

Figure 3.5: Each codebook entry stores the mean descriptor and the 3D positions of all the similar features which form a cluster.

3.3.1.5 Codebook Construction

Similar to many existing approaches [28], we construct a visual codebook of k elements by clustering the harvested discriminant descriptors with the standard k-means algorithm [61].

Each cluster stores a list of the discrete poses of the features which contributed to this cluster, along with their viewpoint-specific weights (see section 3.3.1.3). In section 2.2.3, we described how the rendering pipeline allows to associate image pixels with their originating 3D points on the model surface. Here, we exploit this functionality to recover the 3D position of each local feature retained after the discriminative filtering. From the 3D positions of all the training features which are merged into a single codebook cluster, we build a data structure consisting of multiple linked lists per discretized training viewpoint. This data structure allows to quickly recover all potentially visible 3D feature positions of a given cluster under a given viewpoint. The information is later used as input for the geometry verification: for each viewpoint hypothesis, we can directly

access the subset of all those codebook entries which occurred under a given viewpoint during training without having to exhaustively search through the entire codebook. Figure 3.5 shows an example cluster with the links to the 3D positions of its contributing features. Without preserving the full 3D position origins of each cluster contribution, the geometric structure of the codebook would loose its consistency when the number of training models is increased; figure 3.6 compares the geometry layout of a feature map for the motorbike class when neither discriminative filtering (see section 3.3.1.4) nor clustering of the descriptors is performed and each descriptor is stored unchanged with its original 3D position (left), when k-means clustering on discriminatively filtered features is performed and a weighted average of the 3D positions is computed based on descriptor distances within each codebook entry (center), and when all 3D positions are retained in the above described linked lists after filtering and clustering (right). The unfiltered and unclustered representation retains many outliers, the clustered feature map with averaged positions has a tendency of smoothing out many of the potentially important geometric characteristics such as handle bars, while the proposed feature map offers a tradeoff between the removal of outliers and the preservation of geometric structure.

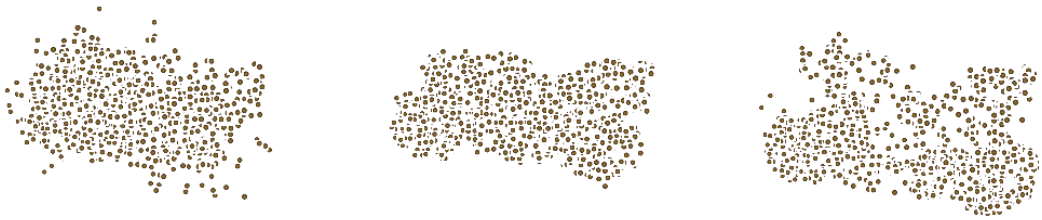

Figure 3.6: Comparison of the geometric consistency of differently constructed feature maps for the motorbike class; unclustered without discriminative filtering (left), clustered and weighted 3D position average (center), clustered and 3D position lists (right).

3.3.2 Detection

In this section, we outline the detection process, consisting of a voting to establish initial pose hypotheses and a consistency check based on a full 3D pose estimation; we compare the performance of three different estimation approaches for this task.

3.3.2.1 Pose Hypothesis

Figure 3.1 provides an overview of the different steps of the detection process. During detection, local features are extracted from the image and matched with at most the n closest codebook entries, provided that the matches fulfill the nearest neighbor distance ratio (*NNDR*) criterion [67]; in our experiments, we use $n = 5$. Next, votes are cast for the respective pose parameters. Each

cluster casts votes for the voting bins of the discrete poses contained in its internal list. The interpretation of the voting result can be expressed as follows:

Each extracted feature descriptor f corresponds to a codebook entry c_j with probability

$$p(c_j|f) = 1.0 - e^{-(d_b(f)/d(f,c_j)-1)} \tag{3.2}$$

where $d(f, c_j)$ is the descriptor distance of f and c_j, and $d_b(f)$ is the distance of f to the closest cluster different from c_j. For each codebook entry, we can derive the distribution of the parameter sets $\lambda = (a, e, d)$ from the training data as $P(\lambda|c_j)p(c_j|f)$. The vote of each match in favour of an object hypothesis λ then has the weight

$$p(\lambda|f) = \sum_{j=1}^{n} P(\lambda|c_j)p(c_j|f). \tag{3.3}$$

Note that the 2D location of the extracted feature does not contribute to the voting weight. It is only used in the subsequent geometry verification step for a 2D-3D pose refinement.

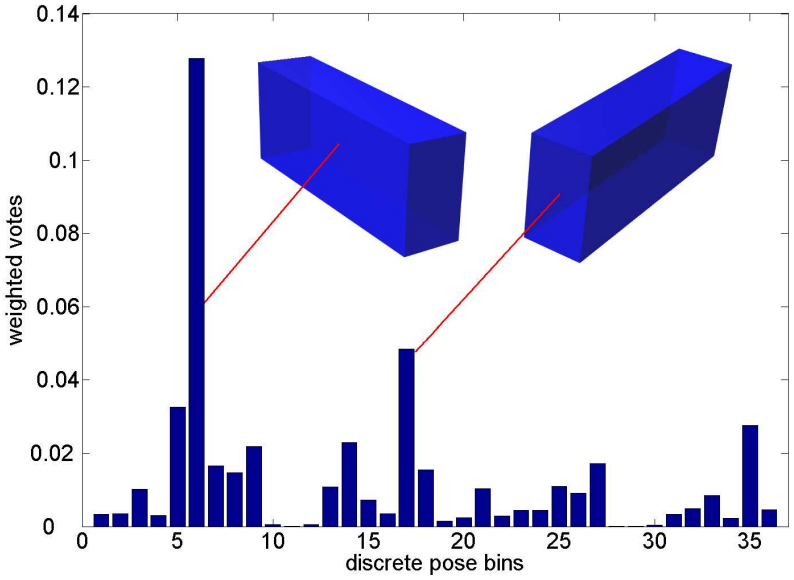

Figure 3.7: Histogram of the votes cast by the matched features into the discretized pose bins. The bounding boxes illustrate the poses corresponding to the two local vote maxima. Symmetric object orientations yield similar features. For simplification, only the azimuth pose bins for a single elevation bin are shown.

Figure 3.7 visualizes the result of voting within one class-specific voting space. The bin with the maximum sum of votes over all features indicates the most likely object hypothesis λ in the *0-pose* (see section 3.3.1.3). We perform a non-maxima suppression within the voting space and retain the maximum votes as the potential pose hypotheses. Note that the voting cannot distinguish between symmetric orientations which yield nearly identical feature distributions. These ambiguities have to be resolved in the pose refinement step described in the next section.

If multiple object instances in similar poses are present in the image, they will cast votes into the same bins. These ambiguities will be resolved in the following steps, described in the next sections, by exhaustively evaluating the consistency of groups of features with the training model geometry. If a group of consistent features of a pose has been found, they will be removed from the set and the process will be restarted on the remaining features of this pose until no more consistent feature groups are found.

3.3.2.2 Pose Consistency

Due to incorrect descriptor matches, ambiguous symmetric poses, multiple object instances in similar poses, background clutter and limited feature invariance, the pose hypotheses typically contain geometrically inconsistent results as well as overlapping or multiple detections of the same object.

To separate the correct from the inconsistent hypotheses, we perform an approximate pose estimation to determine the number of matches which are consistent with the model geometry. From the test image, the previous voting step generates a set of pose hypotheses which are associated with sets of *image features*, each consisting of a feature descriptor and a 2D image position. From the training step, the codebook contains *model features*, each consisting of the descriptor of the cluster it belongs to and a list of the 3D positions on which the cluster feature occurred during training; the *model features* are further linked to their training viewpoints. For a given pose hypothesis, we can now recover all model features associated with this pose during training and match them to their closest image features. However, these tentative matches can contain outliers and ambiguities; by performing a pose estimation, inconsistent matches can be pruned and the pose hypothesis from the voting step can be evaluated in terms of the similarity of the pose-consistent features.

The subsequent pose estimation consists of iteratively executing the following steps to determine the extrinsic camera parameters; it assumes a known camera calibration (i.e. known intrinsic parameters) determined by the virtual camera used to render the 3D training models.

1. **Feature correspondences**: Establish a set of potential correspondences between subsets of model and image features based on minimizing a distance function (see equation 3.4). Initially, the distance function relies on descriptor distances alone; in the following iterations, the descriptor distances will be used as weights to the geometric distances, either in image or in model space.

2. **Pose Estimation**: Assuming the previously established pairs of model and image features, sorted in terms of the distance function, robustly estimate the projective transformation which maps the set of 3D model points onto the 2D image features, minimizing the overall combined geometric and descriptor distance between pairs of corresponding 3D and 2D features. We consider three different approaches to this problem.

3. Based on the estimated projective transformation, return to step 1 and refine the correspondences based on the geometric distances resulting from the current projection, until the pose change between two iterations is below some threshold.

Note that in order to keep computation times feasible, the solutions within one pose estimation iteration are evaluated only in terms of the preestablished feature correspondence pairs; the full correspondence matrix is recomputed only after a pose estimation iteration has converged or exceeded a fixed number of evaluations.

3.3.2.3 Feature correspondences

During training, we harvested local features together with all their 3D positions on the surface of the training models and associated each feature to a codebook cluster. During testing, we are given a set of local features in the test image together with their 2D positions in image space. As described in section 3.3.2, initial matches between features from the codebook and the test image are established based on their assignment to the same codebook clusters, not taking into account their geometric distance. In order to perform a pose estimation for verification and pruning of the initial voting hypotheses, the quality of potential pose estimations has to be evaluated in order to choose the best suited estimation result and establish a ranking of the detections for assessing the overall performance on a given test database. This quality measure should be based on descriptor and geometric distance to account for appearance similarity as well as geometric consistency. Consequently, we introduce a combined distance measure between a 3D feature from the training database c_j with its associated list of n 3D positions $P_j^{3D} = \{p_{j,0}^{3D} \dots p_{j,n-1}^{3D}\}$ under a perspective projection Φ_λ and a feature descriptor f with its 2D position p^{2D} extracted from the test image as

$$d_{combined}(c_{j,i}, f) = max(1 - p(c_j|f), \gamma)(\Phi_\lambda(p_{j,i}^{3D}) - p^{2D})^{2\delta} \tag{3.4}$$

where γ is the minimum contribution of the geometric distance per correspondence; in the experiments, we use $\gamma = 0.1$. Since the voting step provides only a discretized pose hypothesis, it cannot initially be used to project 3D features into 2D image space; consequently, δ is set to 0 in the first iteration of the correspondence estimation and to 1 in the subsequent iterations when precise pose estimations are available. The descriptor distances can be precomputed and remain constant during the pose refinement iterations. A distance matrix D can now be assembled, containing in each row $r_{j,i}$ the m distances $d_{combined}(c_{j,i}, f_k)$ between a 3D feature c_j at 3D position $p_{j,i}^{3D}$ with all 2D features f_k for $k \in 0..m-1$ found in the test image.

The task of establishing consistent correspondences between pairs of 3D and 2D features to be used in the subsequent pose estimation can now be formulated as an optimal assignment problem which can be solved with a dynamic programming approach on the above distance matrix D as described in [73] as a refinement of [47]. We suppose that each 2D feature found in the test image can be explained by at most one model feature at a distinct 3D position in the feature map; the quality of such an assignment is given by the corresponding entry in the distance matrix D. The 2D features contained in the image have to be assigned to a subset of model features from the feature map such that the sum of the distances of all assignments is minimal, i.e. given matrix D, the objective is to choose a permutated subset π of length m of its rows $r_{j,i}$ such that $\sum_{k=0}^{m-1} D(\pi(k), k)$ is minimal. The subsequent pose estimation step will then try to find a perspective projection which consistently maps as many of the model features in this subset onto their assigned 2D features as geometrically possible. Both steps will eliminate outliers in terms

of the distance function $d_{combined}$ and in terms of the geometric consistency with a perspective projection.

The algorithm of [73] solves the assignment problem in polynomial runtime by iteratively subtracting the smallest entry per row in the distance matrix D from all other row elements until a set of zeros has been determined which covers all columns of D, thereby constituting an optimal assignment; for a detailed algorithm outline and runtime analysis, see [73]. The algorithm is guaranteed to converge since in each iteration, the (positive) entries of the matrix are decreased until they are zeroed out which only allows for a finite number of iterations.

3.3.2.4 Pose Estimation

We evaluate three different approaches to the pose estimation problem for known intrinsic camera parameters and compare their performance and suitability for the overall task. All three pose estimation approaches rely on a set of assignment pairs between 3D model and 2D image features as outlined before. Since the voting step 3.3.2 provides a pose hypothesis with a discretized parameter set, the refined pose estimation can be constrained to the pose parameter subspace associated with a discretized pose hypothesis, thereby allowing to exclude unlikely pose estimation results and increasing robustness.

The pose refinement generally allows for the detection of multiple object instances present in an image, since each of the locally maximal hypothesis votes will be evaluated. For object geometries similar to those of the training models, an object instance will typically yield a single refined hypothesis. Since the chosen 3D representation contains all 3D locations under which a given feature cluster was found during training, the geometry matching can accommodate for variations of the object geometry. In case of significant deviations from the trained geometric configurations, a single object might result in several pose estimations of its subparts, differing only slightly in translation and scale; our method detects these cases and combines them into a single hypothesis with an extended bounding box. Occlusions are handled implicitly: as long as the visible part of the object yields enough geometrically consistent feature matches, the correct object pose will be found; see section 3.4.2, figure 3.15, bottom left, for an example.

In the following sections, we describe and compare the three pose estimation approaches.

3.3.2.4.1 Robustly Sampled Closed-Form Perspective 3-Point

This approach does not use the voted-for pose hypothesis as initialization, but only as a constraint to post-filter estimation results not covered by the voted hypothesis bin; for details on the chosen discretization, see 3.4.1. Pose estimations outside the initial hypothesis bin are considered negative detections, assuming that the corresponding features found in the current image are not consistent with the trained model geometry.

Matching pairs of model and image features are sorted according to their descriptor distances and fed into a RANSAC loop [23]. Inside the RANSAC loop, on each subset of three 3D-2D model-image feature pairs a closed-form perspective three-point (P3P) method [23, 37, 129] estimates the extrinsic camera parameters which project the model features onto the paired image features.

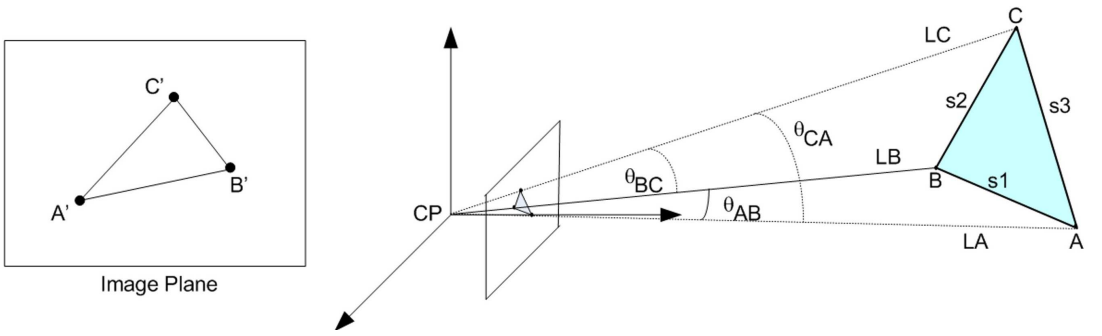

Figure 3.8: Approach to the perspective three-point problem by solving for the lengths of the three sides of the tetrahedron formed by the camera center CP and the 3 model points A,B,C; illustration in [129].

The P3P is based on the intrinsic parameters of the virtual camera which has been used to render the models in the training step.

To solve the perspective three-point problem, first the lengths of the three sides of the tetrahedron formed by the camera center CP and the 3 model points A,B,C are computed as suggested in [23]; see figure 3.8. Then the orientation of the 3 model points A,B,C in their original coordinate system is determined using the method of [44] with modifications suggested by [121].

From figure 3.8, assuming known intrinsic camera parameters that in the present case are given by the virtual camera used for rendering the training models as outlined in chapter 2, and given the lengths of the tetrahedron base sides $s_1,\ s_2,\ s_3$ and the angles $\phi_{AB},\ \phi_{BC},\ \phi_{AC}$ of a known image, the cosine rule allows to derive the following equation system:

$$\begin{aligned} s_1^2 &= L_A^2 + L_B^2 - 2L_A L_B \cos(\phi_{AB}) \\ s_2^2 &= L_B^2 + L_C^2 - 2L_B L_C \cos(\phi_{BC}) \\ s_3^2 &= L_A^2 + L_C^2 - 2L_A L_C \cos(\phi_{AC}). \end{aligned} \tag{3.5}$$

Several approaches to solving this system of equation have been discussed in the literature; we follow [23] to obtain a quartic polynomial which we then solve by computing the eigenvalues of the associated companion matrix [15]. The problem can have a maximum of eight solutions, but four solutions result in negative tetrahedron side lengths corresponding to a virtual object position behind the camera and can thus be discarded; the remaining four solutions represent four possible projective configurations; for a detailed review of the possible solution configurations, see [23, 37, 129]. Based on the known intrinsic camera parameters corresponding to the virtual camera used to render the 3D models during training, the 3D positions of the model points can be computed in camera coordinates. It remains to determine the transformation which aligns the model points from camera coordinates to their original frame of reference which is equivalent

to solving the absolute orientation problem [44, 121]: given two sets of 3D points $x_{i=1,2,3}$ and $y_{i=1,2,3}$, solve for the rotation R_o and translation t_o such that

$$(R_o, t_o) = \underset{(R,t)}{\operatorname{argmin}}(\sum_{i=1}^{3} \|y_i - (Rx_i + t)\|). \tag{3.6}$$

The solution in least square sense which guarantees the orthonormality of the resulting rotation can be found via a singular value decomposition of the covariance matrix of the data, taking the signs of the eigenvalues into account to avoid obtaining a reflection instead of a rotation [121].

The best of the four solutions is selected as that which maximizes the number of inliers in terms of the combined descriptor and geometric distance of the preestablished pairs of corresponding features for the current pose estimation iteration. The RANSAC loop terminates after a fixed number of 3-point samples have been evaluated; see section 3.4 for the chosen parameters.

Note that in this case, we perform neither a non-rigid registration of the model nor an iterative optimization. Instead, the success of the closed-form pose estimation depends on the fact that among all the feature positions identified during training, a minimum of 4 corresponding, geometrically consistent feature positions can be discovered in the input image. Features occurring at positions which have not been trained, cannot be matched either.

3.3.2.4.2 Robustly Sampled Iterative Estimation

As in the three-point estimation, matching pairs of model and image features computed in 3.3.2.4 are sorted according to their descriptor distances and fed into a RANSAC loop. Inside the RANSAC loop, on each subset of three 3D-2D model-image feature pairs an iterative estimation which follows the work of [81] computes the extrinsic camera parameters which project the model points Q_i onto the paired image points p_i. In contrast to the previously described closed-form estimation, it explicitly relies on the voted pose hypothesis as an initialization, which is then iteratively refined.

Assuming a known camera calibration, which in the present case is given by the virtual camera used for rendering the training models as outlined in chapter 2, and a set of n image points $p_{i=1\ldots n}$, the unit length projection ray $v_i = K^{-1}p_i$ of each image point $p_i = (x_i, y_i, 1)$ is defined as the location of all 3D points P_i in camera coordinates which project onto this image point p_i as

$$P_i = \lambda_i v_i = \lambda_i K^{-1} p_i \tag{3.7}$$

where λ_i is a scalar corresponding to the distance of P_i to the camera and K the intrinsic camera matrix. The forward projection chain for some set of 3D points Q_i into an image consists of a transformation of the 3D points Q_i into the camera coordinate system with some rotation R and translation t such that

$$\hat{Q}_i = RQ_i + t \tag{3.8}$$

and the subsequent projection of the transformed points $\hat{Q}_i$ along the rays v_i into the image. In the case of pose estimation, the projection chain is inverted, i.e. the depths λ_i, rotation R and translation t are chosen to minimize the distance function

$$(\lambda_i, R, t) = \underset{(\lambda_i,R,t)}{\operatorname{argmin}}(\sum_{i=1}^{n} \|\lambda_i v_i - (RQ_i + t)\|^2) \tag{3.9}$$

for known 3D model points Q_i. The iterative pose estimation procedure recovers these unknowns as follows:

- Assume a known initialization for the rotation $\tilde{R}$ (which in the present case is given by the pose hypotheses of the voting step) and compute estimated $\tilde{Q}_i = \tilde{R}Q_i$.
- Determine estimations $\tilde{\lambda_i}$ for each image point p_i from

$$(\tilde{\lambda_i}, \tilde{t}) = \underset{(\lambda_i,t)}{\operatorname{argmin}}(\sum_{i=1}^{n} \|\lambda_i v_i - (\tilde{Q}_i + t)\|^2). \tag{3.10}$$

  Partial differentiation gives

$$\begin{aligned} \tilde{\lambda_i} &= v_i^t(\tilde{Q}_i + t) \\ \tilde{t} &= -\left(\sum_{i=1}^{n}(I - v_i v_i^t)\right)^{-1}\left(\sum_{i=1}^{n}(I - v_i v_i^t)\tilde{Q}_i\right) \end{aligned} \tag{3.11}$$

  where I is the identity matrix.

- Align the two point sets $P_i = \tilde{\lambda_i} v_i$ and $\tilde{Q}_i$ in least-square sense such that

$$(dR, dt) = \underset{(R,t)}{\operatorname{argmin}}(\sum_{i=1}^{n} \|P_i - (R\tilde{Q}_i + t)\|^2) \tag{3.12}$$

  This step corresponds to the absolute orientation problem [44, 121] described above as part of the 3-point estimation procedure.

- Update the $\tilde{Q}_i$ by the incremental transformation defined by (dR, dt): $\tilde{Q}_i^{\{T+1\}} = dR\tilde{Q}_i^{\{T-1\}} + dt$.
- Iterate until the change in the $\tilde{Q}_i$ is below some threshold.

Note that the above procedure distributes the estimation of the translation into the two steps (3.10) and (3.12) which increases robustness of the first step (3.10). The projection error of the selected subset of paired 3D model and 2D image points is computed to assess the quality of the current RANSAC iteration. The RANSAC loop terminates after a fixed number of 3-point samples.

3.3.2.4.3 Fully-Paired Iterative Estimation

The approach is based on the previously described robustly sampled iterative estimation, but does not wrap the estimation step inside a RANSAC loop. Instead, the iterative estimation initially uses the k_0 best-scoring 3D-2D model-image feature pairs in terms of the descriptor distances of equation 3.2. In each estimation iteration j, all k_j pairings are used in the optimization; after each iteration, the quality of the pairings is reevaluated with the combined descriptor and geometric distance $d_{combined}$ in equation 3.4, where the geometric distance is based on the pose estimation of the current iteration. Pairings resulting in projection errors above a threshold are removed. This procedure iteratively prunes the pairings; note that pairings can only be removed in order to guarantee the consistency of all pairs with the previous estimation and achieve a stable convergence.

3.3.2.4.4 Experimental Comparison of the Estimation Methods

In order to assess the estimation precision and robustness, a large number of precisely labeled groundtruth tests is necessary. Since different 3D imaging methods, notably a time-of-flight and a stereo camera, did not provide satisfactory precision, the comparison was performed on 180 synthetically generated images with known 3D geometry. A 3D feature map with $K = 2000$ codebook entries was built from 20 car models, and 10 additional car models not contained in the training set were used to render 18 test images per model in 18 azimuth steps for a single distance of two units and 0 elevation; see chapter 2, figure 2.2, for examples of synthetic test images with background variation. Since the focus of the comparison is entirely on the estimation process, the voting process described in section 3.3.2 was replaced by systematically providing different initial pose hypotheses around the groundtruth pose with their corresponding codebook entries and subsequently evaluating the refined pose after convergence of the respective method; table 3.1 provides details on these pose variations.

Figure 3.9 compares the estimation results of the three methods by their residual distance, centroid distance and angular error relative to the deviation of the initialization hypotheses in terms of the three parameters a azimuth, e elevation and d distance and table 3.2 provides the average errors for each of the estimation methods. It is obvious that the fully-paired iterative estimation is not suitable for this task regardless of the initialization, since the outliers in the correspondence matching result in irrecoverable errors which show clearly in the residuals of the distance function; the position errors are multiples of the object size and the angular errors exceed $45°$. The robustly sampled perspective three-point estimation method performs better and usually achieves alignment errors of less than 30% of the model diameter and angular errors below $1°$; however, it fails in several cases: From the right row in figure 3.9, one can clearly see that it has difficulty in correctly estimating the orientation when the initial elevation is close to the groundtruth elevation, since the object features visible from a camera view with zero elevation usually remain visible when increasing the viewpoint's elevation up to $45°$; consequently, ambiguous elevation estimations can be found which reduce the overall residual distance, but do not correspond to the actual viewpoint elevation. Since the P3P does not use viewpoint initialization values (i.e. only the codebook entries differ according to the initial viewpoint), it cannot resolve these ambiguities. A similar behaviour, although resulting in smaller average errors, can be observed for an azimuth initialization close to the groundtruth (see figure 3.9, center row at 0 initial azimuth). Finally, larger initial distance offsets slightly reduce the estimation performance (figure 3.9, left row). The third method, the robustly sampled iterative estimation, outperforms the previous approaches as can be seen from

the low average errors in table 3.2. Note that the residuals of the distance function are due to background features and variations in the descriptor values resulting from viewpoint changes and the clustering procedure. Furthermore, reduced estimation performance has to be expected on real images due to reduced descriptor similarity and potentially incorrect initial pose hypotheses deviating by more than the values tested above. The convergence speed of the two robustly sampled methods is comparable; for the synthetic tests on a codebook of $K = 2000$ entries and test images of dimension 512×512 yielding $200 - 500$ relevant image features per initial pose hypothesis, the estimation takes approx. $40\,sec$ per image. Convergence of the unsampled iterative method with approx. $2.5\,sec$ per image is faster, but comes with the cost of a significantly lower precision.

Table 3.1: Discrete pose variations relative to each ground truth pose for pose estimation testing.

description	value range	stepping
δazimuth	$-0.8.. + 0.8$[rad]	0.1[rad]
δelevation	$0..0.8$[rad]	0.1[rad]
δdistance	$+0.6.. + 5.5$[units]	0.7[units]

Table 3.2: Average errors of the three estimation methods on synthetic test cases; also see figure 3.9.

Fully-Paired Iterative Estimation			
	initialization offset		
description	δ r	δ a	δ e
residual dist.	750.8623	462.0260	439.8999
3D centroid err. [units]	14.9325	31.5237	31.6720
angular err.[rad]	0.7191	1.0288	0.8402

Robustly Sampled Closed-Form Perspective 3-Point			
	initialization offset		
description	δ r	δ a	δ e
residual dist.	34.2035	33.4687	242.9908
3D centroid err. [units]	0.3196	0.2257	7.6644
angular err.[rad]	0.0125	0.0126	0.2309

Robustly Sampled Iterative Estimation			
	initialization offset		
description	δ r	δ a	δ e
residual dist.	26.4129	30.2583	24.5239
3D centroid err. [units]	0.02	0.02	0.031
angular err.[rad]	0.0026	0.0065	0.0043

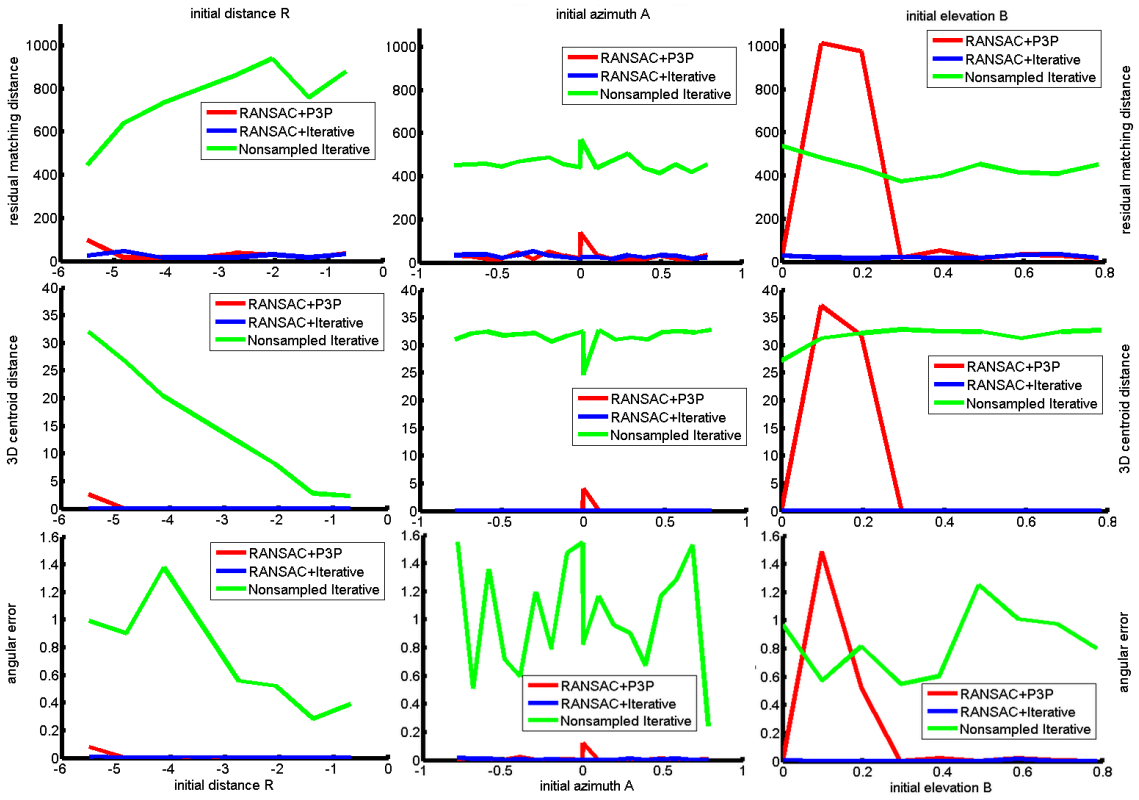

Figure 3.9: Comparison of the pose estimation approaches on synthetic test cases; each row plots one evaluation measure (residual of the distance function, 3D centroid distance, combined angle difference) of the estimation result relative to groundtruth, each column refers to one initialization variation (distance, azimuth, elevation).

3.4 Experimental Evaluation

In order to assess the performance of the complete approach for the task of object class detection as outlined in the problem statement in section 1.1, we apply the approach to different publicly available benchmark data sets, chosen to represent different aspects of the task.

3.4.1 Dataset and Evaluation Criteria

For training, we used 8 synthetic models for the class "motorbike" and 50 synthetic models for the class "car". The models come from different free and commercial CAD model databases, notably turbosquid.com, 3d02.com and doschdesign.com; see chapter 2 for details. The experiments were performed using these two classes of 3D models rendered on the background dataset described in section 2.2.1 in addition to the pure negative training data provided by the respective benchmark data sets. The codebook contains $K = 2000$ clusters per class as described in section 3.3.1.5.

Table 3.3 summarizes the parameter space discretization used for training. We have experimentally found these values to best cover the viewpoints and object poses, given the invariance of the

descriptors used. Both increasing and decreasing resolution resulted in a loss in performance due to less pronounced maxima or less precise pose estimations. Note that for our experiments, we have chosen to include the model scale variation into the camera pose estimation as outlined in section 3.3.1.3.

Table 3.3: Choice of discretization parameters for training

description	values
azimuth a	$0..360°$ in $10°$ steps
elevation e	$0..40°$ in $20°$ steps
object distance d	$4.5, 6, 7.5$ [units]

In order to evaluate the performance of our detector w.r.t. 2D ground truth bounding boxes, we use the detection quality criterion suggested by [18]: For a correct localization, the overlap a_o between predicted bounding box B_p and ground truth bounding box B_{gt} must exceed 50% as defined by

$$a_o = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})}. \tag{3.13}$$

Our 2D localization is created by projecting the 3D bounding box into the image plane and computing the convex hull of the 2D projection of the bounding box corners from which an axis-aligned rectangular 2D bounding box is determined. Please refer to section 1.1 for additional details on the benchmark criteria.

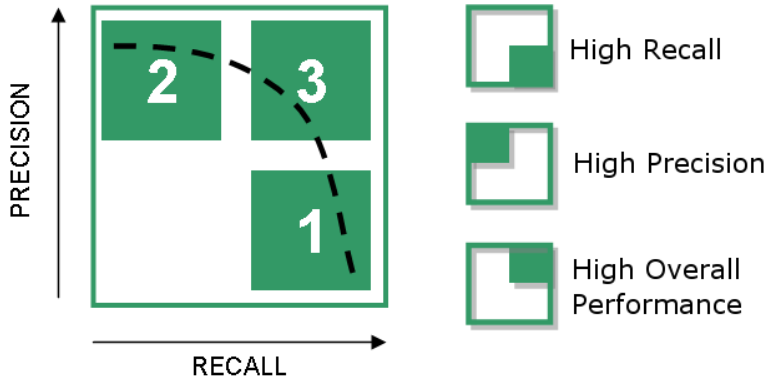

Figure 3.10: Simplified scheme of a precision/recall curve and the different performance criteria for an object class detector. Illustration courtesy K. Schertler.

Since algorithmic approaches to the detection task often depend on a set of parameters governing the algorithm, the detection performance of different algorithms for different parameterizations has to be assessed in terms of the general behavior of the algorithm on a given dataset. In the domains of Information Retrieval and Data Mining, several performance metrics are commonly used for the task. Some authors, for example [93], have compared the suitability of these metrics with regard

to dataset characteristics. For object class detection on the above mentioned benchmark datasets, the distribution of positive and negative samples is skewed in that the number of potential negative detections outnumbers the positive detections by far. Consequently, it has been suggested in the literature that such a problem setting is best evaluated in terms of a precision/recall metric. In the following, we outline the metric, assuming a binary detection outcome which is positive when it matches the groundtruth annotation as defined in equation 3.13, and negative when it fails to match the groundtruth annotation. We term *true positives (TP)* the positive samples correctly detected as positive, *false positives (FP)* the negative samples incorrectly detected as positive, *true negatives (TN)* the negative samples correctly detected as negative, and *false negatives (FN)* the positive samples incorrectly detected as negative. From these definitions, we define *recall* as the fraction of true positive detections among all positive samples in the dataset, i.e. $\frac{TP}{TP+FN}$ and *precision* as the fraction of true positive detections among all positive detections, i.e. $\frac{TP}{TP+FP}$. Since neither term depends on the number of true negatives, a biased representation of detection performance on datasets with a predominantly large number of negatives is avoided. The described performance metric is commonly visualized as a plot of precision over recall. The plot is built by sorting the positive detections by their decreasing classification score; for each detection, the performance of the detector is visualized by a point on the curve which is defined by the precision and recall values of the accumulated TP and FP counts prior to the current detection; see figure 3.10 for a simplified illustration. Optimal overall performance corresponds to a precision/recall curve integrating close to 1 (target area 3 in figure 3.10); in practice, the curve reflects the tradeoff between a strict parameterization of the detector designed to achieve high detection precision (target area 2 in figure 3.10), thereby generating fewer detections and sacrificing recall, and a more lenient setup which maximizes the number of detections in favor of a high recall (target area 1 in figure 3.10) and at the cost of a lower precision. From the precision/recall curve, the simplified criteria of *average precision (AP)*, *area under curve* and *P/R break-even point*, the point of equal precision and recall, can be derived as single values to characterize the overall detector performance.

The use of a pose estimation approach for the detection task allows to identify potentially failed detections and to discard them based on a *geometric consistency criterion* as follows: given a pose hypothesis provided by the voting step as outlined in section 3.3.2 and given the training pose discretizations as defined in table 3.3, we can determine if the pose obtained from the estimation process (section 3.3.2.4) is within the range of discretization of the initial pose hypothesis. If it is beyond the hypothesis discretization range, we have one of the following cases:

- strong symmetries of the object appearance between different hypotheses result in similarly scoring pose estimates for significantly different poses,
- the initial pose assumption was incorrect, for example due to erroneous feature matches, a wrong initial hypothesis or unknown object geometry,
- the pose estimation failed despite a correct hypothesis.

In all three cases, we can discard the entire detection: in case of symmetries, the corresponding symmetric poses will also generate similarly high scoring pose hypotheses which will be evaluated in addition to the current hypothesis; in the other two cases, our approach does not provide reliable information on the presence of an object instance. This consistency check can be considered

a *hard breakdown* behavior of the detector as opposed to the *soft breakdown* of pure 2D detection approaches for which a clear distinction between obviously correct and obviously incorrect detections usually has to be made by choosing a threshold on the continuous detection scores. The precision/recall plots in the following sections clearly illustrate the difference, see for example figure 3.14. While a soft breakdown behavior may be advantageous when aiming at high detection recall on benchmark data sets, many industrial applications, in particular in safety-critical environments, require a clear separation based on a hard, semantically meaningful criterion; in the present work, this separation does not consist in an arbitrary cutoff threshold, but it intuitively results from the success or failure of the pose estimation process. Since our work aims at showing the added benefit of a 3D model-based approach, we chose to apply this strict consistency criterion to all subsequent evaluations. Note that this does not prevent from fine-tuning the approach in order to optimize detection precision based on the continuous pose estimation scores of the remaining consistent detections.

3.4.2 2D Localization

In the following, we present the evaluation results of our detector on two datasets.

3.4.2.1 PASCAL 2006

We evaluate the 2D localization performance on the PASCAL 2006 test set [18]. The evaluation follows the conventions of the PASCAL 2006 object detection challenge.

Section 3.3.2.4 compared three pose estimation approaches on synthetic datasets; the robustly sampled iterative and the perspective three-point approach both showed suitable performance. Figures 3.11, 3.12 compare the two approaches on real data, the PASCAL VOC2006 car (figure 3.11) and motorbike (3.12) test datasets. Note that for better visualization of the difference between the two methods, the plots are scaled to only show $prec \in [0.7 \cdots 1], rec \in [0 \cdots 0.6]$.

Although the average performance of both approaches is comparable, the three-point estimation performs better in terms of precision. The advantage of the iterative approach in the synthetic tests in section 3.3.2.4 originated mainly from its use of the pose hypothesis as an initialization of the estimation procedure and its more precise estimation of the elevation angle. On real data, however, hypotheses often deviate more significantly due to erroneous feature matches in the voting step. In these cases, the three-point algorithm has a better chance of succeeding since it relies only on the features of the pose hypothesis and does not use the pose associated with the hypothesis as an initialization. The iterative method, which uses the initial pose, gets stuck in less advantageous local minima. Moreover, precise estimation of the elevation does not have a significant impact on the 2D overlap criterion, since the backprojected 2D bounding boxes for different elevations usually have similar outlines (at least for the typical viewpoints contained in the VOC 2006 dataset). Since only the 2D overlap is evaluated in these precision/recall plots, this advantage of the iterative method cannot be measured.

In terms of recall, both approaches depend on the same feature matches and consequently perform comparably. The small advantage of the iterative approach seems to be due to its convergence

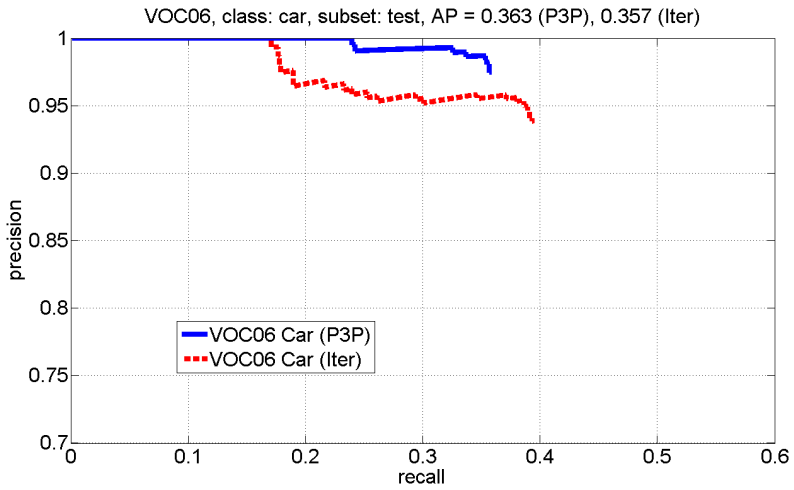

Figure 3.11: Comparison of two pose estimation methods (see section 3.3.2.4) on the PASCAL 06 car dataset; for better visualization of the difference, the plot is scaled to only show $prec \in [0.7 \cdots 1], rec \in [0 \cdots 0.6]$. Although the robustly sampled iterative approach performed better on synthetic data (see section 3.9), the results of the perspective three-point approach on real datasets are better, since it depends less on the pose initialization.

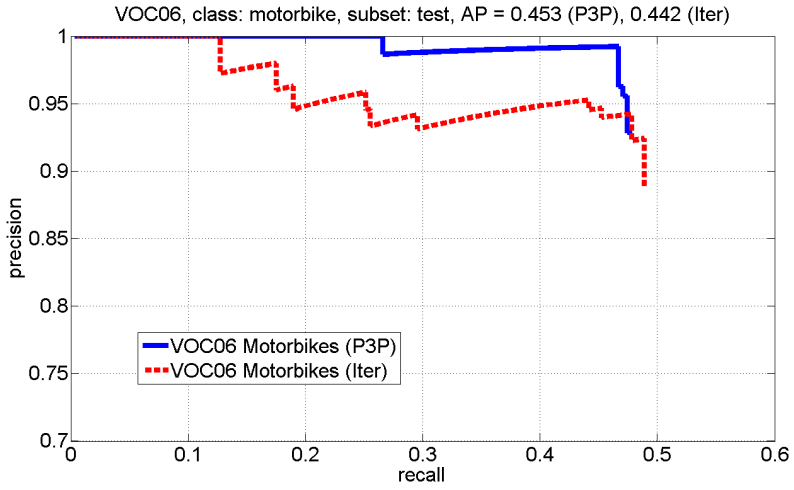

Figure 3.12: Comparison of two pose estimation methods (see section 3.3.2.4) on the PASCAL 06 motorbike datasets; for better visualization of the difference, the plot is scaled to only show $prec \in [0.7 \cdots 1], rec \in [0 \cdots 0.6]$; see above.

behavior in difficult cases which are dominated by a large number of bad feature matches; in these cases, it will converge to a pose close to the initialization which may not correspond to the pose of the actual object in the image, but in some cases may still result in a 2D bounding box fulfilling the overlap criterion. In contrast, the three-point estimation fails with improbable poses which deviate significantly from the hypothesis and can be more easily identified and discarded, which results in a higher precision, but reduces recall. Since the objective of the present work is to emphasize

the advantage of a 3D pose estimation for improving precision, the above results suggest that the remaining experiments in this chapter rely only on the three-point estimation approach.

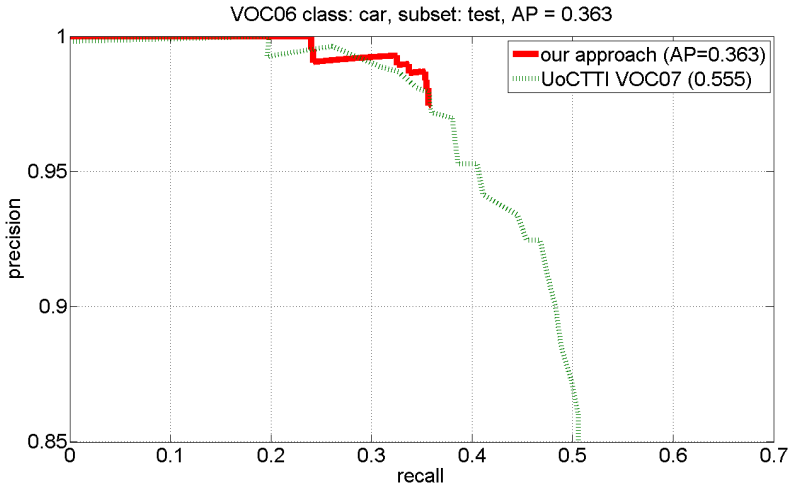

Figure 3.13: Precision/Recall for the PASCAL 2006 car dataset of our approach and and the best PASCAL Challenge 2007 detection on the 2006 test set; for better visualization of the difference, the plot is scaled to only show $prec \in [0.85 \cdots 1], rec \in [0 \cdots 0.7]$.

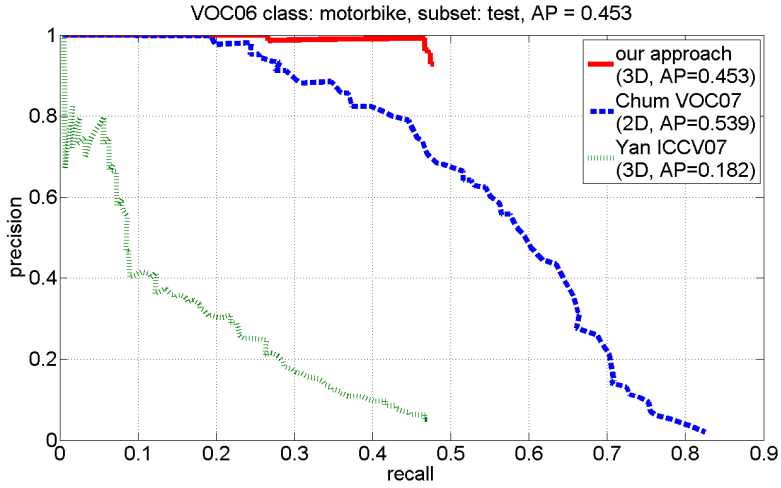

Figure 3.14: Precision/Recall for the PASCAL 2006 motorbike dataset of our approach, the 3D approach of [134] and the best PASCAL Challenge 2007 detection on the 2006 test set.

Figures 3.13, 3.14 show precision/recall curves for the PASCAL VOC2006 car (3.13) and motorbike (3.14) test datasets; we evaluated on the entire test sets. Our approach produces few false positives and achieves an excellent detection precision as long as sufficient feature matches for an accurate pose estimation can be found. No fallback 2D detectors are used when the minimum required number of 4 geometrically consistent matches necessary for a 3D pose estimation can no longer be found; consequently, recall is lower than for pure 2D detectors. In figure 3.14, we provide the results of the 2D detector of [8] which performed best on the 2006 motorbike dataset in the PASCAL Challenge 2007. Their method achieves a higher average precision due to a better

recall behavior; however, on the motorbike test set, our method detects fewer false positives due to the more restrictive geometry verification and consequently maintains a detection precision above the results of [8] for a significant fraction of the test set. On the motorbike dataset, our approach performs better than on the car dataset (figure 3.13). Car objects usually have less textured structure which reduces the number of pose-discriminant features found during training and matching. Still, our approach achieves similar precision when compared to the 2D detector of [22] which performed best on the 2006 motorbike dataset in the PASCAL Challenge 2007. Since the work of [134] is similar to our approach in that they resort to an existing 3D model geometry, we also show their results on the VOC2006 motorbike test set in figure 3.14. Although they use real training images and directly focus on 2D localizations, their P/R curve is lower. This might be due to the much smaller number of images used in their training procedure.

Figure 3.15: Some successful 2D detections from the PASCAL 2006 car test set. This figure is best viewed in color.

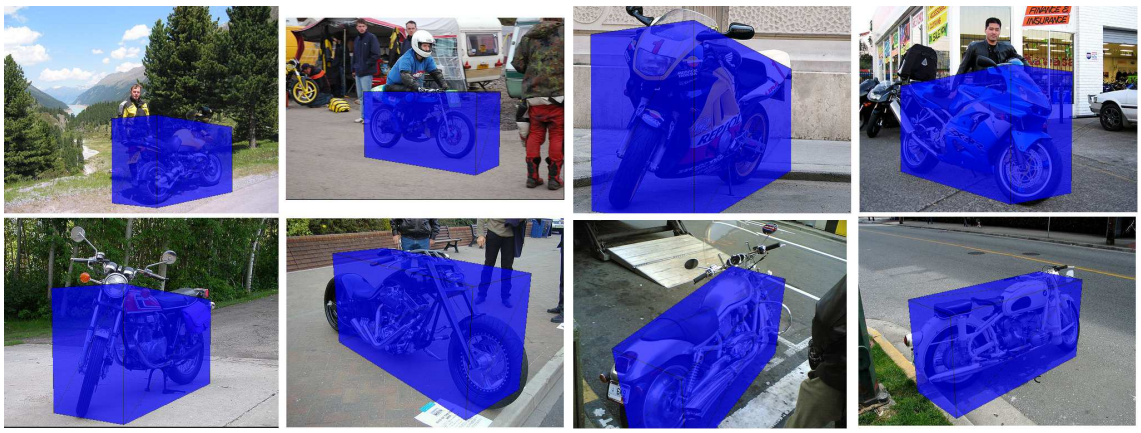

Figure 3.16: Some successful 2D detections from the PASCAL 2006 motorbike test set. This figure is best viewed in color.

Figures 3.15, 3.16 show some examples of successful detections on the PASCAL 2006 test set. The 2D detections and the estimated 3D poses are visualized using semitransparent 3D bounding

Figure 3.17: Remaining issues illustrated on a few examples from the PASCAL 06 motorbike dataset (from left to right): nonrigid geometry, incorrect feature matches, unknown camera transformation, unknown object geometry. This figure is best viewed in color.

boxes, backprojected onto the image plane. Figure 3.17 depicts some failed detections. The failed detections are due to incorrectly established feature correspondences, failed 3D pose estimations due to ambiguities or geometrically inconsistent feature layouts due to nonrigid deformations, object geometries which have not been trained for, camera transformations which cannot be represented with the chosen simplified camera model or an insufficient number of features in case of small objects.

It should be noted that detection performance vitally depends on descriptor similarity. Since our training and testing descriptors stem from different data types (synthetic resp. real images), the overall descriptor similarity is reduced, resulting in fewer correct matches. As a consequence, the discriminative filtering step outlined in section 3.3.1.4 is crucial in achieving a sufficiently accurate matching performance. If the discriminative filtering step during training was omitted, training would result in a codebook which has little discriminativity with respect to object/background separation, pose resolution and 3D position stability. On average, more than 90% of the features are discarded during filtering as being not sufficiently stable and discriminant. We found that the average precision on the same dataset falls from 0.453 to 0.125 for motorbikes and from 0.363 to 0.1 for cars when not using discriminative filtering.

3.4.2.2 3D Object Category dataset

The 3D Object Category dataset CAR [102] contains images of 10 different object instances from 42 different viewpoints with their 2D annotated bounding boxes. We follow the evaluation protocol described in [102], using 3 randomly selected object instances for testing. Unlike [102] who chose to omit the farthest distance, this approach is trained and evaluated on all viewpoints in the database. The training set outlined in section 3.4.1 is kept unchanged; in order to account for the different background characteristics, we re-train the background from images of the 7 object instances not used for testing, where the positive object annotations in each image are masked out. Detection results are given in terms of the PASCAL overlap criterion as described above.

Figure 3.18 provides the precision/recall plots on the 3D Object Category dataset CAR [102] in comparison to the approach in chapter 4. Although the present approach uses synthetic training data and no 2D fallback in case of a failing 3D pose estimation, the precision is comparable. As for the PASCAL 2006 dataset in section 3.4.2.1, recall is reduced since none of the object instances contained in the 3D Object Category dataset has an exact equivalent in the synthetic training set.

See section 4.3.2 in chapter 4 for a comparison with related work. Some detection results are given in figure 3.19; for better visualization, a synthetic car model representing the object class is rendered in the estimated pose alongside each detection. On the largest of the three test distances, no detections have been found; this illustrates one of the main shortcomings of relying on a full pose estimation which requires a minimum number of feature correspondences, since these are frequently too sparse on the lower object scales. Figure 3.20 shows examples of failed estimations due to underestimating the camera distance (top left) and incorrect elevation estimates; elevation estimation is unstable since poses with similar azimuth and different elevations have similar scores.

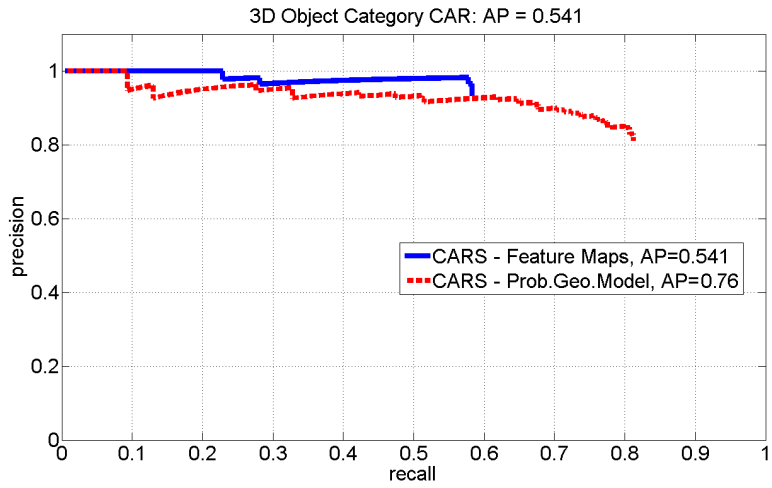

Figure 3.18: Precision/Recall curves on the 3D Object Category dataset CAR [102]; the current approach in comparison to the method proposed in chapter 4.

3.4.3 3D Pose Estimation

In addition to the 2D localization, the proposed approach yields an estimate of the generic object's 3D pose. The precision of the 3D pose estimation was evaluated in the following two experiments.

3.4.3.1 Estimation of Object Orientation

In addition to the 2D annotations used in section 3.4.2.2 to assess the 2D detection performance, the 3D Object Category dataset [102] also provides approximate orientation annotations for each image. Since each object instance is shown in the same discretely sampled viewpoints, the dataset allows to benchmark the pose estimation performance of the present approach. In chapter 4, we also evaluate orientation estimations on the bicycle class. However, the discriminative filtering step of the present approach did not yield enough stable features on the delicate bicycle frames, since the local features computed on the rendered training images contained a significant amount of background which the filtering step could not match over a sequence of pose variations. This issue will be addressed in chapter 4.

Figure 3.19: Some successful detections on the 3D Object Category dataset CAR [102]; for better illustration, a synthetic car model is rendered alongside the detection in the estimated pose.

Figure 3.20: Some failed detections on the 3D Object Category dataset CAR [102] due to underestimation of the distance or wrong elevation estimates.

The orientation estimates on the 3D Object Category dataset CAR are given in figure 3.21 as a confusion matrix between groundtruth annotations (rows) and estimations (columns) for discretized azimuth annotations in 45° steps. It is important to note that the orientation estimates are computed as averages of the successful detections only; consequently, the results of the current approach are slightly biased since the approach from chapter 4 has a higher recall and the estimations are thus averaged over a larger result set. The advantage of the full pose estimation as part of the current approach results in more precise orientation estimates. The main source of errors is the incorrect interpretation of rear views as being frontal views; interestingly, the error is not symmetric. Apparently, the training of the feature map and the clustering of the descriptors have resulted in an imbalanced representation which favours frontal over rear hypotheses in the voting step. Whenever the estimation process is initialized with the correct hypothesis, however, the resulting pose estimation is able to achieve a high accuracy. This becomes particularly obvious in contrast to the estimation performance of the probabilistic model from chapter 4. As mentioned in section 3.4.2.2, no detections have been found on those test images which contain an object instance on the largest of the three discrete distances. Since smaller object instances result in fewer feature detections with lower detail resolution, these cases are more difficult for pose estimations and would have likely resulted in more estimation errors. Their absence in the orientation evaluation thus represents a further bias in favor of the present approach.

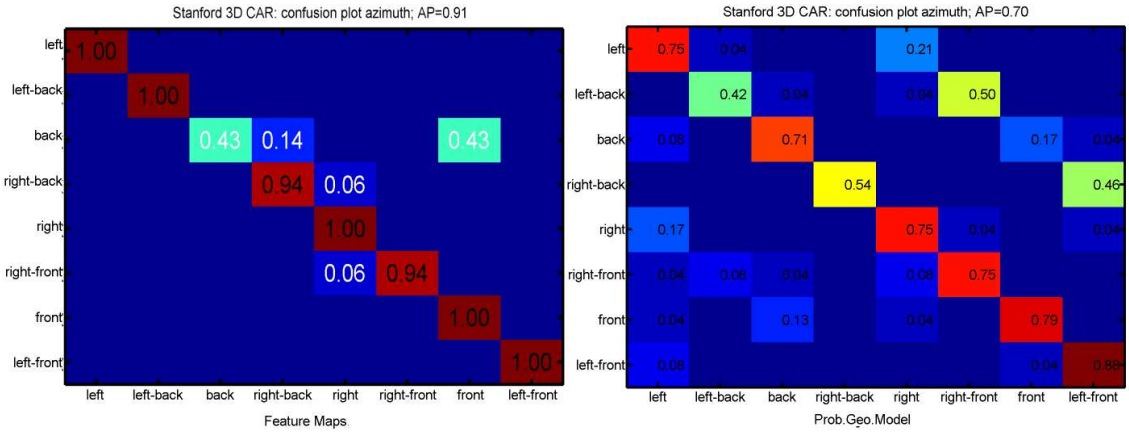

Figure 3.21: Confusion matrix (rows: groundtruth, columns: estimates) for orientation estimates on the 3D Object Category dataset CAR [102]; the current approach (left) in comparison to the method proposed in chapter 4 (right).

3.4.3.2 Pose Estimation on 3D Groundtruth

In this section, we aim at evaluating the precision of the 3D pose estimated by the detection process. We took a set of images of two toy cars with a calibrated camera; see figure 3.22 for an example. For each image, the actual 3D world coordinates of the toy cars ($x^{gt,real,3D}$) w.r.t. the calibration pattern as well as the intrinsic (K^{gt}) and extrinsic ($V^{gt,real}$) camera matrices are known. In particular, the intrinsic camera parameters focal view, principal point and pixel dimensions are computed and any distortion in the test images is removed; we use the calibration approach

of [135]. The synthetic camera which governs the rendering process of the training data is now modified to match the intrinsic parameters of the real camera recovered during calibration, the training process described in section 3.3.1 is performed again on the same training data as in section 3.3.1.1 and the detection is applied to the test images, resulting in estimated virtual 2D bounding box positions $x^{est,virt,2D}$.

In order to compare virtual and real poses in a common 3D coordinate system, it has to be taken into account that the estimated pose (extrinsic camera matrix $P^{est,virt}$) is relative to the 3D bounding box determined from the synthetic training objects in the training world coordinate system. The task of mapping the synthetic pose into the coordinate system of the real camera for metric comparison of errors in centroid position and box orientation can be described as another pose estimation. We assume that the detected synthetic bounding box can be interpreted as the projection of a known metric bounding box; given the known intrinsic matrix of the real camera K^{gt}, the metric dimensions of the bounding box of each toy car ($x^{gt,real,3D}$), and the 2D coordinates of the detected synthetic bounding box after projection into the image ($x^{est,virt,2D}$) with known correspondences between the real and the synthetic bounding box corners, we wish to compute the transformation ($P^{est,real}$) such that

$$\bar{P}^{est,real} = \underset{P^{est,real}}{\operatorname{argmin}} \| x^{est,virt,2D} - K^{gt}\ P^{est,real}\ x^{gt,real,3D} \|^2. \tag{3.14}$$

The problem has an exact solution only when synthetic $x^{est,virt,3D}$ and real $x^{gt,real,3D}$ bounding boxes have the same dimensions; otherwise, the solution that minimizes the projection error in image space is retained. The iterative estimation (section 3.3.2.4) is used; in this setting, a random sampling is not necessary since the correspondences between synthetic and real bounding box corners are known. The iterative estimation is better suited since no closed-form estimation can be performed when synthetic and real bounding boxes do not have the exact same dimensions and only a sparse set of bounding box corners instead of the dense maps (such as in figure 3.6) is available. Based on the pose of the 3D bounding box mapped from the synthetic to a metric coordinate system, metric position and orientation errors can be derived.

Table 3.4 lists the errors of the 3D estimations over 14 calibrated poses of toy cars. The position error is measured as the Euclidian distance between the centroids of the ground truth and the estimated bounding boxes, while the orientation error is measured as the angle between their dominant axes. Although the precision of our pose estimation cannot compete with methods for registration or tracking of a specific model (cf. chapter 5), it is sufficient as an initialization for these methods. The position error is mainly due to the underestimation of the distance between object and camera. In section 5.4.4, we show that the approximate pose estimations obtained by the detection can serve as initializations for more precise registration methods.

3.5 Conclusion

In this chapter, we have presented an approach to viewpoint-independent object class detection. The main contributions lie in the training process of local 3D-aware features from synthetic 3D

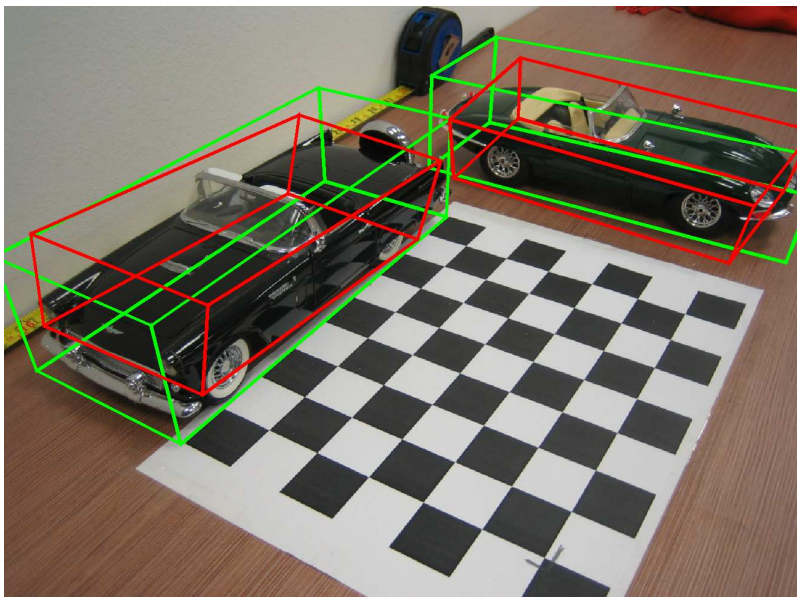

Figure 3.22: Calibrated scene used for 3D evaluation. Measured ground truth bounding boxes are displayed in green, estimated bounding boxes in red. (Errors pos./orient.: left car $21.9\,mm$/$6.55°$, right car $60.3\,mm$/$6.14°$).

Table 3.4: Evaluation of the 3D pose estimation for a calibrated scene; 14 ground truth experiments with two toy cars (cf. toy car length $280\,mm$).

	mean	std. dev.
position error of bounding box centroid	$33.9\,mm$	$21.74\,mm$
angular error of main bounding box axis	$10.7°$	$5.2°$

models, the selection of pose- and class-discriminant descriptors and the extension of the traditional probabilistic voting scheme to 3D (i.e., beyond 2D interest regions). The method generates an approximate 3D pose hypothesis for generic object classes which is then refined by a full 2D-3D pose estimation. However, the low recall of the method in conjunction with the relatively slow pose estimation suggest a comparison to a more flexible, lightweight approach. In the next chapter 4, we outline such an approach, explicitely addressing the shortcomings of the present method, and compare the performance of the two methods.

4

Detection with a Probabilistic Geometric Model

In this chapter, we describe an approach to object class detection which is based on combining a part-based appearance representation learnt discriminatively from real training images and a 3D geometry representation learnt generatively from synthetic CAD models. In section 4.1, we relate this approach to the method from chapter 3, we describe the training and detection steps in section 4.2 and provide the results of the experimental evaluations in section 4.3.

4.1 Introduction

In the previous chapter, an initial approach to 3D model-based object class detection was introduced. While the approach performs well on a number of test databases, it suffers from some limitations which will be addressed in this chapter.

- The unsupervised clustering of local features into a 3D map tends to saturate for large numbers of training objects. There is no explicit grouping of features according to their positions on the 3D model geometry: those local features which are found on object parts that reoccur on the object due to symmetries will fall into the same clusters due to similar appearance, but they will also be scattered in multiple geometric positions on the 3D feature map. While these symmetries can be compensated up to a certain level in the pose verification step in section 3.3.2.4, a leaner representation would be advantageous which groups features according to their appearance, their geometric and potentially their semantic role. In this chapter, we address this issue by first, splitting the appearance and the geometry training processes, second, relying on a generative geometry model which does not saturate even for larger training sets, and third, resorting to a part model to take into account both appearance semantics and geometric layout.
- The performance of the approach described in chapter 3 depends significantly on the quality of the CAD model textures in the training set and their similarity to the appearance expected in the test data. In order to alleviate this dependency, it might be advantageous to allow for incorporating real training images into the training process in order to cover more exotic

object appearances which may not be found in the available CAD model database. In this chapter, the proposed separation of appearance and geometry training will allow to use real training images for appearance and CAD models for geometry, thereby circumventing the potential gap between synthetic textures and real object appearance.

- The previous approach performs a pose estimation based on matching geometrically consistent groups of local features in order to determine the approximate 3D pose of an object in a test image. The pose estimation result then allows to determine the 2D location of the object by backprojecting the 3D estimation into the image plane. This 3D estimation step is inevitable for the previous method. In certain cases, such as for small object scales, the requirements for a full 3D pose estimation, notably a sufficient amount of local features found, might not be fulfilled. Consequently, it would be preferable to have an initial 2D detection step which already incorporates certain geometric constraints, and a subsequent 3D pose estimation based on the detection of larger 2D image regions corresponding to object parts. In this chapter, we describe an approach which meets these requirements.

- The pose estimation techniques used in the previous chapter rely on potentially non-deterministic matching by random sampling. Although their robustness has been shown [23], faster, deterministic pose estimation methods exist. In addition, the pose estimation approaches by matching as outlined in chapter 3 will fail when no subset of at least three matching model and image features in a trained geometric constellation can be found. Of the evaluated pose estimation approaches in section 3.3.2.4, the three-point estimation does not tolerate any deviation from trained geometric constellations and the iterative estimation has only limited robustness towards layout perturbation. In this chapter, we present a probabilistic pose estimation which does not depend on random sampling. It is faster to evaluate and allows to deal with significant variation in geometric constellation of object parts.

To summarize this chapter, based on discriminative part-based 2D detectors which are both robust and straightforward to train, only a few synthetic 3D models for each object class are used to learn a generative 3D representation of the object class geometry without relying on the presence of synthetic textures. In particular, no manual annotation of individual part locations is necessary. Moreover, a probabilistic pose estimation allows to obtain an approximate 3D object pose alongside a precise 2D detection which is robust towards small perturbations in the geometric layout. This estimation step provides an effective evaluation measure to assess the consistency of the 2D part detections with respect to the full 3D geometry of the object class. We thereby show that a joint model for geometry and appearance can be avoided by learning separate models for both and combining them at a later stage. As a result, one can use better adapted, leaner representations and separate training sources and exploit the ubiquitous availability of geometrically faithful synthetic 3D CAD models for object class detection.

The chapter is structured as follows. In section 4.2.1, an overview of the training approach is given. Details on the appearance model for part-based detection on 2D training images are presented in section 4.2.2. The geometric representation of the object classes, which is built from synthetic 3D models, is described in section 4.2.3. Section 4.2.4 describes the combined detection process. Experimental results and a comparision with the state of the art are given in section 4.3 for the 3D Object Category datasets CAR and BICYCLE [102] as well as for the 2006 and 2007 PASCAL

datasets for motorbike and car models [17, 18]. For a survey of the related work on object class detection, see section 3.2.

4.2 Our Approach

In the following, we describe our approach, consisting of two separate models for appearance (section 4.2.2) and 3D geometry (section 4.2.3), and outline how they contribute to the detection process in section 4.2.4.

4.2.1 Overview

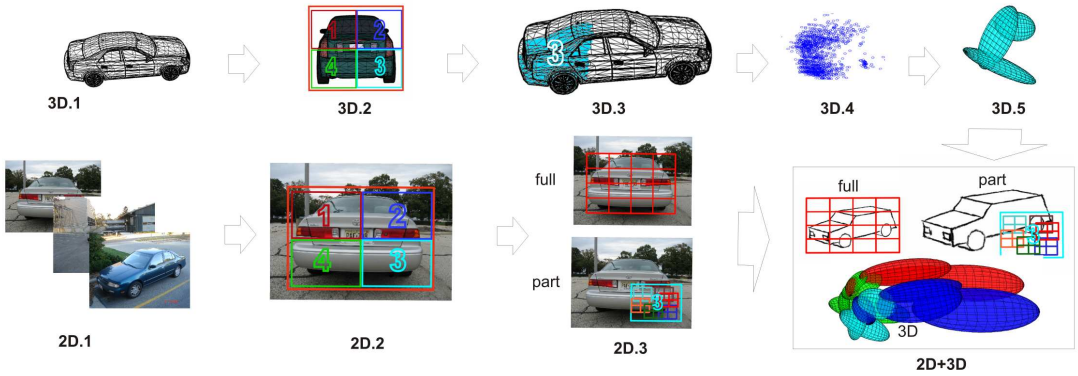

Figure 4.1: Overview of the two training steps. (Top) Mixture models are learnt from synthetic 3D models to describe the class geometry. (Bottom) Full object and part appearance are learnt from a 2D image database. See text for details. The figure is best viewed in color.

Figure 4.1 illustrates the approach presented in this chapter, which is based on combining a 2D appearance model with an external 3D geometry.

The object class appearance is learnt from a database of 2D images, showing the objects from different viewpoints (figure 4.1, 2D.1); each image is annotated with the 2D bounding box and the viewpoint of the object, but neither manual part annotations nor segmentations are necessary. Each annotated bounding box is subdivided (2D.2) into a regular grid where each grid block represents a part of the object. A single spatial pyramid detector is used for the full object regions of interest (2D.3 top), while for each part region under each viewpoint, several smaller, overlapping spatial pyramid detectors are trained (2D.3 bottom).

The 3D geometry is learnt from one or several synthetic 3D CAD models representative of the object class geometry. The models are rendered from many viewpoints (figure 4.1, 3D.1); the rendered images are subdivided (3D.2) into the same regular grid as in (2D.2). For each rendered pixel inside a part region, its original position on the CAD model surface is known (see section 2.2.4); thus the image pixels belonging to the same part can be backprojected onto the surface (3D.3), sampled into discrete 3D points (3D.4) and the distribution of all 3D points belonging to one object part can be modeled by a mixture of Gaussians (3D.5).

The resulting object class representation now consists of a 2D pre-detector of regions of interest, dense 2D part detectors per viewpoint, and an approximate representation of the 3D geometry of the object class (figure 4.1, 2D+3D). To summarize, by subdividing both the annotated real training bounding boxes (figure 4.1, bottom) and the rendered images of the 3D models (figure 4.1, top) into the same regular grid of parts (figure 4.1, 2D.2 and 3D.2), the link between local 3D geometry and local 2D image appearance is established. Note that this requires bounding box annotations as well as approximate viewpoint annotations in the 2D training images.

4.2.2 Part-based Appearance Model

Learning the appearance of an object class needs to take into account large intra-class and viewpoint variations in addition to significant background clutter and partial occlusions. Moreover, when dealing with part-based object class detection, one aims at learning sufficiently powerful part descriptors for relatively small image patches. These patches do not always contain sufficient structure to be suitable for discriminative classifiers. In addition, manually including detailed annotations on the location of each object part is tedious. Consequently, some authors have suggested using fixed part layouts for 2D detection [8, 29, 92] where each detector is associated with an object part depending on its location inside the grid. More recently, the use of hierarchical structures as a representation for both the entire object and its subparts has been advocated [10, 22]. This work builds on these ideas in relying on spatial pyramids [49] both for the global object and the local parts. It extends beyond previous, sparse part-based approaches [92] by using both densely computed local features and spatial pyramids densely covering the image space. Learning the appearance of an object class consists of a two-fold supervised training process which is both efficient and robust; figure 4.1, 2D.3 illustrates the two detection components which are described in the following paragraphs.

4.2.2.1 Detector Layout

Both detection steps build on densely computed local features as their basic building blocks. The DAISY descriptor [118] was chosen because of its efficient implementation. Initially, from all positive and negative training images, DAISY descriptors are randomly sampled and clustered into a small codebook of fixed size C using a standard k-means algorithm with random initialization. The codebook size can be adapted to the complexity of the object class; see section 4.3 for details on the chosen parameter. Given each positive training annotation, DAISY features are then computed densely within the annotated training region and assigned to their respectively closest codewords to build localized occurrence histograms.

A single detector is trained on entire objects to identify regions which have a higher likelihood of containing an entire object instance; figure 4.1, 2D.3 top, shows an example layout. The dimensions and aspect ratio of the training annotations determine the dimension of the detector used during testing.

For the part-based detectors, instead of manually selecting semantic parts, the training annotations are further subdivided into a regular grid of $V \times W$ regions, assuming that the densely sampled detectors whose centers fall into the same region can be considered to share some common

characteristics of this part of the entire object under this viewpoint; see figure 4.1, 2D.3 bottom. Consequently, $V \times W$ groups of detectors are obtained, each representing the appearance of a part of the entire object under the current view. These part detectors have to deal less with background variation, but focus primarily on differentiating between the appearance of areas of an object under changing viewpoints.

The negative training examples for object parts and full objects are initially chosen randomly on the background of the training images; the detector layouts which were used for the positive training instances are re-used to determine the layout of the negative samples.

4.2.2.2 Appearance Representation

Following [49], localized occurrence histograms are combined into spatial pyramids. DAISY features are computed densely on the image (figure 4.2, left) and assigned to their closest codebook entries. On each pyramid level, local occurrence histograms are built (figure 4.2, center) and successively merged into the next higher level (figure 4.2, right). The full spatial pyramid descriptor contains a concatenation of all subhistograms of the different levels (figure 4.2, bottom). For the full object pre-detector, a single spatial pyramid is built to represent the appearance of the entire object under the current view; see figure 4.1, 2D.3 top. For the part-based detectors, $V \times W$ spatial pyramids are obtained, each representing the appearance of a part of the entire object under the current view. The spatial pyramids of each part are densely sampled and allowed to overlap in order to completely cover the part area as shown in figure 4.1, 2D.3 bottom.

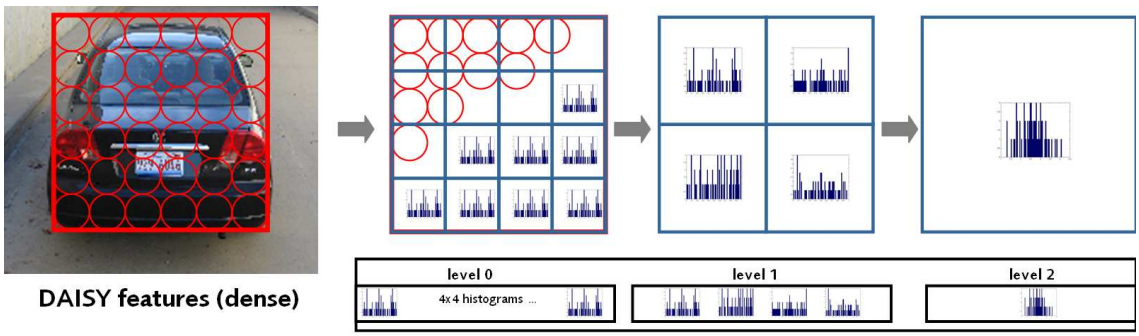

Figure 4.2: Appearance is coded in a spatial pyramid of a codebook of DAISY descriptors. DAISY features are computed densely on the image (left) and assigned to their closest codebook entries. On each pyramid level, local occurrence histograms are built (center) and successively merged into the higher levels (right). The full spatial pyramid descriptor contains a concatenation of all subhistograms of the different levels (bottom).

Given positive and negative training examples, separate SVM classifiers are now trained, one for the entire object under all viewpoints and one for each of the $V \times W$ object parts under each of the weakly-annotated views as provided by the training database. See appendix C for an outline of SVM classifiers. In the case of the 3D Object Category datasets CAR and BICYCLE, annotations are given for discrete distances and elevation and azimuth angles (also see section 4.3). As

illustrated in figure 4.3, the proposed approach parameterizes the viewpoints in spherical coordinates of $v = (r , a , b)$ where r is radius, a azimuth and b elevation, assuming a simplified camera which is always oriented at the centroid of the object. A part is assigned to one block of the fixed grid when the center of the support of its representing spatial pyramid is inside the block, thereby allowing for some overlap between the parts (see figure 4.1, 2D.3 bottom). All views having azimuth angles within a given range together with all distances and elevation angles associated with this azimuth angle are combined to train $V \times W$ part detectors for this particular base viewpoint; see section 4.3 for details on the chosen parameters. To compensate for the random choice of initial negative training instances, a standard bootstrapping procedure is used to iteratively select the most difficult false positives and false negatives for each part classifier. The SVMs are learned on a minimum intersection distance kernel with the per-level weighting scheme suggested in [49]; the distance $k_{A,B}$ between two spatial pyramids A and B, each consisting of L levels on subhistograms with dimension d, is given as

$$k_{A,B} = \sum_{l=0}^{L-1} \left(\frac{1}{2^{L-l}} \sum_{n=0}^{d} (min(A(l,n), B(l,n))) \right). \tag{4.1}$$

4.2.2.3 Training Set Selection by Bootstrapping

On the datasets on which we evaluate our approach, the number of potential negative samples outnumbers the positive ones significantly; moreover, by computing spatial pyramids densely on the entire image, the training of the SVM on all negative training samples becomes computationally intractable. Consequently, we have to derive a method of selecting a representative subset of negative training samples.

The properties of an SVM and its generalization capacity are mainly defined by the training samples closely located to the separation plane; they are the *difficult* training samples x in that their distance to the separating plane is smaller than some threshold, $\| y(x)\| < \epsilon$; typically $\epsilon \in [0 \cdots 0.5]$. It is advisable to include a large number of these difficult samples in the training process in order to achieve a suitable class separation. The difficulty of a training sample, however, can only be assessed in terms of an existing separation plane, i.e. a pre-trained SVM. We propose the following approach which can be interpreted as a variant of *bootstrapping* [7].

A positive sample in the context of the present work is defined as a spatial pyramid overlapping by more than 50% with the region of its respective part annotation; a negative sample is any spatial pyramid overlapping by less than 50%. For a justification of the chosen overlaps from the evaluation criteria, see section 3.4.1.

1. Create an initial training set $M_0 \subset M$ from the set of all possible samples M with their annotations T by randomly adding N positive (x_p with $T_i = 1$) and N negative samples (x_n with $T_i = -1$). Each subsequent bootstrap iteration is denoted by its index t, initially $t = 0$.

2. Train an SVM classifier c_{M_t} on the training set M_t.

3. Evaluate the SVM classfier c_{M_t} on the entire set M and assign each sample s its signed distance to the separating plane d_s.

4. Select all samples s_i with $T_i = -1 \ \wedge \ d_{s_i} > \epsilon$ (false positives); sort them by decreasing positive d_s and keep at most $N/2^t$ of the *worst*, i.e. highest positively scoring, samples. Assume that fp_t false positives are kept in a set FP_t.

5. Select all samples s_i with $T_i = 1 \ \wedge \ d_{s_i} < \epsilon$ (false negatives); sort them by increasing negative d_s and add at most $fn_t = min(N/2^t, fp_t)$ of the *worst*, i.e. lowest negatively scoring, samples. In order to avoid overfitting to positive samples, in each step t we do not add more false negatives than false positives to the training set. Assume that fn_t false positives are kept in a set FN_t.

6. Assemble a new training set $M_{t+1} = M_t \cup FP_t \cup FN_t$.

7. Repeat from step 2 in the next iteration $t + 1$, until $FP_t = 0$ either because $N/2^t = 0$, i.e. the maximum number of boostrap iterations has been reached, or because no false positives have been found in the last step. In that case, c_{M_t} is the final classifier.

In the present work, $N = 2500$ and $\epsilon = 0.5$. The parameter C controlling the tightness of the SVM classification margin is determined by gridsearch with $C \in [0.001 \cdots 100]$ with 5-fold cross-validation of the SVM on the entire training set; see appendix C for details on the parameterization of SVMs.

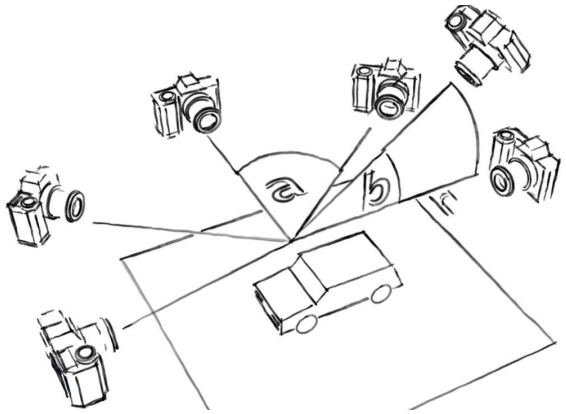

Figure 4.3: Discretization of the viewpoints for initial classification into "base viewpoints" in discrete azimuth steps, each combining multiple elevations and distances.

4.2.3 Geometry Model

The following section outlines how the model of the object class geometry is built to represent the 3D distribution of the centers for each of the $V \times W$ parts per object class and for each discretized camera viewpoint (figure 4.3).

Recently, some publications have proposed methods for building a 3D representation from training data to be used for detection tasks. In most cases, groups of consistently deforming image regions are promoted to a higher geometry model to reflect their co-occurrence [2, 102, 114].

In this section, a different approach is proposed which relies on commercially available synthetic 3D models; see figure 4.4 for some examples. However, unlike previous approaches, the geometric learning task is separated from the appearance component described in the previous section 4.2.2. No explicit matching between synthetic textures and real images is required; still the precise geometry of synthetic models can be used in an extremely flexible way to learn the 3D distribution of parts of an object class, as long as the models represent characteristic object class geometries. In particular, no manual annotation of part locations is required for geometry training.

4.2.3.1 3D Training Data

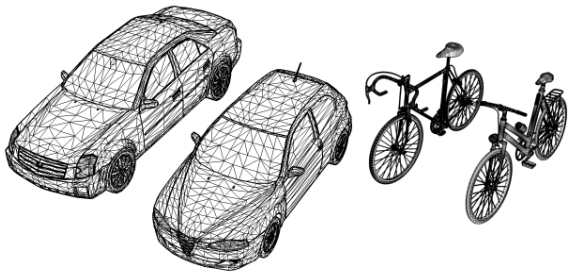

Figure 4.4: Synthetic 3D models used for the geometry training.

The use of synthetic models as training sources for the geometry allows to densely sample the space of possible viewpoints and to choose the models such that the training database includes representative object surface geometries. The approach follows the pose space parameterization of [102] as defined by their test database; the parameterization is based on a spherical coordinate system as illustrated in figure 4.3; a simplified camera model is assumed where rotations around the camera view axis are not part of the parameter space and the camera is always directed at the model centroid.

When working with synthetic models, the appearance of synthetic textures is difficult to relate to real images. By limiting the contribution of the synthetic models to their geometry, this problem is efficiently avoided; furthermore, far fewer models are needed to only represent the geometry of an object class rather than all possible textural appearance variations. Still, if some texturing is available, it would be possible to train a 2D detector similar to section 4.2.2 on the synthetic parts and, by applying it on the synthetic images, to identify symmetry relations of parts on certain surface regions; one could then account for these symmetries in the subsequent model building steps.

For each object class, all its 3D models are rendered into images of fixed dimensions, along with their automatically generated bounding boxes; see chapter 2 for details on the synthetic rendering process. Each model is rendered from the same viewpoints that are present in the real image

database (termed "base viewpoints") as well as from additional densely sampled viewpoints, reflecting intermediate distances and object orientations. By using synthetic models, viewpoints can be more densely sampled from the space of all relevant poses to account for the typical visibility of the parts under perspective projection, depending on 3D surface structure and local self-occlusions. For each rendered view, the bounding box is subdivided into a regular grid of $V \times W$ parts (see figure 4.1, 3D.2) in the same way as for the appearance training (see figure 4.1, 2D.2). The assignment of 3D surface locations to parts does not require any annotation, since for synthetically rendered images, the actual 2D bounding boxes are known; their automatic subdivision into the same regular grid that was used for the real training images directly establishes the link between appearance parts and 3D geometry.

4.2.3.2 Mixture Models

After perspective projection of a synthetic model, for each image pixel its 3D position on the original 3D model surface is known; as a consequence, the 3D points belonging to each part under each of the specified viewpoints can be determined and projected into a common object coordinate system as shown in figure 4.1 (3D.3). More specifically, the rendering pipeline of modern graphics processors as described in section 2.2.4, figure 2.3, allows to annotate each rendered pixel with more than just its color values; we use this property to pass on the original 3D coordinates of each point on the model surface to the final image pixel. Figure 4.5, left and center, displays the 3D point clouds for one car base viewpoint and four parts; different colors indicate different parts. This representation now allows to associate regions of the 3D object surfaces under perspective projection to the corresponding appearance parts, since for the rendered as well as the real images the same regular grid of $V \times W$ parts was used. This link holds true independently of the real image training data on which the appearance has been trained, as long as the same regular grid has been used to determine the part regions in synthetically rendered images and annotated real training images.

Once all available models of one object class have been processed under all discretely sampled viewpoints, Gaussian mixtures are fitted to the point clouds of each part per base viewpoint, using the standard Expectation-Maximization procedure [13]. The choice of Gaussian mixtures reflects a trade-off between a faithful representation of the 3D geometry and a conveniently parameterized formulation which later allows to efficiently evaluate the probability of co-occurrence of parts, given the geometry model. This trade-off is reflected in the number of mixtures to represent each parts' geometry: more mixtures will allow to better represent the geometry, while at the same time increasing computational cost during pose estimation. In this approach, the number of mixtures per part is iteratively chosen according to the MDL criterion [94]. Assuming that each part obeys a multivariate multimodal Gaussian distribution with parameterset $\theta_{k\in\{1\ldots K\}} = (\mu_k, R_k, w_k)$ in 3D (where μ_k is the centroid of each mixture component, R_k its covariance and w_k the weight of the mixture component), for each part distribution L the likelihood

$$p(L|K,\theta) = \prod_{n=1}^{N} \sum_{k=1}^{K} p(x_{(n,3D)}|\theta_k) \tag{4.2}$$

of each of the N 3D points' $x_{(n,3D)} \in L$ belonging to mixture k is maximized, while accounting for the MDL penalty term. Figure 4.5, right, shows the fitted mixture models for the 3D point distribution of the parts of a car from base viewpoint "rear". For each part under each base viewpoint, such a representation of its originating 3D surface positions is built.

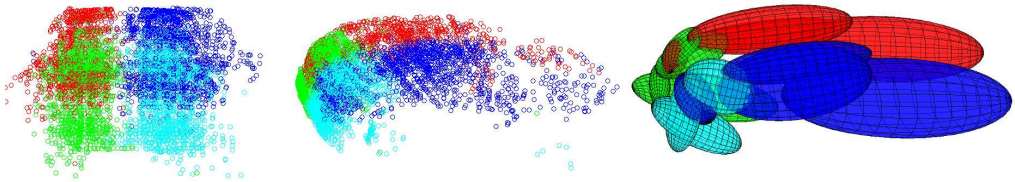

Figure 4.5: 3D point distributions and fitted mixtures for four parts of the car class from base viewpoint "rear" (left: projection from actual viewpoint, center: rotated, right: estimated mixtures).

4.2.4 Detection

This section outlines the detection steps, starting with the initial 2D predetection of the entire object, the detection of pose-specific 2D parts for the most probable base viewpoint, and the maximum-likelihood optimization process to estimate the remaining pose parameters.

4.2.4.1 2D Detection

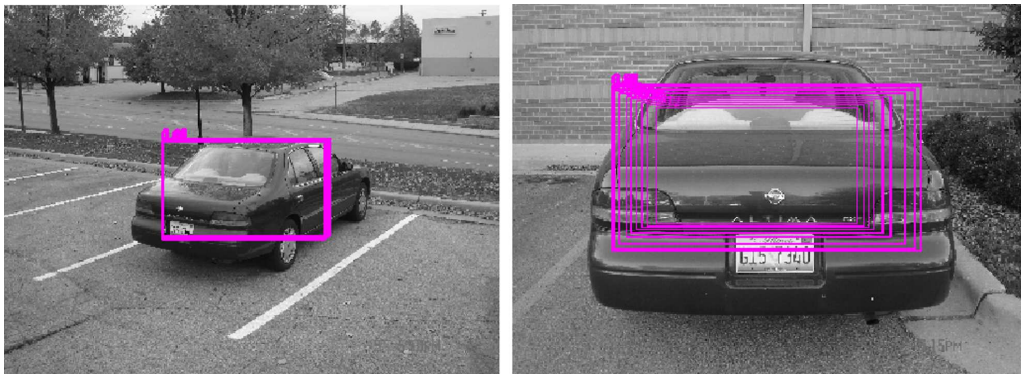

Figure 4.6: Initial detections with a full object spatial pyramid classifer; note the frequent underestimation of object scale due to the lack of object pose information.

4.2.4.1.1 Pre-Detection

The 2D detection process starts with an initial pre-detection to identify regions of interest potentially containing fully or partially visible objects. This method follows the work of [12] in using a sliding-window detection and a subsequent mean-shift mode estimation to merge and localize these regions of interest in image and scale space. This detection step alone is usually unable

to generate a reliable localization, since it does not deal with occlusion and is sensitive to the detector window dimensions chosen. In addition, no sliding-window approach can sample all possible window layouts on all possible scales; consequently, additional verification steps are necessary. Figure 4.6 shows some example detections for the full object pre-detector; note that this pre-detector frequently underestimates the actual scale of the object which is due to its lacking information on the full object pose. While this issue could potentially be solved by other means, for example by the use of segmentation masks as in [63], we suggest to deal with this issue by including more comprehensive information on the object pose into the detection process. In the following steps, knowledge on the full 3D geometry of the object class allows to accurately choose the entire image region containing the object, thereby significantly improving this initial detection and providing an evaluation score which measures the consistency of the detected parts with the learnt geometry model. Multiple object instances visible in the same image are handled implicitly by applying the subsequent steps to each of the pre-detection results in the image; every object instance that is pre-detected in this step can potentially yield a separate final detection, provided that enough consistent parts are found to allow for a successful pose estimation.

4.2.4.1.2 Pose-Specific Parts Detection

The part detection forms the fundament which the 3D pose estimation will rely on. Section 4.2.2 described how classifiers for different regions of an object under each base viewpoint are computed. Typically trained on much smaller image parts, the discriminativity of these part detectors is reduced; however, by computing them only on the previously identified regions of interest in the test images, much of the background variability is removed which allows to focus the training process on differentiating between base viewpoints and parts on the objects. In addition, these parts can be densely computed on every pixel within the region of interest. The large number of resulting detections increases robustness of the following pose estimation step. A simple voting procedure is used to determine the most likely azimuth-only base viewpoint $\upsilon_i = (1, a_i, 0)$, given all the N detected parts at 2D image locations x which were classified as belonging to base viewpoint hypothesis h and part label $l \in 1...n_L$ with discriminative detection probability $p(x, h, l)$:

$$p(\upsilon_i) = \sum_{n=1}^{N} \sum_{m=1}^{n_L} p(x_n, h = \upsilon_i, l_m). \tag{4.3}$$

Note that this voting does not yet take the distribution of parts into consideration; it only selects the most promising base viewpoints to evaluate in the subsequent pose estimation. In particular, for a given base viewpoint, we may have several different part detections at the same image location; these ambiguities have to be resolved in the pose estimation step. Some part detection results are visualized in figures 4.22, 4.23 along with the most likely base viewpoint votes illustrated as histograms.

4.2.4.2 3D Pose Estimation

For all detected 2D parts at 2D image locations x with part labels l of a specific base viewpoint hypothesis υ_i, an iterative pose estimation now provides an evaluation of the probability of occurrence of the refined viewpoint in simplified spherical camera parameters $\upsilon = (r , a , b)$

as illustrated in figure 4.3. We are now searching for those camera parameters that maximize the probability of occurrence of the detected constellation of object parts in the image when the 3D model is projected into the image. This is equivalent to finding those camera parameters υ within the constraints Υ_i given in section 4.3 which maximize the likelihood of the detected 2D parts after perspective projection Φ_υ of the K 3D Gaussian mixture components of this base viewpoint into the image space:

$$\begin{aligned} &\underset{\upsilon \in \Upsilon_i}{\operatorname{argmax}} \quad \prod_{n=1}^{N} p(x_n, \upsilon_i, l|\upsilon) = \\ &\underset{\upsilon \in \Upsilon_i}{\operatorname{argmax}} \quad \prod_{n=1}^{N} \sum_{k=1}^{K} w_k \, \mathcal{N}\left(x_n, \upsilon_i, l|\Phi_\upsilon(\mu_{(k,3D)}), \Phi_\upsilon(R_{(k,3D)})\right), \end{aligned} \tag{4.4}$$

where $\mathcal{N}\left(x_n, \upsilon_i, l|\Phi_\upsilon(\mu_{(k,3D)}), \Phi_\upsilon(R_{(k,3D)})\right)$ denotes a 2D Gaussian mixture component evaluated at the 2D image location x_n of the detection with viewpoint hypothesis υ_i and specific part label l, where the 2D Gaussian is the result of projecting the 3D Gaussian with mean $\mu_{(k,3D)}$ and covariance $R_{(k,3D)}$ into a 2D image, using the perspective projection Φ_υ. To simplify the computation of the likelihood under the perspective projection Φ_υ of the per-part covariances $R_{(k,3D)}$ into image space, Φ_υ is approximated by the Taylor expansion localized at the mixture centroids $\mu_{(k,3D)}$, assuming the projection to be locally affine:

$$\Phi_\upsilon(x_{(3D)}) \approx \Phi_\upsilon(\mu_{(k,3D)}) + J_{\Phi_\upsilon}(x_{(3D)} - \mu_{(k,3D)}) \tag{4.5}$$

which allows to compute the approximate covariance of the projected 3D mixtures $\Phi_\upsilon(R_{(k,3D)})$ from the original covariances $R_{(k,3D)}$ using the Jacobian J_{Φ_υ} of the projection Φ_υ evaluated at the 3D centroids $\mu_{(k,3D)}$:

$$\Phi_\upsilon(R_{(k,3D)}) \approx J_{\Phi_\upsilon}(\mu_{(k,3D)}) \cdot R_{(k,3D)} \cdot J^t_{\Phi_\upsilon}(\mu_{(k,3D)}). \tag{4.6}$$

The optimization problem in equation 4.4 is solved under the constraints Υ_i given in section 4.3 which reflect the discretization used during training. Solving requires choosing unambiguous part label assignments l for each detection at a location x with pose hypothesis υ_i. Two options exist for assigning l to one of the n_L part labels:

- For each detection at a location x, choosing part label

$$\hat{l} = \underset{l \in \{1 \ldots n_L\}}{\operatorname{argmax}} \left(p(x, \upsilon_i, l)\right) \tag{4.7}$$

  based only on the scores of the discriminative classifiers. The choice is made once initially and the chosen part labels are kept constant over the optimization.

- As part of an EM-like iterative optimization, where the update step is given in equation 4.4 and the part label selection is done in the E-step as

$$\hat{l} = \underset{l \in \{1 \ldots n_L\}}{\operatorname{argmax}} \left(p(x, \upsilon_i, l|\tilde{\upsilon})\right) \tag{4.8}$$

where $\tilde{v}$ is the result of equation 4.4 from the previous iteration. For initialization, part labels are chosen as the maximum of the classifier scores as in the first option.

In both cases, optimization of equation 4.4 is done using a genetic algorithm [46] which is analyzed in detail in section 5.4.3. The genetic approach performed best in the experiments due to its robustness towards local optima. In practice, we chose to minimize the negative log-likelihood for better numerical stability. Since the discriminative classifiers tend to assign small scores to part detections on ambiguous image regions, in our experiments, the fixed part choice and the dynamic part relabeling showed identical performance; a different behavior might be expected when a different type of classifier is used.

The resulting detection now allows to evaluate the probability of occurence of an object of the searched-for class under a consistent 3D pose. Moreover, the 3D bounding box backprojected into the image can be used to determine the smallest circumscribed 2D rectangle which significantly improves the 2D scale estimates of the initial detection step. Note, however, that the 3D pose estimation obtained is relative to the virtual camera parameters used to generate the geometry training data from the synthetic model database. Without information on the real camera used to take the specific test image, the computed virtual 3D pose does not relate to the actual metric 3D pose of the object, but provides only orientations and relative distances. Still, if metric calibration data of the camera used to take each test image was available, the virtual camera pose could be promoted to an actual 3D measurement. The pose estimation scores are not used to rank detections over different images of a database, since the log-likelihood is not normalized. Instead, the improved 2D bounding box after pose estimation and 2D backprojection is re-classified with the full object classifier (see section 4.2.4.1.1) to generate a comparable overall score for ranking.

4.2.5 Implementation

An important building block of the approach lies in the appearance representation as spatial pyramids of occurrence histograms of DAISY features. The process of computing the pyramids densely at each pixel position in the image is time consuming. Since on their most detailed level, the pyramids share occurrence histogram blocks of local image patches (see figure 4.2), both the computation of the histogram blocks as well as the combination of the blocks into spatial pyramids can be parallelized since the blocks and the pyramids are independent of each other. The parallelized computation performed on the graphics processor; in the following paragraphs, we outline its implementation.

4.2.5.1 General Purpose Computation on the GPU

During the last decade, speeds of single CPU cores have stopped increasing; at the same time, the graphics processor (GPU) has changed from a hardware component for dedicated graphics output into a fully programmable processor architecture. Driven by early work on general purpose computing on GPUs [82] for visualization and mathematical co-processing, programming models such as $OpenCL^{©}$ and $CUDA^{©}$ have become established tools for solving computationally expensive operations, notably in the domain of computer vision [27]. By subdividing tasks into small,

locally independent computation kernels with a limited amount of shared memory and a common, albeit slower access to global memory, operations can be deployed in a massively parallel way on several hundred subprocessors with reduced instruction sets (Single Instruction Multiple Data, SIMD). The $CUDA^{©}$ programming model which we rely on for the following implementation allows the kernels to be written in a C language; one computation kernel is executed by one or more threads depending on their organization into blocks of a regular grid; threads are grouped into *warps* which run on one processing unit (PU) and share its local memory. Although certain limitations exist as to the amount of parallelization and memory access, the approach offers a high degree of scalability.

4.2.5.2 Parallelized Operations

Given a codebook which has been built during training, containing Nb descriptor centroids (see section 4.2.2.1), and a test image on which we wish to compute spatial pyramids, we initially compute dense DAISY features on the entire test image, resulting in Na descriptors and their associated positions in the image.

4.2.5.2.1 Nearest-Neighbour Codebook Assignment

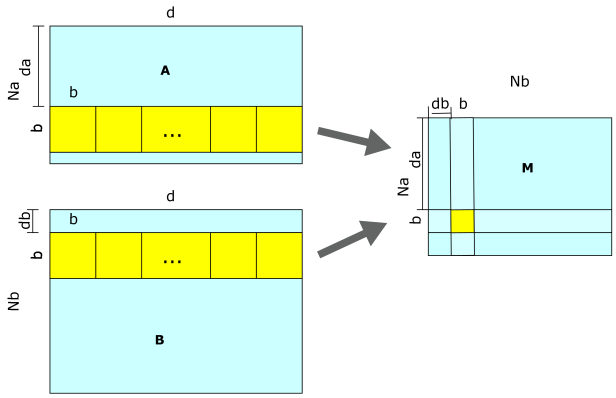

Figure 4.7: Parallelization scheme for the descriptor distance matrix, containing the pairwise distances between all Na descriptors found in a test image and the Nb descriptor centroids of a codebook built during training. The matrix is divided into regular blocks which are computed in parallel to account for the limited shared memory of the GPU.

In order to build occurrence histograms, we initially need to assign each of the Na descriptors found in the test image to its nearest neighbour among the Nb descriptors in the codebook in terms of a descriptor distance; we use the L2 distance. First, the distances between each descriptor in the test image and all the codebook descriptors have to be computed in a matrix M of dimension

$Na \times Nb$. Figure 4.7 illustrates the parallelization scheme used for this task. Given the two sets A and B consisting each of Na and Nb DAISY descriptors with per-descriptor dimension of $d = 200$, we wish to compute the matrix M of dimension $Na \times Nb$ such that each entry $m_{i,j}$ in M corresponds to the sum of the squared distances between each of the d elements of two descriptors a_i and b_j. To adapt the problem to the limited shared memory on the GPU, the entries of matrix M are computed in regular blocks of dimension $b \times b$ such that two of these blocks fit into the shared memory of one processing unit (PU) and each element of the block is computed by one thread. The elements of one block starting at position (da, db) in M depend on the b descriptor rows starting at da in A and db in B. We subdivide each group of b descriptor rows into the same set of regular blocks, step through each two corresponding blocks in A and B and load their entries into shared memory. Each thread then accumulates the result of an operation on the elements in two different rows of the two blocks into one entry in M.

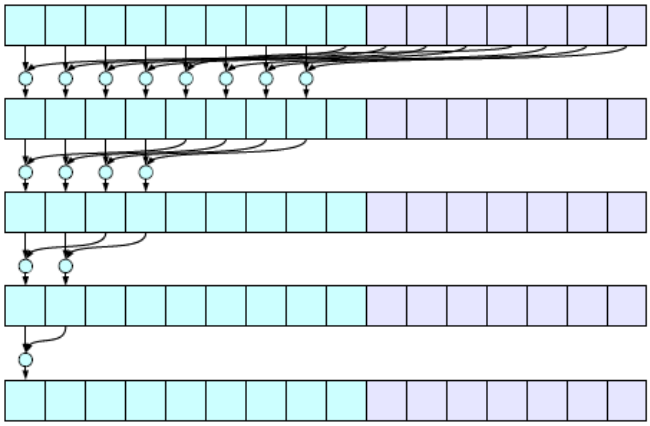

Figure 4.8: Parallelization scheme (parallel *reduce*) for the computation of the nearest neighbour in the codebook for each of the descriptors in the test image. In each iteration, two values from the lower and the higher part of each row of the distance matrix are compared in parallel and the smaller value of the two is kept for the subsequent iteration.

Based on the matrix M, one row containing the distances of one test descriptor to all codebook entries, we wish to find the codebook entry having the smallest distance to the test descriptor. The parallelization scheme suited for this task is a standard *reduce* operation which runs on each row of the matrix M to recover the minimum value and its column index; figure 4.8 illustrates the concept. On each of the Na rows of M, initially $Nb/2$ threads compute the minimum of two entries and save the minimum value and its column index in shared memory. In each subsequent iteration, half of the threads repeat the operation on the remaining values until a single minimum and column index per row remains. The column index per row is the codebook entry which is most similar to the test descriptor; this index is stored in a new matrix at the 2D position on which the corresponding test descriptor has been computed in the image.

4.2.5.2.2 Local Histograms

The previous steps resulted in a matrix which contains at each pixel position of the test image the index of the codebook entry which is closest to the DAISY descriptor found at that position. In the next step, we subdivide the entire matrix into a regular spatial grid; for each block in the grid, we build a histogram counting the frequency of occurrence of the codebook entries in that block. In order to account for optimized operations on the GPU, we only use power-of-two histogram bin counts and power-of-two spatial block dimensions; to densely cover the entire index matrix, the full grid is offset by multiple values up to the dimensions of one grid block and a new set of histograms relative to this offset is built. The occurrence histogram of one block in the grid is computed by all threads available on one PU; current CUDA hardware and API limit the maximum number of threads on one PU to 512. Since the codebook dimensions typically required exceed this value (see section 4.3), one thread runs over the entire block area and accumulates the occurrence count of a range of codebook indices into shared memory; the shared memory blocks are then concatenated to form the initial level 0 of the spatial pyramids.

4.2.5.2.3 Spatial Pyramids

In order to assemble spatial pyramids building on the blocks of local histograms computed in the previous steps, on each of the L levels, 2^2 local histograms per dimension from the previous level have to be combined into a single histogram which is then concatenated with all previous subhistograms; figure 4.9 shows one such combination step for a single spatial pyramid. A full spatial pyramid with $L >= 0$ levels therefore has dimension $\sum_{l=0}^{L-1}((2^{L-1-l})^2 d)$ where d is the number of bins per histogram, i.e. the codebook size. We assign one PU to the merging of each group of 2^2 local histogram blocks (yellow in figure 4.9) on each level; again, one thread steps through a range of bins of each local histogram and accumulates the bin counts into the corresponding range of bins of a single new histogram which is kept in shared memory. In order to avoid resizing the global memory on each pyramid level, we initially reserve the entire memory space required for the full set of pyramids and save the histograms in an interleaved way.

4.2.5.2.4 Minimum Intersection Kernel

The nonlinear SVM classifier operates on a distance matrix containing a subset of spatial pyramids from training and testing instances, the *kernel*. We have chosen the Minimum Intersection Distance [49] between spatial pyramids. The operation is implemented in the same way as the sum of squared differences matrix for the Nearest-Neighbour assignment described above; it operates on the spatial pyramids as inputs instead of the DAISY descriptors and performs a per-element min instead of a difference.

In comparison to a pure CPU-based C++ implementation without processor-specific optimizations, the full process, as described above, of building spatial pyramids on 3 levels from a codebook of dimension 2048 (resulting in a full pyramid dimension of $21 * 2048$) sampled in steps of 4 pixels on an image of dimension 512×512 takes approx. $80ms$ on an nVidia Tesla C1060 GPU as opposed to approx. $21sec$ on a dual-core CPU at $2.1GHz$.

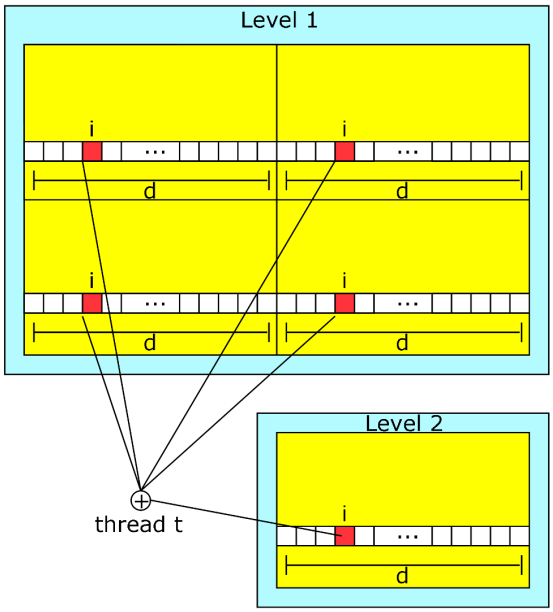

Figure 4.9: Parallelization scheme for the building of spatial pyramids on multiple levels (blue) from local histogram blocks (yellow).

4.3 Experimental Evaluation

On the publicly available 3D Object Category datasets CAR and BICYCLE [102], our approach was evaluated on two tasks, 2D object class detection and approximate 3D pose estimation. Object class detection in 2D was used to assess the contribution of the geometric model with respect to object localization in image and scale space. The accuracy of our approximate pose estimation was evaluated with respect to groundtruth orientation annotations of the 3D Object Category datasets. On the PASCAL VOC 2006 and 2007 motorbike, bicycle and car testsets [17, 18], only the 2D detection was evaluated, since the provided viewpoint groundtruth is too sparsely discretized and not available for all test images.

4.3.1 Dataset

The approach relies on training data from two separate sources. The 3D geometric representation is built from 3D models available from the commercial distributors turbosquid.com and doschdesign.com; see section 2.2.1 for details on our 3D model database. We used two motorbike, two car and two bicycle models (some examples are shown in figure 4.4) which are representative of the object class geometries contained in the test databases. For training, the 3D models are normalized to unit scale in the virtual camera coordinate system and rendered from distances $r \in \{1 \cdots 5\}$ (in multiples of 3D model radius), six elevation angles in steps of 10° and all azimuth angles in 22.5° steps. Consequently, given an initial hypothesis υ_i, the pose estimation step is constraint to $\Upsilon_i = \{r \in \{1 \cdots 5\}, a \in \{a_i \pm 22.5^\circ\}, b \in \{0 \cdots 60^\circ\}\}$ where a_i is the azimuth angle associated with the base viewpoint hypothesis.

Appearance training for the 3D Object Category datasets CAR and BICYCLE [102] relies on the annotated training images of the dataset. For each object class, it contains images of 10 different object instances from 42 different viewpoints. We follow the evaluation protocol described in [102], using the images of 7 randomly selected object instances per class for training and those of 3 unseen instances for testing; the results are averaged over 10 evaluations with different random subset selections. Unlike [102] who chose to omit the farthest distance, this approach is trained and evaluated on all viewpoints in the database. 2D training bounding boxes are computed from the provided groundtruth segmentation masks and the base viewpoint classifiers are trained on the available approximate viewpoint annotations, consisting of three distance units (near, medium, far), three elevation units (low, medium, high) and 8 azimuth steps of approximatively 45°. Codebook sizes of 2048 codewords for the full detector on 3 pyramid levels and 1024 codewords on two pyramid levels for the part detector performed best in our experiments.

On the PASCAL VOC 2006 and 2007 motorbike, bicycle and car testsets [17, 18], no complete viewpoint annotations exist; instead, a subset of the database contains annotations for the discrete object orientations front, rear, left and right. We use these annotations in the training sets whenever available and skip training images without viewpoint annotations. Since our approach focuses on a part-based representation which requires a minimum object size, we skip training images marked as truncated, difficult, or having a maximum axis-aligned bounding box diameter of less than 100 pixels. In order to cover diagonal viewpoints as well, we merge the appearance training data from each VOC database separately with the training data from the 3D Object Category datasets CAR and BICYCLE. Since the 3D Object Category dataset does not contain a motorbike object class, our approach will only able to generate hypotheses for motorbike detections from the four discrete orientations provided by the VOC annotations; the detection of intermediate orientations relies entirely on the local invariance of the part detectors in combination with the pose estimation procedure. See figure 4.25, second from the right, for an example of the impact of this sparse viewpoint annotation, resulting in an incorrect orientation estimation.

4.3.2 Experiments

The 2D localization task is evaluated with the standard 50% VOC Challenge overlap criterion of [18] on the axis-aligned rectangular 2D bounding boxes obtained from backprojecting the 3D bounding boxes generated by the pose estimation; the same method has been used to evaluate the approach in the previous chapter, see section 3.4.1 for details. As in the previous chapter, we resort to a strict geometric consistency criterion in order to remove detections where the pose estimation result is beyond the discretization range of the initial viewpoint hypothesis.

The precision/recall curve obtained with our approach on the CAR dataset [102] is given in figure 4.10 (AP 76.7%). We compare to the best currently reported pure 2D approach [29] (AP 72.6%) and the most recent 3D approach of [114] (AP 55.3%). As can be seen, our detection approach outperforms the state of the art on the CAR dataset. On the BICYCLE dataset [102], our method achieves an AP of 69.8% as shown in figure 4.11 which is slightly below the 2D results reported by [29], probably because on the narrow image regions of bicycle frontal and rear views, the 3D backprojections into 2D image space used by the present approach tend to overestimate relative to the provided groundtruth annotations.

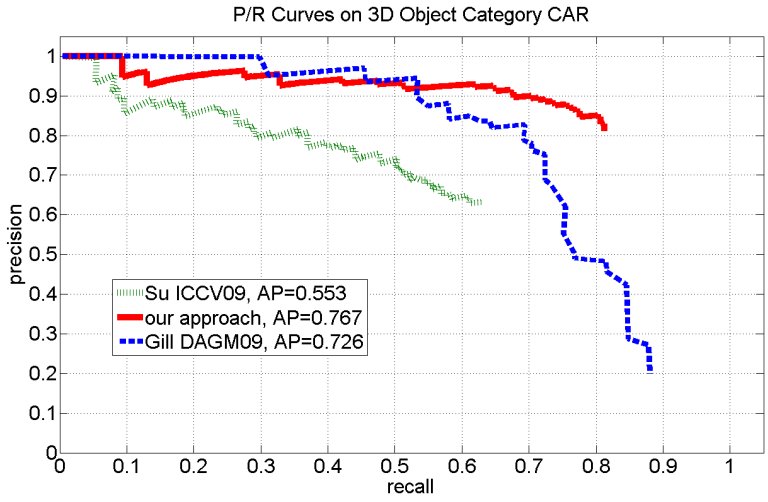

Figure 4.10: Precision/Recall curves on the 3D Object Category dataset CAR [102].

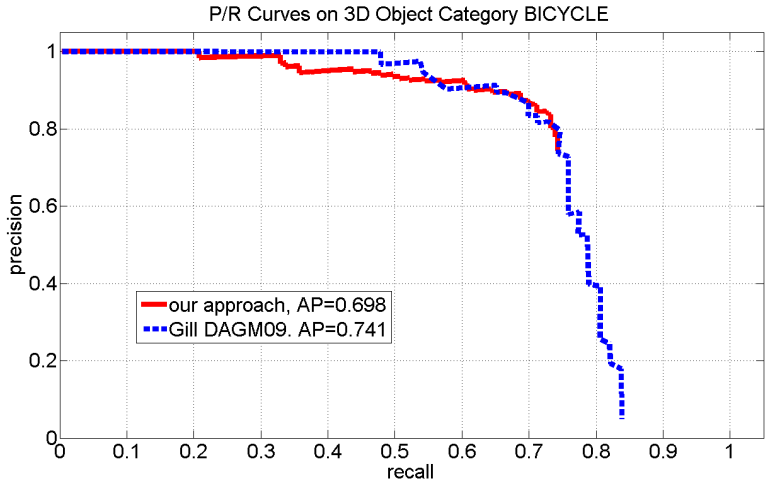

Figure 4.11: Precision/Recall curves on the 3D Object Category dataset BICYCLE [102].

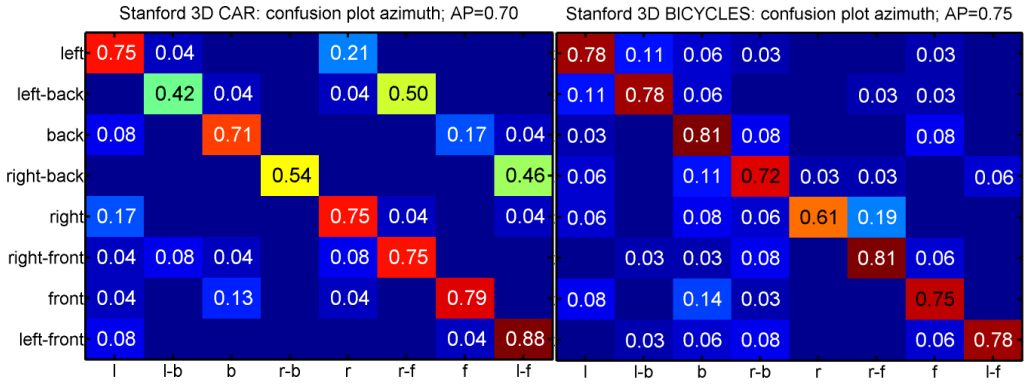

Figure 4.12: Confusion matrices (rows: groundtruth, columns: estimates) for orientation estimates on the 3D Object Category datasets CAR and BICYCLE [102].

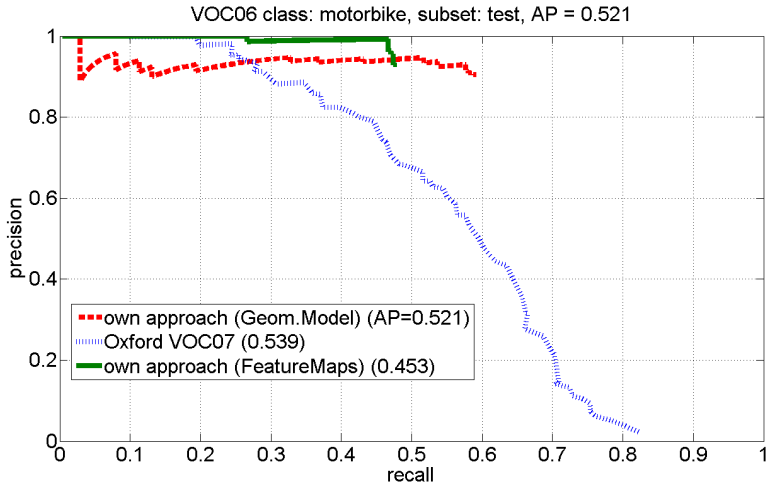

Figure 4.13: Precision/Recall curves on the PASCAL VOC2006 testset MOTORBIKE [18].

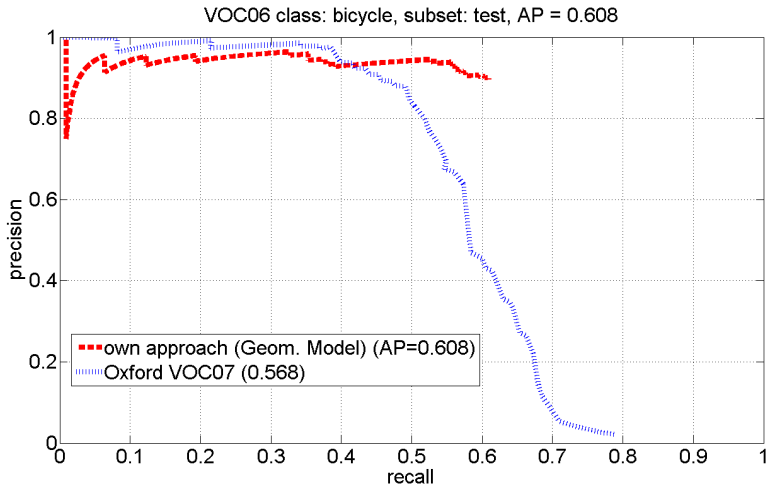

Figure 4.14: Precision/Recall curves on the PASCAL VOC2006 testset BICYCLE [18].

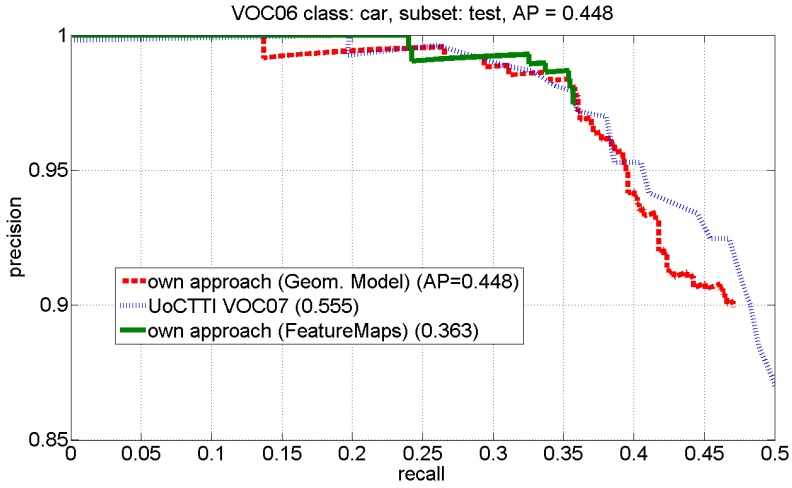

Figure 4.15: Precision/Recall curves on the PASCAL VOC2006 testset CAR [18]; for better visualization of the differences, the plot is scaled to only show $prec \in [0.85 \cdots 1], rec \in [0 \cdots 0.5]$.

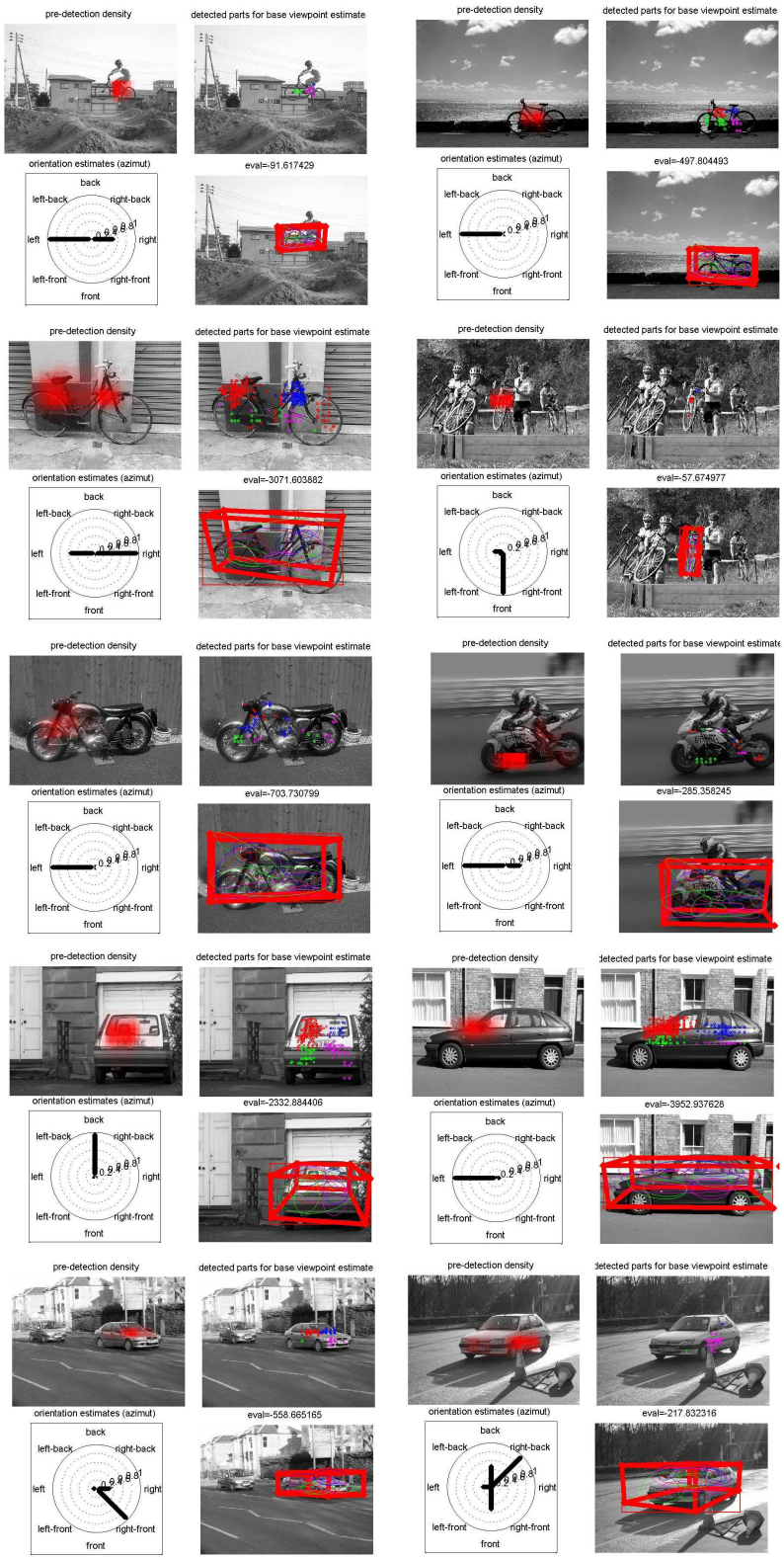

Figure 4.16: Some results illustrating successful detections on the PASCAL VOC2006 testsets CAR, BICYCLE and MOTORBIKE [18]. For each result, the predetection density, the detected parts, the base viewpoint votes and the final pose estimation are visualized.

On the PASCAL VOC 2006 motorbike, bicycle and car testsets [18], we benchmark the results of our approach against the best results achieved on these datasets with 2D detection methods as part of the PASCAL VOC challenge [18], i.e. the work of [8] (referred to as *Oxford* in figures 4.13, 4.14) and [22] (referred to as *UoCTTI* in figure 4.15). For comparison, we provide the precision/recall curves of the approach described in chapter 3 where available (referred to as *FeatureMaps* in figures 4.13, 4.15). Some successful detection results are shown in figure 4.16, some failed detections in figure 4.25. Incorrect detections are mostly due to camera rotations which are not representable by the chosen simplified pose parameterization (figure 4.25, left), nonrigid deformations which result in ambiguous pose votes (figure 4.25, second and third from left) or failed detections on objects with an appearance that differs too much from the training data (figure 4.25, right).

On the CAR dataset (figure 4.15), we achieve a significantly better recall when comparing to the previous approach (chapter 3); this is due to the use of appearance training data from the corresponding training set instead of relying on synthetic textures. However, traditional pure 2D detectors still perform better on small objects which our approach fails to detect; note that the full predetector is trained only on those training instances which are large enough to yield individual training parts for subsequent processing steps. Since the CAR testset contains a particularly high number of small objects, the difference in recall is significant when comparing to the approach of [22]. The MOTORBIKE and BICYCLE testsets contain fewer small objects; here, the main difficulty resides in nonrigid deformations which occur when the front wheel is not aligned with the object body. On the MOTORBIKE testset (figure 4.13), the recall of the previous approach is once again lower, although the more precise pose estimation achieves a slightly better detection precision. As stated above, it has to be noted that this comparison is somewhat biased due to the limitation to only four discrete viewpoints for appearance training on the MOTORBIKE testset. Still, the average precision of 0.521 is comparable to the performance of the best pure 2D detector with 0.539 [8]. This limitation in viewpoints is avoided on the BICYCLE testset, since we rely on a combination of the appearance training sets from VOC 2006 and 3D Object Category dataset as outlined above. As a result, our approach performs better than the 2D method of [8] with an average precision of 0.608 as opposed to 0.568. This result underlines the importance of a pose discretization in at least 45° degree steps to provide a suitable initialization for the subsequent pose estimation.

On the PASCAL VOC 2007 motorbike, bicycle and car testsets [17], we benchmark the results of our approach against the best results achieved on these datasets as part of the PASCAL VOC challenge [17], i.e. the work of [8] (referred to as *Oxford*), and more recently published results [21] where available (referred to as *UoCTTI* in figures 4.18, 4.19). Some successful detection results are shown in figure 4.20, some failed detections in figure 4.26. Incorrect detections are mostly due to camera rotations which are not representable by the chosen simplified pose parameterization (figure 4.26, second from left), geometries that differ too much from the training data (figure 4.26, left and third from left) and bad classification results which lead to ambiguous pose initializations (4.26, right). With the exception of the bicycle class (see figure 4.18), our approach achieves a consistently higher precision for positive detections; as before, the recall is significantly reduced when compared to the currently best pure 2D detectors of [8, 21]. Not surprisingly, the use of a 3D geometry can be useful in improving detection precision and to indicate and devalue detections which are inconsistent with the geometric model; however, this will inevitably result in

more false negative detections. Note that in [21], training data from 2008, 2009 and 2010 challenges is merged to train a detector which is then applied to the 2007 test set; in comparison, our appearance training is limited to the training data from the 2007 challenge and the 3D Object Category dataset which constitutes a much smaller training set. The successful detections in figure 4.20 show that the approach has the ability to interpolate its pose estimations to viewpoints which were not part of the training set, provided that the part classifiers generalize sufficiently well to these unseen viewpoints.

To demonstrate the contribution of our pose estimation component to the 2D detection, we again evaluated the detection task on the 3D Object Category datasets, this time omitting the pose estimation. Instead, the detected 2D parts cast votes for their potential parent objects, similar to an implicit shape model [117]. The AP scores for both classes, bicycles (AP 63.7%) and cars (AP 59.9%), are significantly below those obtained with our combined detection and pose estimation approach (bicycles (AP 69.8%) and cars (AP 76.7%), see above). By including a pose estimation into the detection process, the detection precision can thus be substantially increased; figure 4.21 shows the corresponding precision/recall curves for the CAR and BICYCLE sets. Note how the 2D detections on the car dataset suffer from several highly scoring false positives due to underestimating the bounding box and subsequently missing the overlap criterion; by including the 3D estimation, these underestimations can be removed, thereby increasing the precision. The recall on the CAR dataset remains unchanged, since objects not detected by the complete 2D predetector do not yield part detections which are required for the pose estimation. On the bicycle dataset (figure 4.21, bottom), the improvement is less significant; as stated above, the small object sizes for frontal and rear views and the ensuing low part count with small spatial support render the pose estimation difficult. In some cases, the pose estimation results in overestimated 2D boxes which reduce the recall on this dataset.

In order to benchmark the 3D pose estimation on the 3D Object Category datasets, only the orientation estimations can be compared against the annotated orientations, since the provided elevation and distance groundtruth of the test datasets is too approximate. We bin the continuous orientation estimates in 45° steps to be comparable to the groundtruth annotations. The confusion matrices obtained on the CAR and BICYCLE datasets are shown in figure 4.12; for cars, the diagonal views suffer from multiple symmetries; for bicycles, front and rear views are more difficult to estimate correctly. On the car dataset, the achieved orientation AP of 70% compares favourably to [114] (approx. 67%); no published pose estimation results on the BICYCLE dataset are currently available for comparison.

Figures 4.22, 4.23 show some examples of the full detection process. In each result window, the predetection density is visualized in the top left area, the detected parts which contributed to the best base viewpoint are plotted in the top right area. The votes cast for each base viewpoint bin are visualized as histograms in the lower left area. In the bottom right, the pose estimation along with the backprojected covariance ellipses of the parts is given; note that no additional priors on viewpoints or ground planes are used. Some failed detections are given in figure 4.24; the most frequent reasons for incorrect results are due to an unstable pose estimation on sparse detections (figure 4.24, left), in this case emphasized by the small area covered by the bicycle in rear view. Some detection constellations cannot be differentiated and cause incorrect elevation estimates (center). When the detections are too sparse, the pose voting space is not representative and may generate incorrect initializations (right).

4.4 Conclusion

This chapter has introduced a method for including external 3D geometry from synthetic CAD models into a 2D part-based appearance detection method, yielding an approximate 3D pose estimation and an evaluation score for the 3D geometric consistency of 2D part detections. Although slightly less precise than the method from chapter 3, the approach achieves a higher recall on challenging data sets and benefits from a more flexible training process. Both approaches to object class detection have used synthetic CAD models as their exclusive or partial source of training data to achieve state-of-the-art 2D detection performance in addition to approximate 3D pose estimations. In the following chapter, we will describe a registration scheme which builds on these pose estimations, improves their precision and thereby allows for their use beyond pure detection tasks.

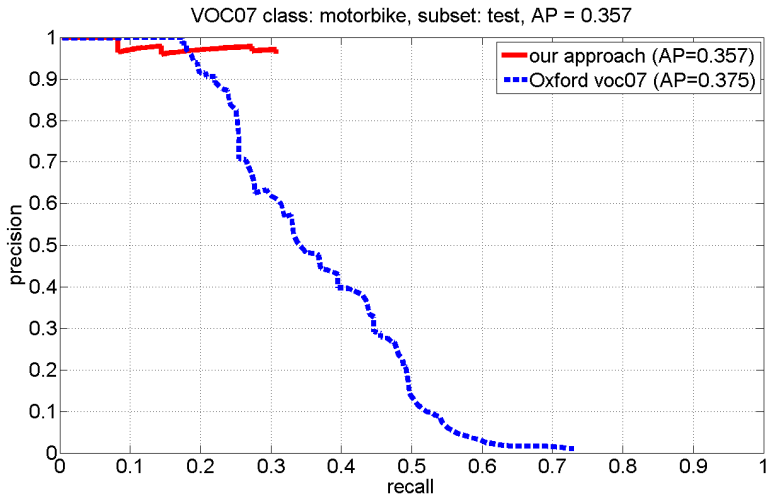

Figure 4.17: Precision/Recall curves on the PASCAL VOC2007 testset MOTORBIKE [17].

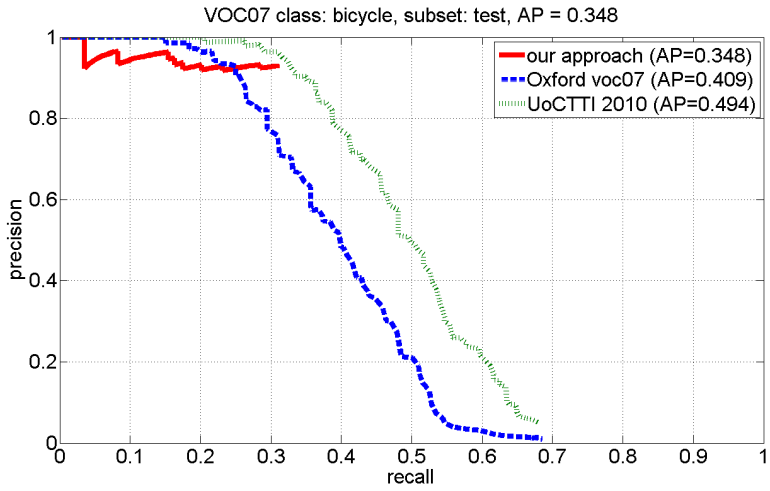

Figure 4.18: Precision/Recall curves on the PASCAL VOC2007 testset BICYCLE [17].

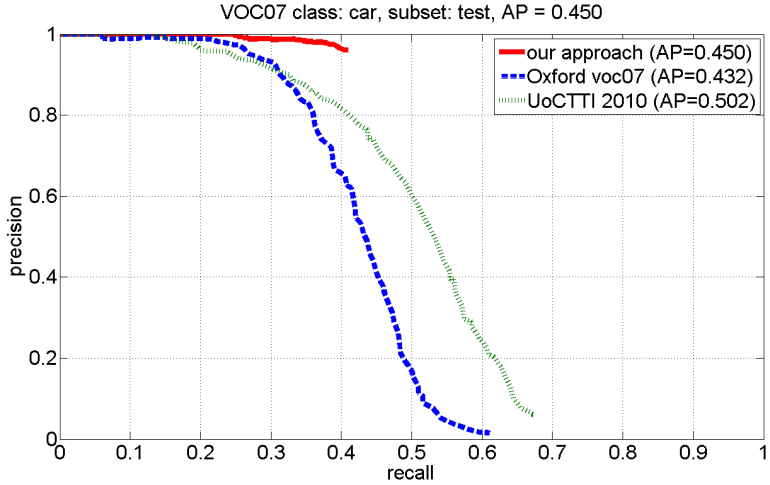

Figure 4.19: Precision/Recall curves on the PASCAL VOC2007 testset CAR [17]; for better visualization of the differences, the plot is scaled to only show $prec \in [0.85 \cdots 1], rec \in [0 \cdots 0.5]$.

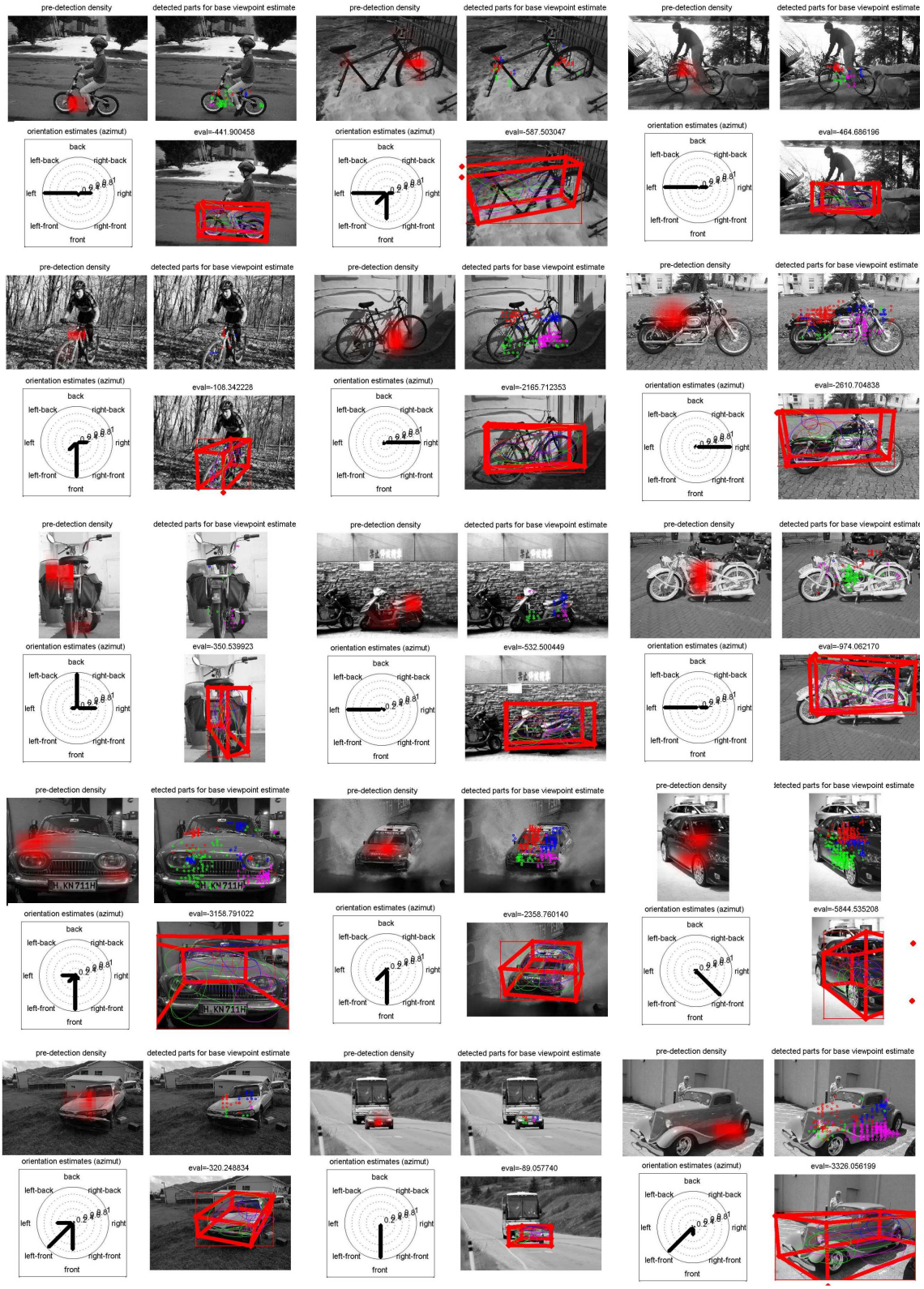

Figure 4.20: Some results illustrating successful detections on the PASCAL VOC2007 testsets CAR, BICYCLE and MOTORBIKE [17]. For each result, the predetection density, the detected parts, the base viewpoint votes and the final pose estimation are visualized.

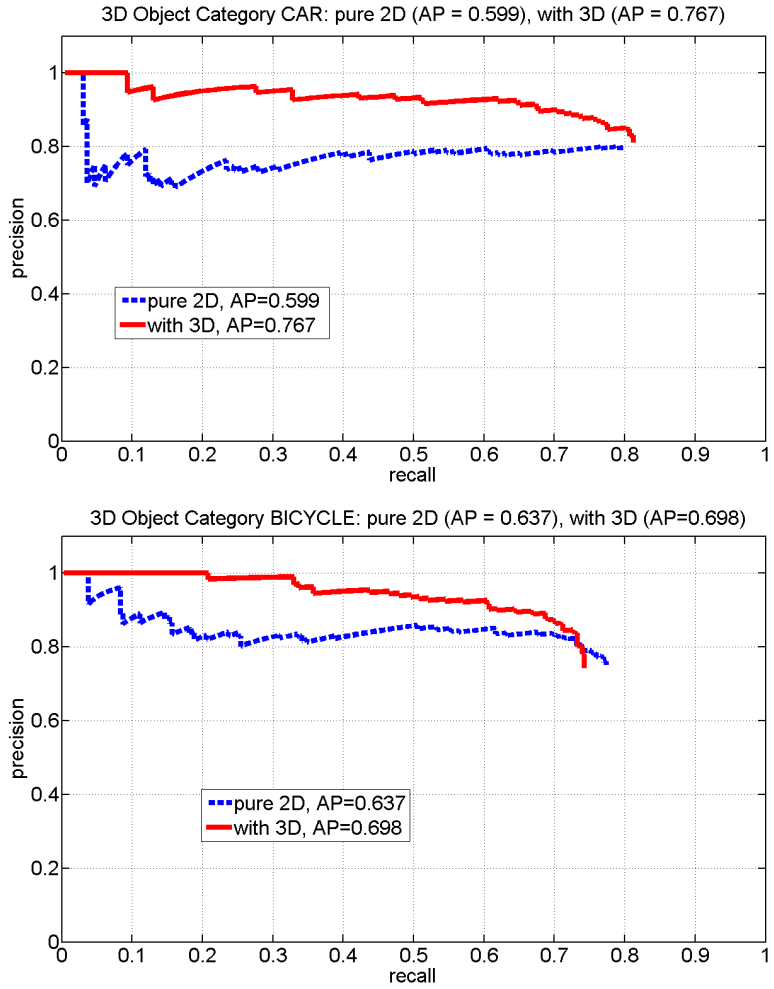

Figure 4.21: Comparison of detection precision/recall without (blue) and with 3D verification on the 3D Object Category datasets [102] CAR (top) and BICYCLE (bottom); the pure 2D predetection can be significantly improved by the 3D geometric model.

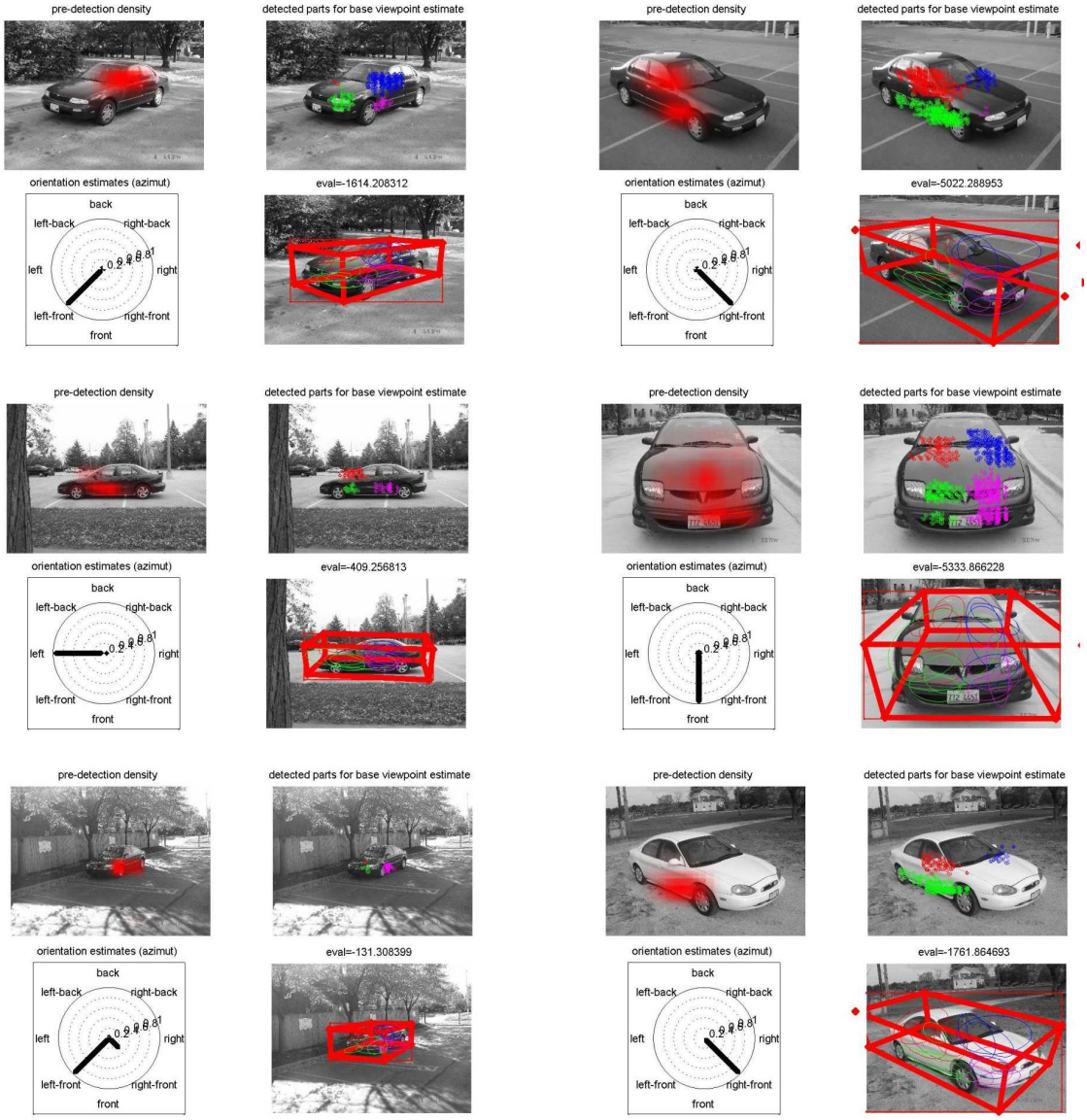

Figure 4.22: Some results illustrating the complete detection process on the 3D Object Category dataset CAR [102]. For each result, the predetection density, the detected parts, the base viewpoint votes and the final pose estimation are visualized.

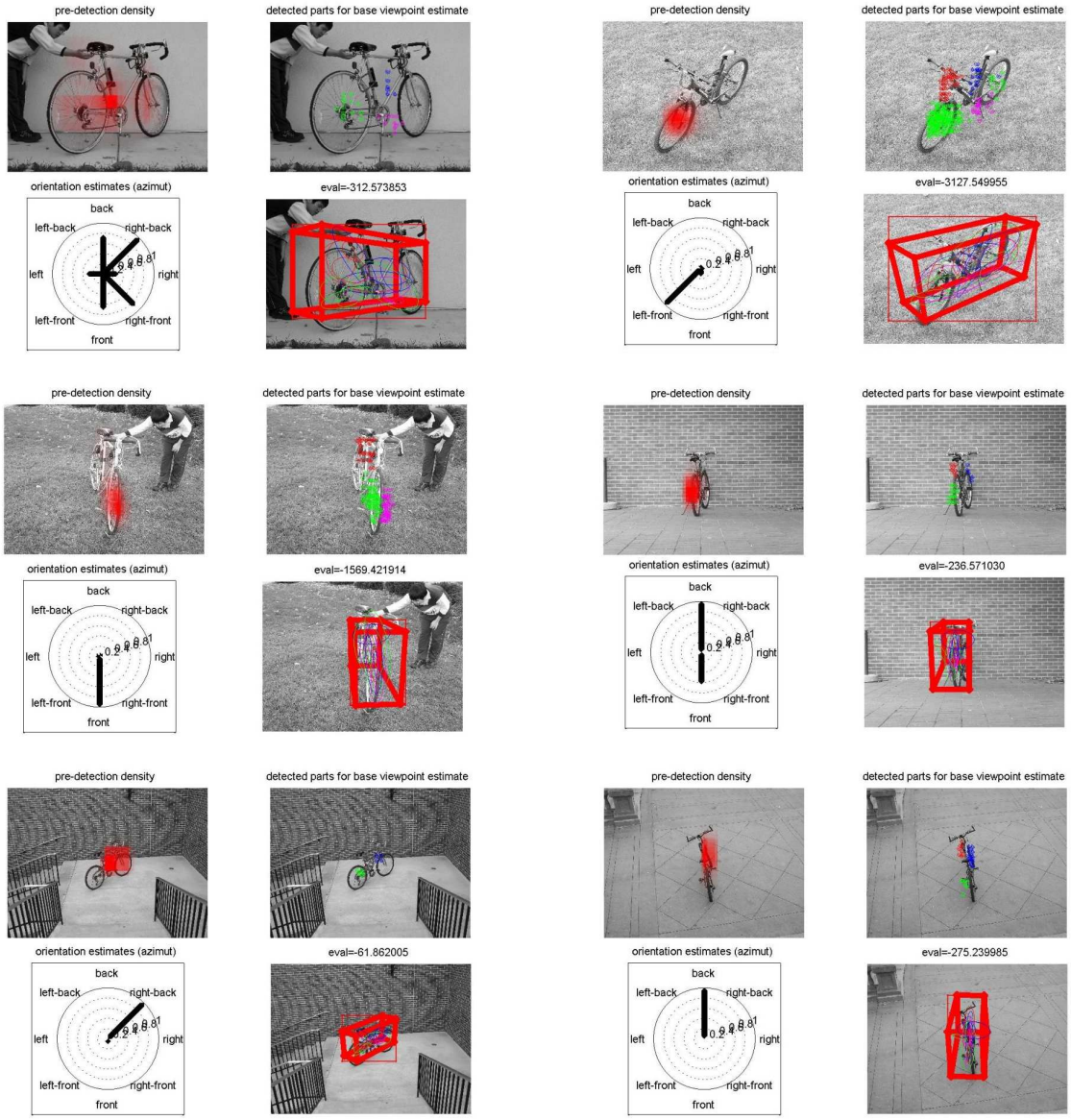

Figure 4.23: Some results illustrating the complete detection process on the 3D Object Category dataset BICYCLE [102]. For each result, the predetection density, the detected parts, the base viewpoint votes and the final pose estimation are visualized.

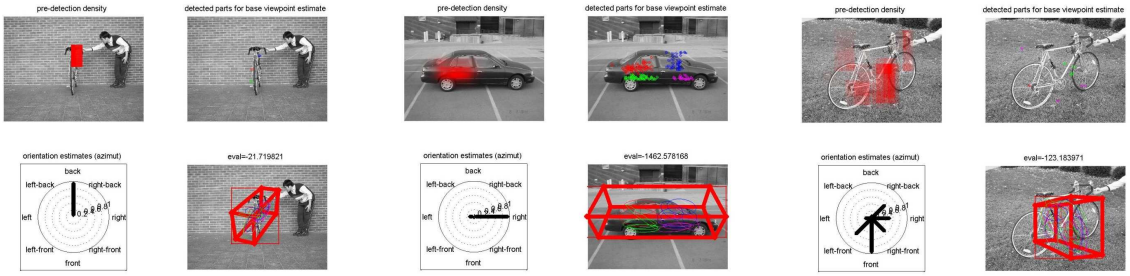

Figure 4.24: Some failed detections on the 3D Object Category datasets CAR and BICYCLE [102]: unstable pose estimation due to sparse detections and a small object area (left), ambiguous elevation estimate (center), scattered pose voting for incorrect detections (right). For each result, the predetection density, the detected parts, the base viewpoint votes and the final pose estimation are visualized.

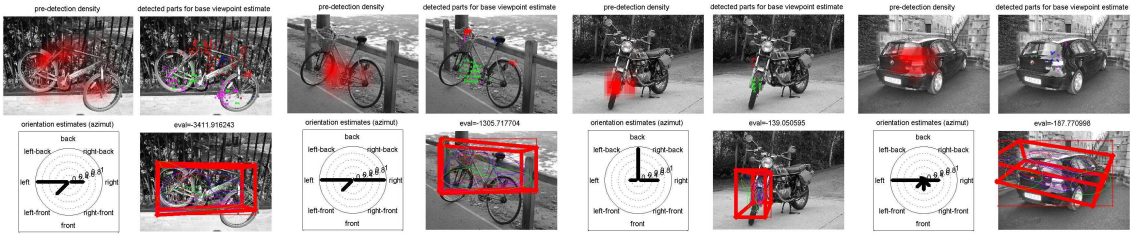

Figure 4.25: Some failed detections on the PASCAL VOC2006 testsets CAR, BICYCLE and MOTORBIKE [18]: (from left to right) camera pose not representable with the chosen parameterization, two incorrect pose estimations due to nonrigid deformations, untrained appearance resulting in wrong pose voting. For each result, the predetection density, the detected parts, the base viewpoint votes and the final pose estimation are visualized.

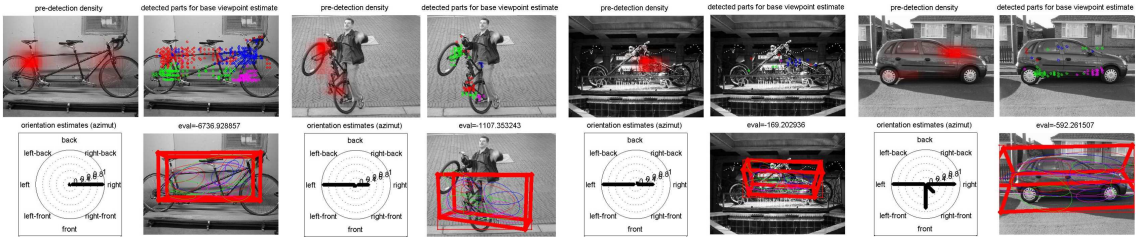

Figure 4.26: Some failed detections on the PASCAL VOC2007 testsets CAR, BICYCLE and MOTORBIKE [17]: (from left to right) unknown geometry, camera pose not representable with the chosen parameterization, unknown geometry, wrong pose voting resulting in incorrect pose estimation. For each result, the predetection density, the detected parts, the base viewpoint votes and the final pose estimation are visualized.

5

Precise Pose Estimation by Registration

In the previous chapters, we have focused on detecting generic object classes in images and on obtaining an approximate estimation of their 3D pose. Since a model of the geometry of a generic object class has to describe large intra-class variations, the obtained pose estimation precision is limited by the amount of uncertainty contained in the model. For many applications, a better pose estimation precision is required. In this chapter, we describe a method to more precisely align a single 3D CAD model to a single image, assuming that an initial approximate pose is given and a model has been selected which corresponds to the actual object in terms of geometry and texture; the detection results of the previous chapters can be considered as suitable initializations.

5.1 Introduction

The registration of 3D models to images in order to recover the pose of an object in a scene usually relies on the optimization of a similarity measure. Typically, a known 3D model of an object needs to be precisely aligned with the signature of the same object in sensor data. In this chapter, we propose a flexible approach to precise 3D model registration, consisting of the generation of the model modalities corresponding to different sensor types, the computation of a robust and accurate similarity measure which combines perspective contour matching and appearance-based Mutual Information [124], and its efficient optimization to recover the six-dimensional object pose. While previous approaches have predominantly focussed on gradient descent approaches, we propose using the evolutionary Particle Swarm Optimization (PSO) [46]. We analyze the suitability and the precision of PSO for the registration task on synthetic as well as on real input images and present a kinetic energy based convergence criterion. We present the results of testing on synthetic and real sensor input to evaluate its precision and convergence properties and apply the system to an exemplary 3D tracking task. It is shown that the registration scheme can sustain significant image noise, small object dimensions and partial occlusion. In addition, we demonstrate that the approach can be efficiently parallelized to exploit the hardware acceleration potential of current generation graphics processors (GPUs).

The chapter is organized as follows: we summarize previous work on CAD model registration in section 5.2, the problem setting is formulated in section 5.3.1 and we outline our choice of a similarity measure underlying the registration process in section 5.3.2. Section 5.3.3 focusses on

the multimodal capabilities of our registration scheme, while section 5.3.4 provides details on the optimization method and its efficient implementation. In section 5.4, we analyze the impact of the optimization scheme on various test cases and in section 5.4.3, we propose a convergence criterion for the optimization method. In section 5.4.4, we show that the registration can be initialized on the detection results of the previous approach in order improve estimation precision and perform a more fine-grained object instance selection.

5.2 Related Work

When looking at the deficiencies of existing model registration approaches, one realizes that a major source of instability arises from their reliance on a common direct metric between the model description and the sensor image. For example, a matching that relies exclusively on the comparison or correlation of intensity values will necessarily fail as soon as the intensity values of the sensor image are significantly influenced by varying lighting conditions. Moreover, the matching of the model and the sensor image has to be performed in the same *data space*, i.e. with some distance metric applicable to both sides of the matching problem. A typical setting is the alignment of sensor readings obtained from standard video cameras, infrared imaging sensors and possibly SAR devices with each other or with a known CAD model structure: although clear functional relationships exist since all devices have captured the same scene, it is impossible to map the sensor readings into a metric space common to all sensors which would allow for the use of traditional distance metrics such as Haussdorff, Chamfer, ECC or SSD. Most publications on image-to-model alignment, for example, resort to various preprocessing stages in order to artificially create an approximation of such a common metric space, usually based on edge filtering (see for example the work of [58]). However, during filtering, significant parts of the information available to the matching system are simply discarded, although they might provide important clues.

An ideal matching approach should be robust towards all changes occuring on the sensor side as long as the observed variations in the sensor image can be explained by the model degrees of freedom and the parametrization of the sensing conditions. Furthermore, it would be desirable to perform the matching between all available sources of information from the model side, even when a representation of a piece of information may not be possible in a metric space common to both the model and the sensor image.

In medical applications of computer vision and image processing, a similar problem exists when trying to align image data obtained from vastly different sensor sources [88]. Typically, the data produced by sensors such as cameras, ultrasound and tomographic devices does not allow for a direct comparison, since the sensor readings require an "interpretation" of some sort. However, when the sensor readings all describe the same scene or object, one can assume the existence of a functional relationship between them. Intuitively, an extended and more flexible metric allowing for the comparison of different sources of information should measure how well one source "explains" another source, regardless of the information type of the compared sources. As a consequence, a number of researchers have come up with the concept of *Mutual Information* as a possible remedy [87, 88, 95, 99].

5.2.1 Measure of Similarity

A number of approaches have been described in the literature to derive similarity measures between models and sensor data.

Some authors resort to computing the difference between 2D-projected edge models and edge-filtered sensor images, for example in [58], thereby increasing generality, but discarding other possibly discriminative information on both the model and the sensor side. Other authors derive shape signatures or similar descriptors from the model and try to identify the corresponding cues on the sensor side by comparing the respective signatures, such as in [133].

In their original work, [124] introduce the concept of *Mutual Information* (MI) as a similarity measure based on information theory and demonstrate its use for aligning untextured 3D objects to images using the interpolated surface normals as clues on the model side. In several publications, extensions to the classical MI formulation are proposed, notably by introducing normalization terms such as in [62] to account for the amount of overlap or weights to account for spatial relationships, usually based on gradients or segmentation [113, 116].

Two fundamental problems of classical MI have been addressed by [100], i.e. the lack of spatial information in the similarity measure and the *curse of dimensionality* which prevented the use of classical density estimators for multivariate MI computation.

In the present work, we propose several extensions to the method of [100], notably the fusion of *Mutual Information* with an edge-based measure, thereby increasing robustness and registration precision.

5.2.1.1 Computation of MI and Estimation of Probability Distributions

Several approaches to computing the MI of two random variables have been introduced in the literature. Essentially, the marginal and the joint distributions have to be estimated using the given sensor readings. In the following, we will briefly summarize some of the most frequently used approaches to estimating these distributions.

5.2.1.1.1 Histograms

The simplest method of estimating probability distributions is to build a histogram counting the number of times a value a_i in one sensor reading coincides with a value b_i in another sensor reading at the same position; each histogram bin is normalized by the total number of evaluated co-occurrences. Using the *Law Of Large Numbers*, one can assume that each normalized histogram bin value (representing the average of a sequence of random variables with a joint distribution) will converge to the common expectation value of the joint distribution, when the number of histogram evaluations are large enough. The marginal distributions can then be computed by simply summing over the rows respectively columns of the histogram. An efficient implementation requires a careful analysis of the spatial requirements of large histograms, in particular since joint histograms are usually sparsely populated.

5.2.1.1.2 Parzen Windows

Another non-parametric estimation method is the *Parzen Window* density estimation. Essentially, it can be described as a sampling whose support is limited by a local windowing function which weights each sample wrt. its distance to the window center. The following equation defines this density:

$$P(x, A) = \frac{1}{|S_A|} \sum_{a_i \in S_A} W(x - a_i) \tag{5.1}$$

where W is a windowing function; the choice of a Gaussian density for W will lead to a Gaussian Mixture Model as outlined in the following section 5.2.1.1.3.

The Parzen Window method combines the advantages of simple histogram estimation, but allows a closed-form computation of its gradients (for additional details on gradient computation and the properties of the Parzen estimation, refer to [125]).

5.2.1.1.3 Mixture of Gaussians

A parametric estimation which is frequently used for random processes is the *Gaussian Mixture Model (GMM)*. The idea behind GMMs is to represent a complex stochastic process as the sum of a number of Gaussian distributions with different unknown parameters. Using a Gaussian Model for the estimation of probability distributions boils down to finding the optimal set of parameters such that the random variables observed in the samples can be accurately described by a combination of Gaussians; parameter estimation can be performed via *maximum likelihood (ML)*- or *Expectation Maximization (EM)*-Algorithms(see [125]). Some authors ([90, 95, 100]) make additional simplifying assumptions on the nature of the distribution in order to use the property of the Gaussian distribution that the entropy of a Gaussian can be directly computed from its variance.

5.2.1.1.4 Influence of Sampling Strategies

All of the above outlined approaches rely on a sampling of the sensor data. In some publications [122], the influence of the chosen sampling method on the quality of the obtained distribution estimates and consequently on the MI itself has been investigated; evidently, smooth convex MI functions are preferable. It has been shown [122] that regular grid sampling may result in artifacts which can cause MI-based extrema search to converge to non-global solutions. The authors suggest resorting to irregular interpolated sampling techniques or employing stochastic sampling to deal with these issues.

5.2.2 Registration to Multiple Modalities

When trying to extract information on objects visible in a scene, precision and robustness can be increased by resorting to different sensors working in different modalities and then combining the captured sensor responses.

Most of the existing methods are confined to working on optical imagery, sometimes exploiting different sensor characteristics such as color and polarization [71], as well as multiple channels

from the model side, such as normals [90] or contours [116]. Most multimodal applications originate from medical 2D/2D and 3D/3D registration tasks, with the exception of [14, 71] who use MI for tracking 2D templates in images.

In our approach, we align 3D objects to different sensor modalities by explicitely simulating and rendering the object appearance for a given sensor modality on the model side not only for optical sensors, but also for *Synthetic Aperture Radar* (SAR) devices.

5.2.3 Optimization for Registration Tasks

Previous work on registration tasks has predominantly relied on gradient descent approaches in order to determine the optimum of the chosen similarity measure. The choice of the optimization scheme depends on the characteristics of the similarity measure as well as on its computation; [124] present several concepts for computing probabilistic similarity measures and outline their optimization using gradient descent.

Due to the lack of directly computable gradients in the formulation of [100] which our similarity measure is based on, we propose an evolutionary optimization approach, Particle Swarming [46], to approach the global optimum. Moreover, this inherently parallel optimization approach is well suited for an efficient implementation on dedicated hardware if the particles are processed simultaneously per iteration; in the present work, we employ the graphics processor (GPU) in a hybrid CPU-GPU implementation.

5.3 Our Approach

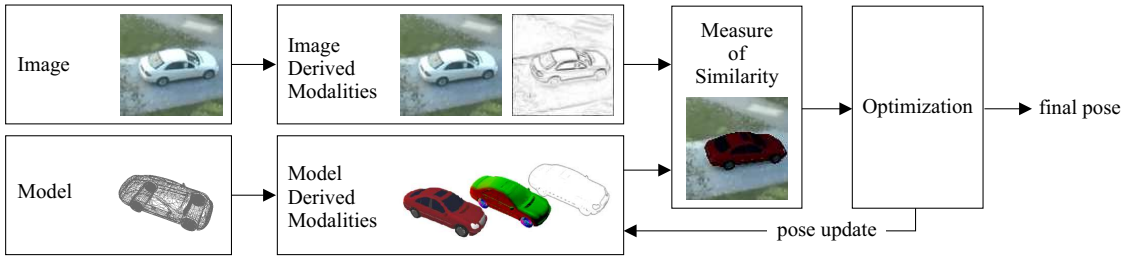

Figure 5.1: Outline of the proposed model registration scheme.

Figure 5.1 gives an overview on the different parts of the proposed scheme for registering a 3D model to 2D sensor inputs. In the following, we provide a formal description of the task and present our contributions to each of the components used to address the problem.

5.3.1 Formal Problem Setting

We suppose that a 3D CAD model M and a vector of sensor modalities $\mathbf{s} = s_1, \cdots, s_n$ are given. The set of model modalities $\mathbf{m} = m_1, \cdots, m_k$ can be described by

$$m_i(\mathbf{p}, \mathcal{P}_i, M) = \mathcal{P}_i \cdot (R \cdot M + \mathbf{t}), \tag{5.2}$$

where the 6 extrinsic parameters $\mathbf{p}$ govern the camera-relative rotation R and translation $\mathbf{t}$ of the 3D model M and $\mathcal{P}_i$ specifies the projection rules for each type of modality m_i with known intrinsic parameters. For an optical camera, for example, $\mathcal{P}_i$ is a perspective projection with intrinsic matrix K.

We now wish to determine for all modalities the common 6 extrinsic parameters $\mathbf{p}$ such that the similarity between the model rendered in each of the k modalities and the n sensor modality inputs is maximized. More formally, we identify the optimal pose $\hat{\mathbf{p}}$ as

$$\hat{\mathbf{p}} = \underset{\mathbf{p}}{argmax}\, S(\mathbf{s}, \mathbf{m}(\mathbf{p})) \tag{5.3}$$

where S represents the similarity measure which will be described in section 5.3.2.

We aim at a flexible approach to model registration which does not require the time-consuming generation of models typical of most previous methods. Instead, we resort to using standard 3D CAD models; see chapter 2 for details on the training data and the synthetic rendering process. Given the choice of the similarity measure, our system can consistently register a 3D model even if its appearance does not exactly correspond to the object visible in the scene, since no direct equivalence between model and sensor object is required, as long as a functional relationship exists. This property is achieved by resorting to a probabilistic similarity measure as discussed in the next section.

5.3.2 Measure of Similarity

Mutual Information was first applied to computer vision problems by [62, 124] and has since become a popular technique both for image registration and feature selection tasks. In the following, we summarize its theoretical basis and describe our own contribution.

Information theory provides a concept for quantizing the amount of new information contained within a signal when interpreted as the possible states of a discrete random variable X, the entropy H

$$H(X) = -\sum_x p_X(x) log(p_X(x)). \tag{5.4}$$

Analogously, the joint entropy of two signals can be computed as

$$H(X,Y) = -\sum_x \sum_y p_{XY}(x,y) log(p_{XY}(x,y)). \tag{5.5}$$

A strong statistical relationship between the two signals reduces their joint entropy, while a weaker relationship will increase this value. Entropy can thus be used as a means of determining how well one signal describes the other by computing the joint entropy of two signals given their separate marginal entropies,

$$MI(X,Y) = H(X) + H(Y) - H(X,Y). \tag{5.6}$$

This term is called *Mutual Information* (MI); it will tend towards zero for completely unrelated signals and assume its maximum, the sum of the marginal entropies, for statistically identical signals where the joint entropy is minimal. Its main advantage over other similarity measures such as SSD is that no direct per-value equivalence is required as long as some form of statistical dependency between the signals can be determined. This concept does not, however, take into account the spatial distribution of a signal, but relies exclusively on a per-element relationship. It is important to note that in this definition of Mutual Information, the influence of the marginal entropies can have unwanted consequences when employing MI for alignment tasks since there is no normalization relative to the amount of actual overlap of the sensor readings. As a result, in the absence of significant overlap, the weight of the joint probability decreases relative to the sum of marginal probabilities. [4] compare several approaches to avoid this problem. The most frequent normalization method is

$$MI_{ECC}(A,B) = \frac{H(A) + H(B)}{H(A,B)} \tag{5.7}$$

as suggested by [62], which we use in our work.

Unlike the registration of entire images, we wish to compute the similarity of a 3D model and a number of images containing input from various sensors. Using the projection of the model into a sensor image, we can determine a precise region of interest which will serve as a mask containing the area of the sensor image relevant for similarity computation. This mask is not precomputed or limited to certain variation modes as is the case for most template-based methods, but instead it varies depending on the current pose of the 3D model. We can thus much more effectively limit the support of the similarity computation to the relevant region in the sensor image and reduce computation time.

When computing the MI between larger image regions and multiple modalities, instead of analyzing the correlation between pairs of one-dimensional pixel values, the data now consists of multi-dimensional vectors of the pixel values in neighbourhoods with a certain radius around each pixel position in each of the modalities. Most traditional approaches to computing the MI as outlined in section 5.2.1.1 are no longer feasible in higher dimensions. Consequently, we follow the work of [100]: if a transformation can be found which projects the high-dimensional vectors into a space where the values of each dimension are uncorrelated, the further assumption of a normal distribution of those values implies independence, allowing to compute the entropy separately in each dimension. Russakoff et al. [100] show that this assumption is justified in practice, which enables us to directly compute the joint entropy of an arbitrary number of d inputs simultaneously from their covariance matrix Cov_d using

$$H(Cov_d) = log((2\pi e)^{d/2} det(Cov_d)^{1/2}). \tag{5.8}$$

In order to further increase alignment precision, we introduce another supporting similarity measure based on edges. We compute precise perspective contours from the available 3D model and determine the fraction of the model contour which matches the edges derived from the sensor input image. This edge similarity E is then multiplied with the MI value using

$$S = (MI_{ECC})^{\alpha} \cdot (E)^{(1-\alpha)} \tag{5.9}$$

where $\alpha \in [0, 1]$ allows to vary the influence of either of the two components. In section 5.4, we analyze the gain in precision which can thus be obtained.

5.3.3 Image Multimodality

In the following, we describe two sensor input modalities which are used in our multimodal registration scheme.

5.3.3.1 Optical Intensity Images

Based on a 3D CAD model, a perspective projection camera model with known intrinsic parameters and a set of light sources, intensity images are rendered. For testing purposes, a background and various noise levels can be added and the resulting intensity images can be used as synthetic input with known ground truth parameters for systematic testing; figure 5.3 shows an example. Chapter 2 provides details on the rendering process and rendered example images.

5.3.3.2 Synthetic Aperture Radar (SAR)

Synthetic Aperture Radar (SAR) is a remote sensing technique which exploits post-processing of a sequence of radar signals over time to fusion the echos from points at different distances. As a consequence, an object remains longer inside the beam of the radar and its resolution can be increased in the resulting radar image by merging the separate echos, given the sensor movement is known. In chapter 2, we described how the rendering pipeline can be modified to produce SAR approximations with sufficient quality for the given task; see figures 2.8, 5.8 for examples.

5.3.4 Optimization

Once the sensor inputs and the model renderings for the given modalities are available, we wish to determine the parameter $\hat{\mathbf{p}}$ which maximizes the similarity measure S described in 5.3.2.

The lack of a closed-form gradient, the costly function evaluations and the necessary alignment precision require a careful choice of the optimization method. Another aspect to take into consideration is the potential of the optimization method for an efficient implementation on acceleration hardware. As a consequence, we propose an evolutionary optimization approach which is experimentally shown to have a suitable convergence behavior (see section 5.4.3.3).

5.3.4.1 Initialization

The objective of this approach is to improve a given approximate pose estimation for a pre-selected 3D model. As outlined in section 5.1, the detection results of the object class detection methods outlined in the previous chapters can be considered as suitable initializations. In section 5.4, we show that our registration scheme is capable of dealing with poor initializations, differences in coloring of object and model and a relatively large search space.

5.3.4.2 Particle Swarm Optimization

In order to overcome some of the difficulties observed with other optimization methods, an evolutionary optimization strategy is chosen. *Particle Swarm Optimization* (PSO) has been introduced by [46] as a concept adopted from nature, where swarms of animals converge towards the richest feeding grounds by communicating between each other. Classical PSO as implemented in this work boils down to randomly placing particles as search agents in a parameter space R^n. Each particle can be described by

1. its position in R^n, $x = (x_1, \cdots, x_n)$, which corresponds to a pose parameter vector $\mathbf{p}$,
2. its velocity $v = (v_1, \cdots, v_n)$
3. and the position vector which has yielded the currently best function value, $b = (b_1, \cdots, b_n)$.

In addition, within randomly chosen subgroups of the swarm, particles are linked with a complete mesh topology amongst each other. Each subgroup can thus communicate the best position currently found by its members, $g = (g_1, \cdots, g_n)$. The link topology and the subgroup affiliations are reassigned at random if no improvement has been made during one time step.

The movement of a particle in each dimension d is governed by

$$\begin{aligned} v_d^{t+1} = \alpha(v_d^t &+ rand(0,\beta_1)(b_d^t - x_d^t) \\ &+ rand(0,\beta_2)(g_d^t - x_d^t)) \qquad (5.10) \\ x_d^{t+1} = x_d^t &+ v_d^t, \qquad (5.11) \end{aligned}$$

where the parameters β_1, β_2 allow varying the influence of the currently known group-wise and globally best function values on the particle movement, α is the so-called *constriction factor* to prevent swarm divergence and $rand(0, \beta)$ adds a random component to the particle behaviour. A review of the properties of PSO is given in [9].

5.3.5 Implementation

Since we are working with 3D CAD models and require a rendering pipeline for the simulation of different sensor modalities, we have chosen to implement our approach as a hybrid system on the CPU and on the graphics processor (GPU). More specifically, the rendering of optical and SAR

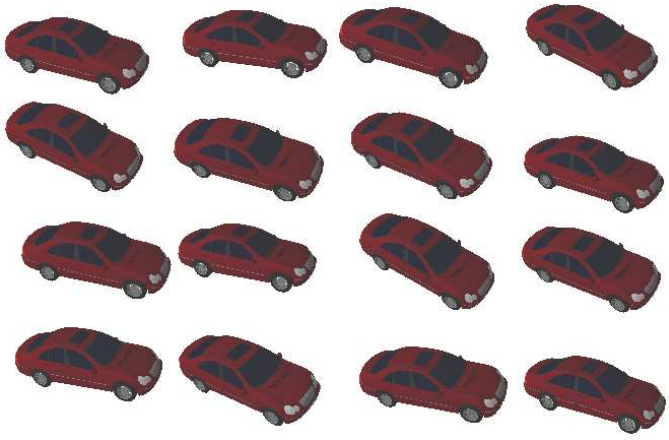

Figure 5.2: Simultaneously rendered model poses of a particle swarm.

modalities on the model side, the contour filtering and the edge distance are computed entirely on the GPU, and the similarity measure described in 5.3.2 is calculated on the CPU.

Figure 5.2 shows a set of rendered particles of a swarm, each of which is associated with a distinct model pose. In the original formulation of PSO [9], particles are updated sequentially and take into account the updates of all previous particles in the batch. In order to allow for a simultaneous update of the particles, which is better adapted to a parallelization of the approach, we chose to update the particles simultaneously after each iteration. Since there are no more interdependencies between the particles in one iteration, their associated model poses can be rendered simultaneously on the GPU and their respective similarity to the sensor input can be computed in parallel. The potentially slower convergence is compensated by the gain in computation time; see figure 5.12. For the experiment shown in figure 5.4, our approach registers a 3D model with 6 degrees of freedom to a single 2D sensor image of size 256x256 on average in about $0.5\,sec$ on a 3GHz processor with an Nvidia GeForce 9600 GPU. The convergence criterion and the optimization parameters used to obtain these results are detailed in section 5.4.3.

5.4 Experimental Evaluation

The proposed model registration scheme has been tested for different object tracking tasks. We use synthetic scenes with known ground truth trajectories as well as real sensor input using a large number of different models.

A test run typically consists of choosing an initial parameter search space and a model and then starting the optimization process. Since we do not use any explicit movement model, it is necessary to specify the center and the range of the parameter search space; the particles of our optimization will initially be placed arbitrarily inside this search space. The pose for the first frame is initialized manually; for each consecutive frame the last estimated pose is used as initialization.

5.4.1 Synthetic Sequences

In order to test the registration precision exhaustively, synthetic input sequences for each sensor modality have been generated by defining trajectories for an object visible in the scene as well as for the camera and the light sources. Gaussian blur and speckle noise with different characteristics can then be added to the image data. Figure 5.3 shows an example image of a synthetically generated scene and its noised version.

In the following, we analyze the influence of noise, initial search space size and edge distance weight on the registration precision. Moreover, we present the results of the model registration to SAR sensor input. In the result plots, we use the standard errors in position and tangential and normal orientation of the model. The position error is given in multiples of the model diameter, while the rotation errors are given in degrees.

5.4.1.1 Noise

A circle is chosen as the ground truth trajectory of the object. The registration process is then initialized on the first of a sequence of synthetic optical images with added noise. Figure 5.4 shows the recovered trajectory (red) as opposed to the ground truth (blue). We systematically increase the noise which is added to the synthetic sequence to analyze the robustness of the registration process. For each noise level used, we plot the standard errors in figure 5.5. The registration process remains stable up to a significant noise level, while the recovery of the rotation parameters appears to be more sensitive towards strong noise than that of the position parameters. Still, registration begins to fail only for noise levels where even the human eye can no longer reliably distinguish the model.

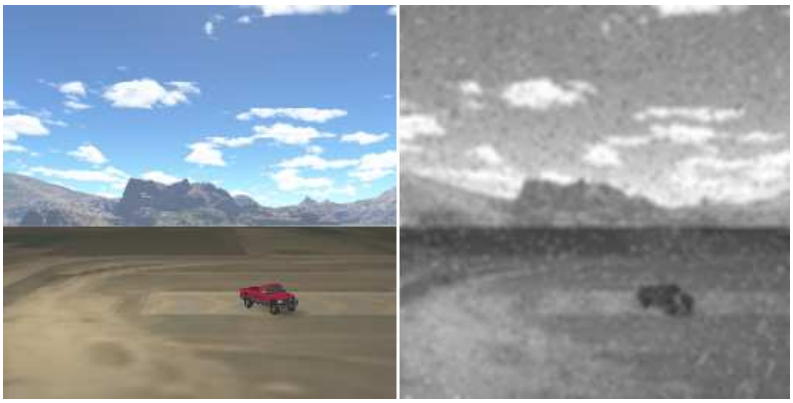

Figure 5.3: Example for a synthetic optical image and its version with added noise.

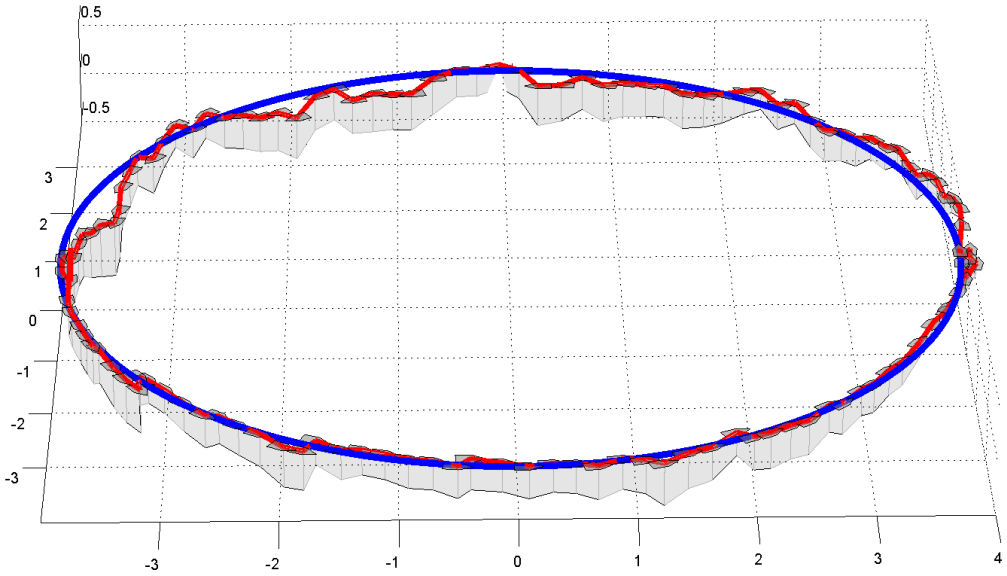

Figure 5.4: Recovered 125-frame trajectory (red) and ground truth trajectory (blue) for a synthetic optical test sequence with added noise; see figure 5.3 for an example image.

5.4.1.2 Search Space Size

In section 5.3.4.2, we outlined the optimization scheme used. The initial search space size is of crucial importance for a successful registration since the particles will tend to get dispersed inside a search space which is chosen too large. As a result, the global optimum might no longer be found reliably. Figure 5.6 shows that, while keeping the swarm size constant, the search space can be safely increased to $\pm 15^\circ$ for rotation parameters and $\pm 40\%$ of the model size for translational parameters without sacrificing an inadmissible amount of registration precision. The result shows that the chosen evolutionary optimization scheme is stable even for less precise initializations.

5.4.1.3 Varying the Contribution of the Edge Distance

In section 5.3.2, we outlined a contour-based extension to the MI similarity measure, allowing to vary the influence of either of the two components as a function of the parameter α. We have experimentally found an optimal choice to be $\alpha = 0.5$, which corresponds to the geometric mean. For this choice, the three standard errors are minimal in the majority of test settings, while for $\alpha \to 0.0$ signifying an exclusive use of the edge distance component and likewise for an exclusive use of the MI measure with $\alpha \to 1.0$, stability and registration precision deteriorate significantly. To illustrate the gain in stability, we registered a model to 605 frames of an input sequence using different α values. In figure 5.7, the frames are shown for which the tracking failed for the first time. When using only the edge distance ($\alpha \to 0.0$), registration failed after 24 frames, when using only the MI measure ($\alpha \to 1.0$), precision was inadmissible after 117 frames showing a characteristic underfitting of the model. The combined measure with $\alpha = 0.5$ completed the sequence successfully and produced precise results.

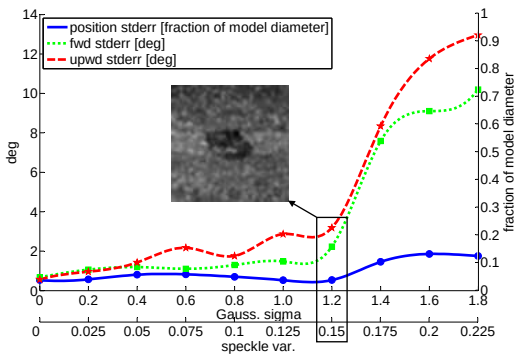

Figure 5.5: Standard errors for a registration sequence of 125 frames (deviation from ground truth position (blue) and tangential (green) and normal vectors (red)) for increasing noise; the optical image of one noise level is shown for visualization.

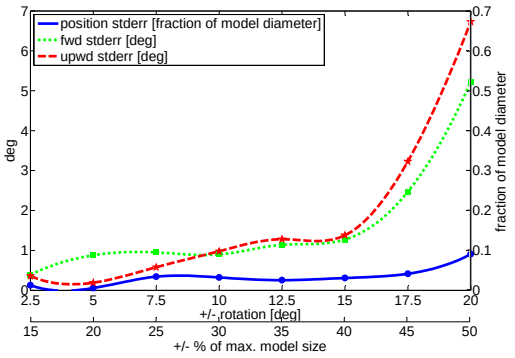

Figure 5.6: Standard errors for a registration sequence of 125 frames while increasing the initial search space size.

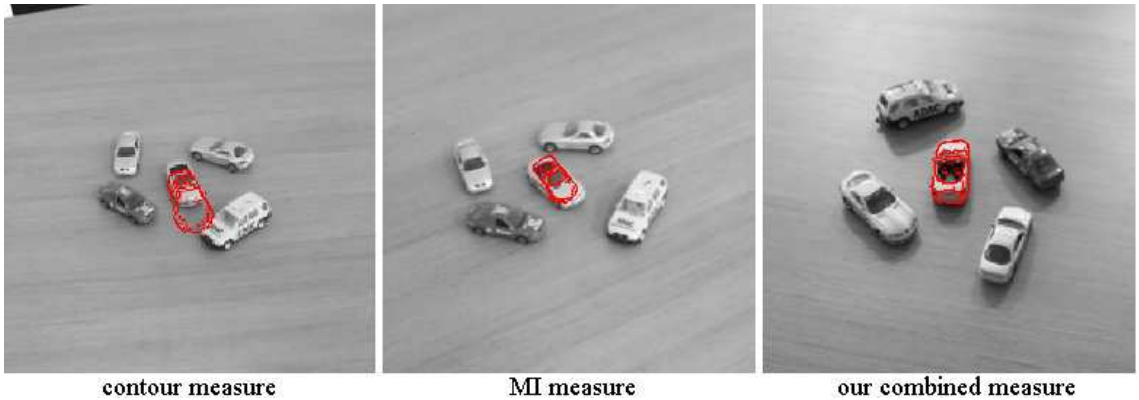

Figure 5.7: Tracking with different α values: using only contour matching, tracking fails after a few frames (left), the MI measure fails after 117 frames (center), the weighted combination handles the full sequence successfully (right)).

5.4.1.4 SAR

Using the simulated SAR response, we performed registration on SAR input data. Figure 5.8 shows two frames of a SAR sequence, with the input images from an industry-grade SAR simulator [66] on the left, the registered SAR signature in the center and the recovered optical 3D model corresponding to the SAR signature on the right. Despite the simple SAR modality generation used, the results show a precision comparable to the optical sequences.

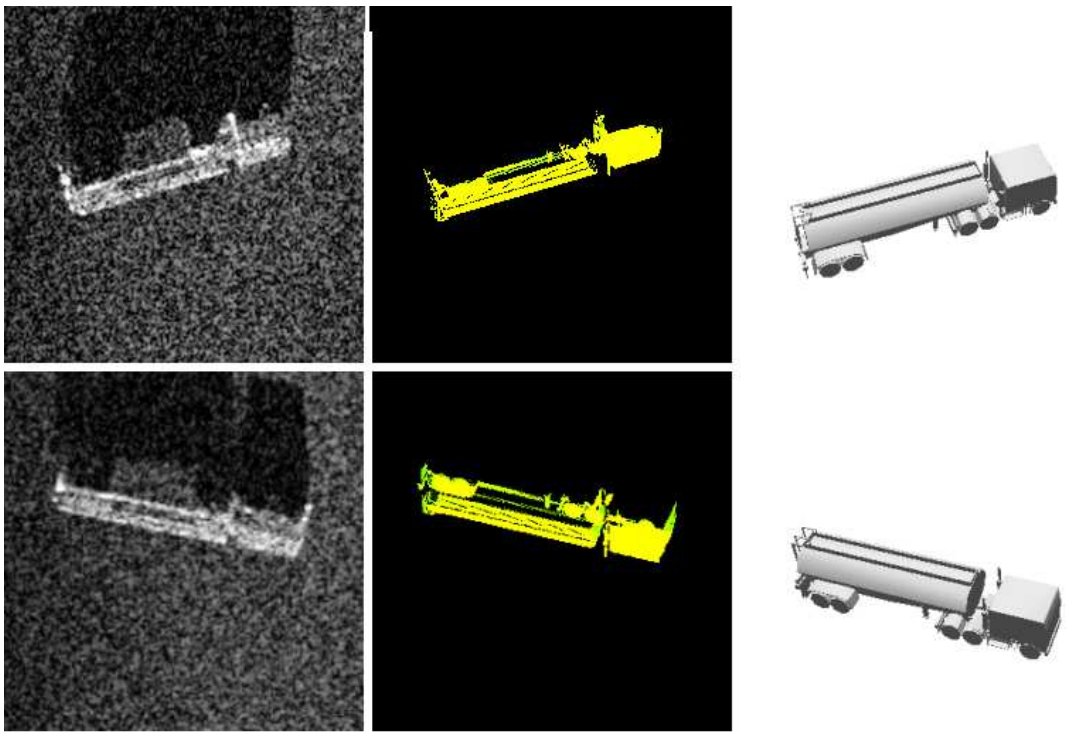

Figure 5.8: 2 Frames from a tracking sequence on SAR input images (left, [66]), the registered SAR model modality (center) and the recovered optical 3D model (right). Images are scaled for better visualization.

5.4.2 Video Sequences

In order to verify the suitability of the proposed registration scheme for realistic sensor input, we registered different models to videos taken with a standard consumer camera. Registration starts on the first frame with the provided initialization, aligns the provided model to the frame until convergence, and uses the convergence result as initialization in the subsequent frame. We did not perform any preprocessing on the input data and did not remove lens distortion, motion blur and artifacts. The tested sequences displayed fully perspective changes and fast movements. Since we recover for each input frame the full 6 pose parameters of the object, we can reconstruct the camera trajectories for the input sequences when supposing that the only movement in the scene stems from the camera.

Figure 5.10 shows some images taken from a 654-frame video of a set of small toy cars on an indoor office table with artificial lighting. For each frame, we show the input image with the reprojected model edges on the left, and the recovered 3D model pose on the right. In figure 5.9, the spline-interpolated camera trajectory for the same scene is plotted together with the model position and the camera position and orientation for each of the 4 images shown in figure 5.10. The average size of the toys in this sequence varies between 60x40 and 120x80 pixels.

In figure 5.11, we present more results taken from different input sequences. The topmost frame is taken from another indoor sequence of a different toy car, while the next two images show real cars in an outdoor setting. Due to the chosen robust similarity measure, significant occlusions of the object can be sustained as illustrated in the third image of figure 5.11. The registration is also suitable for textured objects with shiny surfaces, even in complex settings, as can be seen on the last image of figure 5.11 for an indoor recording of a soda can. Model sizes in the tested sequences vary from 80x60 to 160x120 pixels with video resolutions of 256x256 and 512x512 pixels. For non-matching objects, the value of the similarity measure after convergence ranges an order of a magnitude below the result for a correct match, thus allowing to determine whether the choice of the model for registration to a given scene ought to be reconsidered.

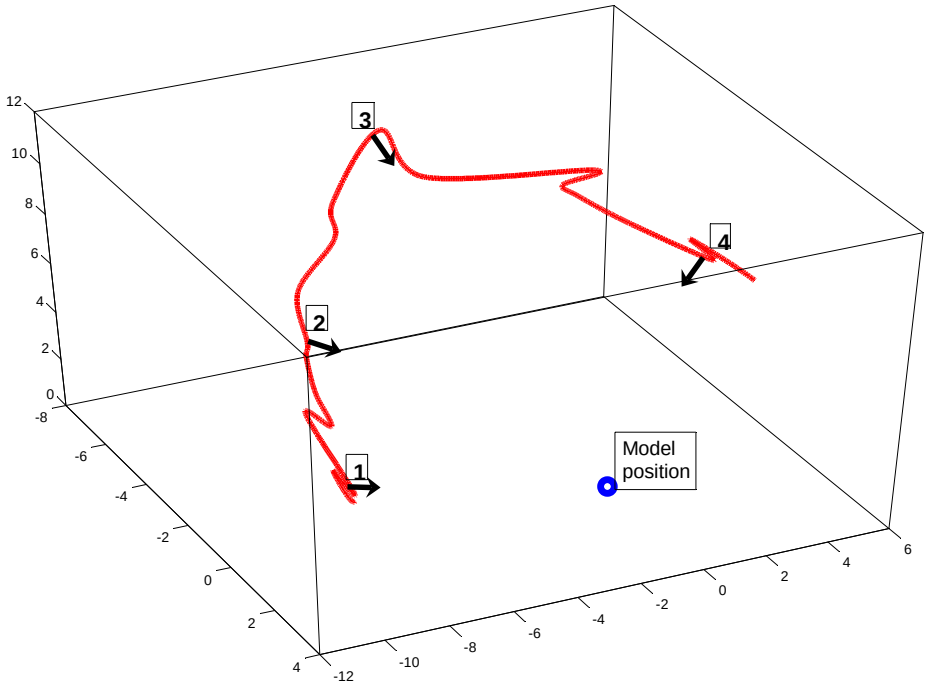

Figure 5.9: Recovered camera trajectory of the tracking sequence 5.10 with the camera positions of the 4 frames shown in the figure.

5.4.3 Optimization Results

In the following, we analyze the behavior of the chosen evolutionary Particle Swarm Optimization method. All experiments have been performed on a 125 frame synthetic optical input sequence to provide accurate ground truth data for precision analysis. As in the previous sections, we specify the registration errors by resorting to three indicators: the position error is given in percent of the major model axis length and the errors in forward and upward model orientation are given in degrees.

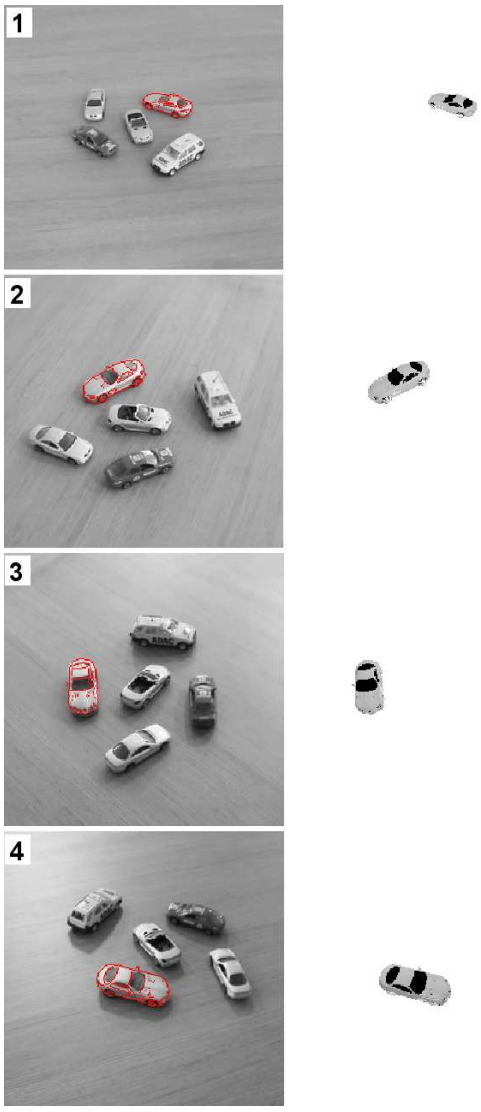

Figure 5.10: 4 Frames from a 654-frame sequence with toy cars; input image with recovered object (left), recovered 3D model (right).

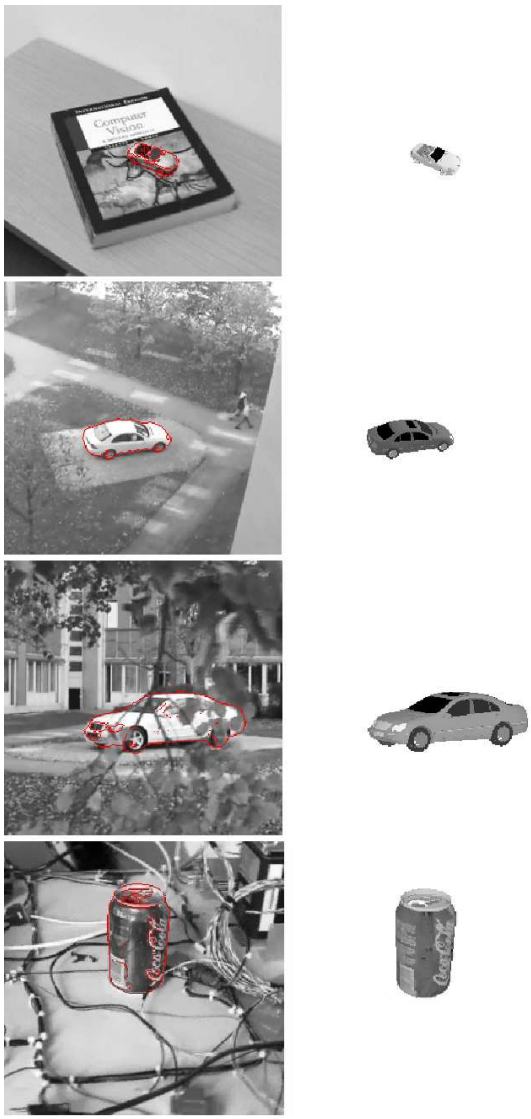

Figure 5.11: Selected frames from four different registration sequences and the recovered 3D models.

As regards the choice of the constriction and weighting parameters for the PSO algorithm, we follow the suggestions in [9], since they were corroborated by our experiments. The choice of the remaining parameters, notably the number of synchronously updated particles of the swarm and the initial search space size, will be outlined in the following sections.

Given the nature of the PSO method as an evolutionary approach with a stochastic component, we focus on three aspects of the Particle Swarm Optimization in order to justify its use for our problem setting: first, we will show in section 5.4.3.1 that the average per-particle kinetic energy, assuming a particle mass of 1, can be used to characterize the swarm's convergence behavior. In section 5.4.3.2, we analyze the influence of varying the swarm size when supposing that the position and velocity of all particles are updated simultaneously. Finally, in section 5.4.3.3 our choice of the decrease in the swarm's kinetic energy as a suitable convergence criterion is presented and justified.

5.4.3.1 Kinetic Energy of the Swarm

The behaviour of the swarm is difficult to analyze due to the complex interdependencies of the individual particles. Local features such as the forming of clusters of particles or the average distances between particle positions are ill-suited for this optimization method whose principal advantage is its global and simultaneous sampling of the search space. Moreover, several authors (e.g. [9, 45]) have noted that outlier particles represent a particular strength of the approach in escaping local optima. As a consequence, we propose to use the average per-particle kinetic energy of the swarm in order to describe its current state. We assume a particle mass of 1 and compute the kinetic energy of the swarm after each synchronized update of the particle positions and velocities.

When the swarm settles down on an optimum, the average kinetic energy of the particles converges towards the purely stochastically induced level governed by the parameters β_1, β_2 in equation 5.11. Once the kinetic energy has fallen below this level for several consecutive iterations, no significant further improvement of the optimum can be expected.

In figure 5.12, left, the average per-particle kinetic energy is plotted over 100 iterations for three different search space sizes; in terms of our model registration scheme, this signifies bigger admissible misregistrations of the model during optimization. The different energy levels shown in figure 5.12, left, are due to the larger search space. Since the position gaps between the particles are bigger, the velocities which the particles are assigned after each iteration are higher as well. Moreover, we can conclude that different problem configurations can result in different energy levels after the same number of iterations.

In figure 5.12, right, kinetic energy curves are plotted for the same search space size, but different swarm sizes. We note that the energy levels are lower for higher particle counts. Since the search space is more densely sampled by higher particle counts, the average inter-particle distances are smaller; the particles therefore have smaller gaps to bridge when moving towards the current global optimum position, which results in smaller assigned velocities. Both plots show, however, that the reduction in the swarm's kinetic energy indicates a converging of the swarm independently of a particular swarm configuration or problem setting.

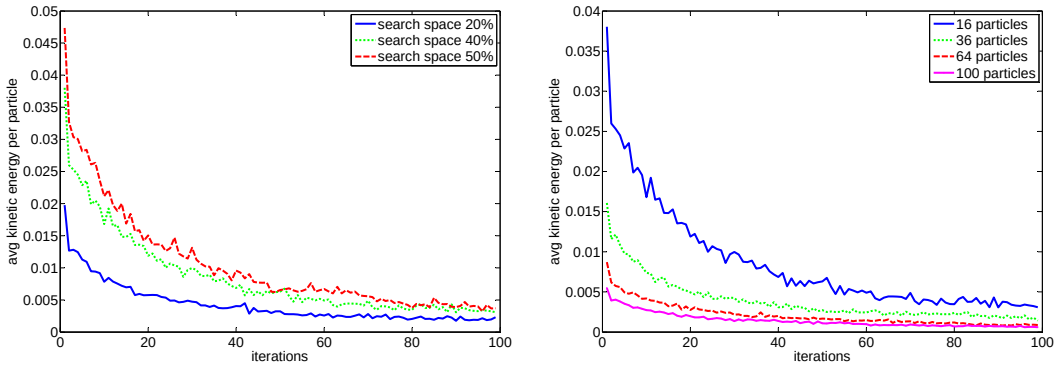

Figure 5.12: Average per-particle kinetic energy for three different search space sizes (left) and four different, synchronously updated swarm sizes (right) in a 125-frame synthetic sequence. The search space sizes are given in percent of the model size.

5.4.3.2 Number of Particles

In section 5.3.5, we have described that only by using a synchronous update of the particles of a swarm, the full hardware acceleration potential can be exploited. This choice of simultaneously updating particle positions and velocities has a significant impact on the convergence behaviour of the method. For asynchronous, sequential updates, every new particle position leads to a new evaluation of the similarity measure. In this case, the computed value can immediately be taken into account for the next particle update. This is not the case for synchronous updates, where all updates rely on the same previous evaluations. As a consequence, one has to expect a slower convergence of the swarm. In figure 5.13, the average errors achieved by swarms of different sizes for the synthetic test sequence with known ground truth are given for five different numbers of similarity function evaluations. It becomes apparent that while increasing the number of particles of the swarm results in a better final precision (third row), the swarms with smaller particle counts converge much faster to acceptable errors. As a consequence, the swarm size needs to be adapted to the specific optimization goal: when a high final precision is essential, higher particle counts should be chosen, whereas smaller particle counts are better suited when execution time is a critical aspect and a slightly reduced precision can be sustained. For our system, a swarm size of 36 represents the best trade-off between precision and execution time as can be seen in figure 5.13. This confirms the findings of [109].

5.4.3.3 Convergence

Although our similarity measure is normalized to values in the interval $[0..1]$, the precise value depends on the entropy of every single input frame, therefore the maximum value of 1 is hardly ever reached in practice. As a consequence, we do not know the value of the optimum in advance and cannot employ an absolute convergence criterion, such as an ϵ neighbourhood.

The results shown in the previous two sections suggest the use of the average per-particle kinetic energy as a means of determining whether the optimization procedure has converged. From the previous results, we have also seen that the absolute energy levels vary depending on factors such

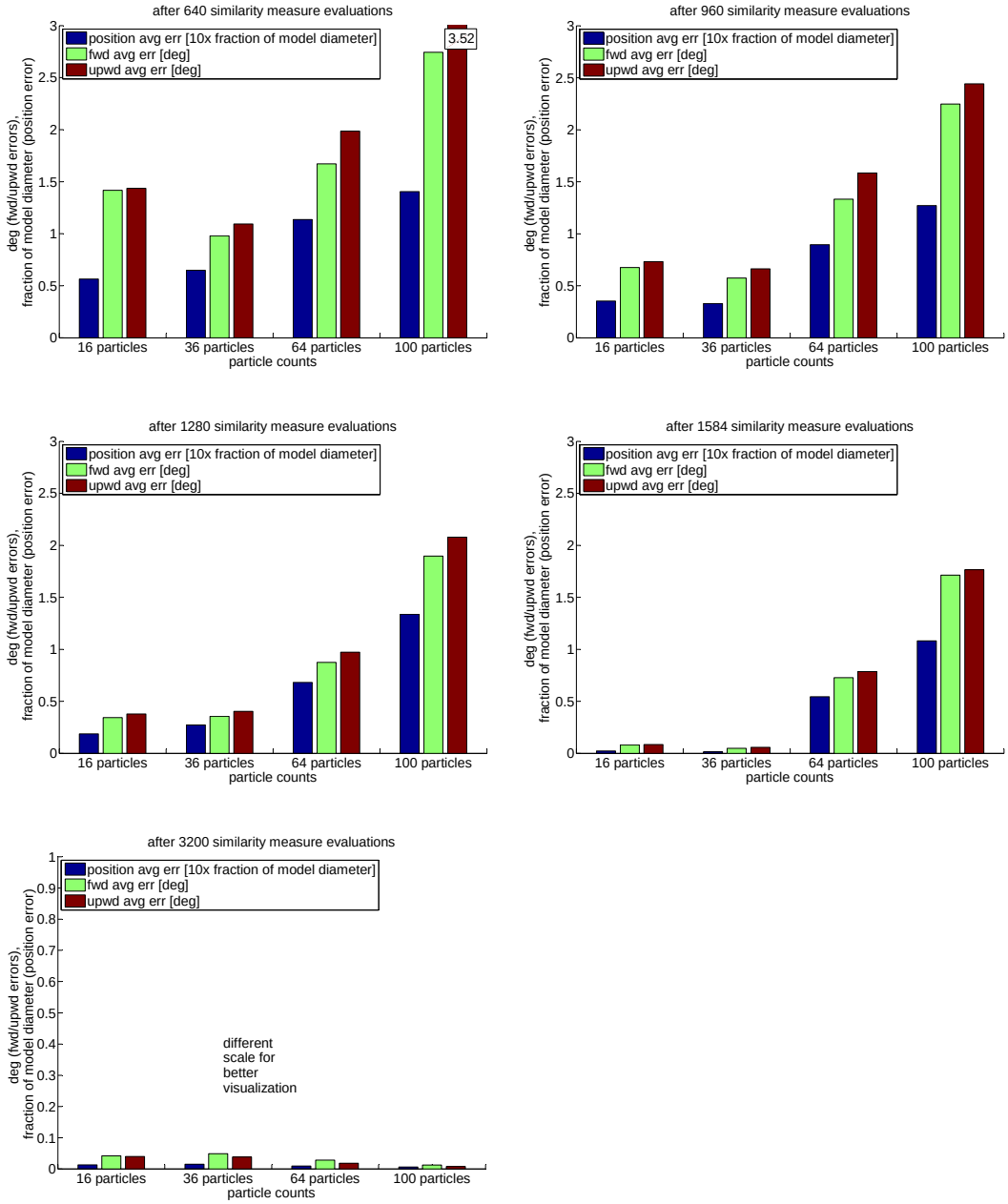

Figure 5.13: Average errors in position (up-scaled by a factor of 10 for better visualization), forward and upward orientation for swarm sizes of 16, 36, 64 and 100 particles after 640, 960, 1280, 1584 and 3200 similarity measure evaluations.

as the search space size, the number of particles used or the initial misregistration. However, a convergence criterion based on the reduction of the kinetic energy relative to the initial energy level is less dependent on such external factors. In figure 5.14, we plot the three average errors in position and forward and upward orientation for two different search spaces. In the first row, 100 iterations of the swarm optimization have been performed. It becomes apparent that the achieved errors differ significantly after the same number of iterations; as a consequence, neither a localized error measure nor a fixed iteration count can reliably indicate convergence independently of the specific problem setting. In the second row of figure 5.14, we plot the three error indicators over the kinetic energy and end the optimization procedure when the initial kinetic energy has been reduced by more than 90% in five consecutive iterations; the threshold is drawn as the rightmost vertical line in figure 5.14. It becomes apparent that this criterion allows to end the optimization process while guaranteeing a certain error level independently of the external problem configuration. For a different threshold chosen at a reduction of 80%, the error levels for the two different search spaces remain comparable as well. The same property holds true for different swarm sizes.

In our model registration scheme, this convergence criterion has shown to produce suitable results. Typically, the registration of a model occurs in several steps which result in larger improvements of the similarity measure, followed by several steps with produce smaller improvements. The kinetic energy level is well suited for this characteristic of the similarity measure: large improvements result in a higher overall kinetic energy of the particles, thus allowing for an extensive sampling of the area around the newly found optimum. The energy level then slowly settles down and allows for a fine-grained sampling of the close neighbourhood around the optimum. In a nutshell, the reduction of the swarm's kinetic energy can serve as a convergence criterion which can be adapted to the desired final precision.

5.4.4 Closing the Loop

In the previous chapters, approaches to object class detection have been introduced which are characterized by their use of 3D geometry during training in order to allow for an approximate pose estimation during testing in addition to a 2D localization in the image. For tasks where a pure 2D localization is sufficient, the 3D information can still be useful for pruning the 2D detection results by evaluating their plausibility in terms of a full 3D model, but the actual outcome of the pose estimation is not used. The registration method outlined in this chapter requires an initialization consisting of a specific object instance and its approximate 3D pose. Consequently, it can serve as a suitable use case for the pose estimation results of the previous chapters; furthermore, the classification results can help to narrow down the range of potential object instances on which a registration is performed. In the following section, we describe an experimental setup to demonstrate the feasibility of this idea, thereby closing the loop between the object detection and the registration components of the present work.

5.4.4.1 Dataset

With a standard digital camera (Canon EOS), a sequence of 24 images of a toy car (Skoda Fabia) is taken under indoor lighting conditions in a calibrated setup at resolution 2352×1568. During the

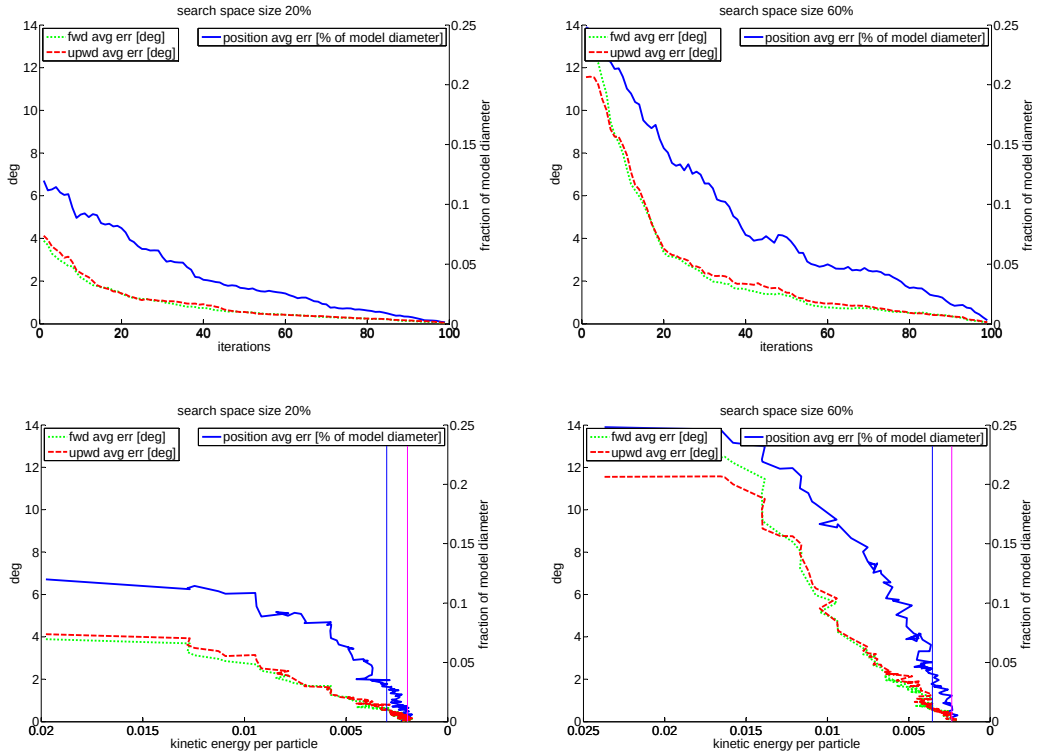

Figure 5.14: Average errors in position, forward and upward orientation for a small (left column) and a large search space (right column). The top row plots the errors over the optimization iterations, where one iteration comprises a synchronous update of 36 particles. The bottom row plots the errors over the kinetic energy. Two possible convergence thresholds are drawn as vertical lines.

sequence, the camera moves on circular trajectories at two different elevations around the object. The intrinsic camera parameters are derived from a calibration pattern with the approach of [135] and used for undistortion and configuration of the synthetic camera.

The probabilistic geometric model approach in chapter 4 is trained on 6 3D CAD car models (3 hatchback, 3 notchback models; see figure 5.18 for the model names); both geometry and appearance training are performed exclusively on synthetic images, rendered in the discrete parametrization from section 4.3.2 using a synthetic camera whose intrinsic parameters match those of the real calibrated one. Note that the pure synthetic training differs from the combination of synthetic geometry and real appearance training data used in chapter 4 and shows the flexibility of the approach.

5.4.4.2 Detection

On each image of the sequence, we apply the detection approach from chapter 4 to obtain an initial detection. The detection and approximate pose estimation results on the first 8 images of the sequence are shown in figure 5.15. Since appearance training has been performed on synthetic images, fewer object parts are detected; the resulting pose estimation still shows an acceptable localization, but in some cases a notable deviation in elevation and distance. The estimated extrinsic

camera parameters (relative to the synthetic camera) and the classification information are passed as (synthetic) initialization to the registration step. In the comparison experiments in section 4.3.2, we have shown that the approach from chapter 3 achieves a slightly better estimation accuracy than the probabilistic approach from chapter 4. Since the focus of this section is on improving the initial detection-based estimation results with the registration method proposed in this chapter, we rely on the less precise probabilistic approach as initialization in order to clearly demonstrate the robustness of the registration.

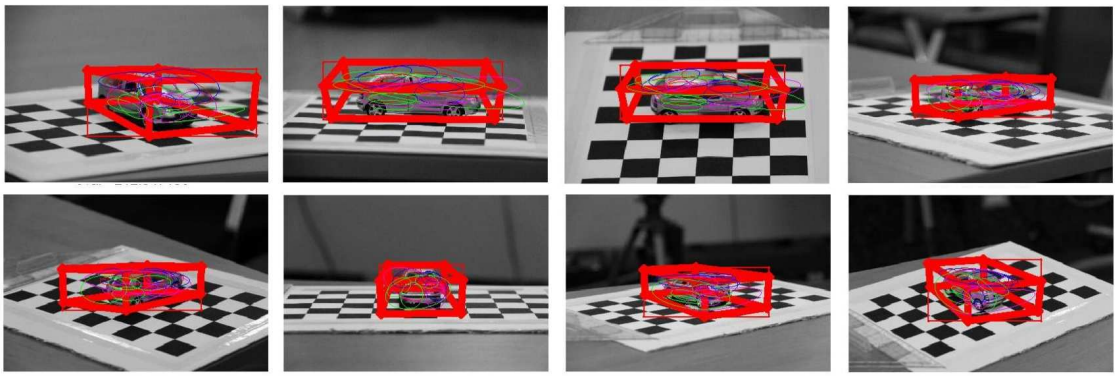

Figure 5.15: Object class detection result with the approach described in chapter 4 for the first 8 images of the test sequence.

5.4.4.3 Choice of the Object Instance

Given the approximate initial pose of the synthetic camera, the intensity and edge images of the 6 car models used for training are rendered; on each car model and for each test image, the registration converges to a new pose which locally maximizes the similarity measure outlined in section 5.3.2. Figure 5.16 shows the initial (left two) and the converged pose (right two) of one test instance; the left columns contain the test images, overlaid by the object instance rendered in the current pose, and a visualization of the similarity measure (red bar). The right columns again show the object instances rendered in the current pose for better visualization. In figure 5.17, convergence results for the three highest-scoring instances on the first test image are given. The corresponding instance (Skoda Fabia, left) converges to the highest similarity score (0.22), followed by the two other object instances with a hatchback geometry (0.18 and 0.20); the three notchback geometries (not shown in the figure) score lowest. As intended, similar colors alone do not influence the score, but similarity in terms of contour and relative color distribution does. The corresponding object instance (Skoda Fabia) scores best in all 24 images of the sequence (see figure 5.19 for some examples). Figure 5.18 compares the similarity scores of the 6 3D car models on the first 8 test images; the score values are listed in table 5.1, along with the average absolute values and relative differences over the entire sequence. The corresponding object instance (Skoda Fabia) consistently scores highest, hatchback car geometries score higher than notchback geometries. The comparison shows that a reliable differentiation between the three hatchback car instances is more difficult for profile and rear views which result in close similarity scores, whereas the views with several sides of the object visible can be differentiated more easily.

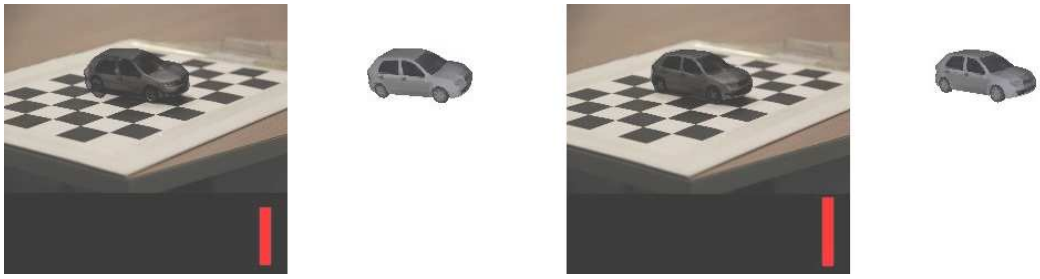

Figure 5.16: Initial pose (left two images) and result of the registration after convergence (right two images) using the best-scoring 3D car model for one image of the test sequence. The first columns show the test images with the 3D model in the current pose overlaid; the red bar indicates the similarity score.

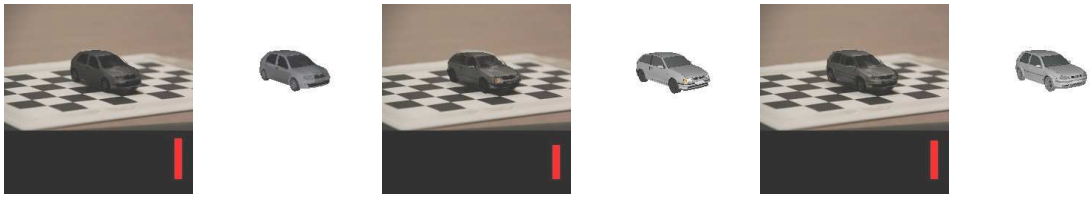

Figure 5.17: Comparison of the similarity scores of the three 3D car models which score highest on the first test image among the 6 training models used. The corresponding object instance (left) scores highest, followed by the other models with hatchback geometry.

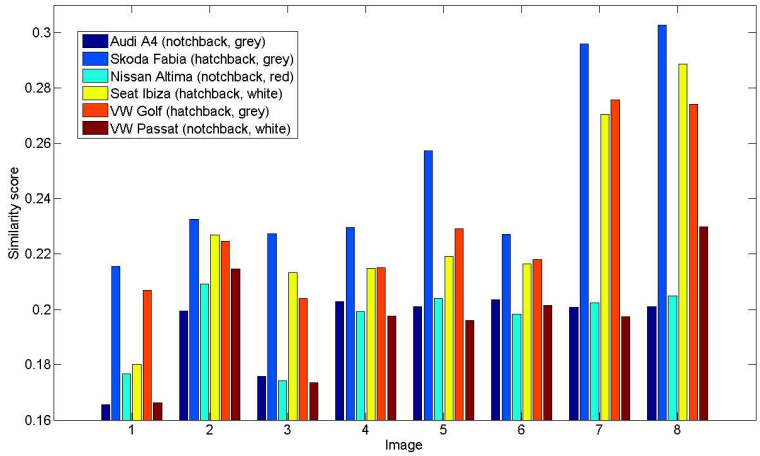

Figure 5.18: Comparison of the similarity scores of the 6 3D car models on the first 8 test images. The corresponding object instance (Skoda Fabia) consistently scores highest, followed by the other models with hatchback geometry. A better differentiation based on the similarity score is possible when several sides of the object are visible.

Table 5.1: Similarity scores for the first 8 images and average values over the entire test sequence. The last row gives the average relative differences between the car instances in terms of the similarity score; as intended, similar geometries achieve similar scores.

Image	Car instance					
	A4	Fabia	Altima	Ibiza	Golf	Passat
1	0.165641	0.215583	0.176648	0.180102	0.206852	0.166227
2	0.199323	0.232634	0.209172	0.226992	0.22456	0.214637
3	0.17581	0.227349	0.174109	0.213399	0.204057	0.173462
4	0.202893	0.229699	0.199265	0.214792	0.215131	0.197615
5	0.201015	0.257301	0.204009	0.219266	0.229141	0.196001
6	0.203445	0.227192	0.198345	0.216436	0.218167	0.201366
7	0.200721	0.295967	0.202379	0.270651	0.27579	0.197367
8	0.201118	0.30278	0.204993	0.288804	0.274159	0.229949
Average absolute score (24 images)						
	0.1937	0.2486	0.1961	0.2288	0.2310	0.1971
Average relative difference to best (24 images)						
	-22.05%	0	-21.1%	-7.95%	-7.07%	-20.71%

5.4.4.4 Registration Precision

Figure 5.19 shows the convergence results of the first 8 images in the sequence for the best-scoring model; figure 5.18 plots the similarity scores for each of the 6 training models on the same 8 images. Despite the crude initial pose estimates generated by the detection step, note that the correct instance consistently scores highest and converges towards better poses, followed by the other two models which have a similar hatchback geometry; table 5.1 provides the absolute and relative score differences between the 6 models. Images with larger visible object surfaces allow a better discrimination between the model instances, since differences in geometry result in more significant alignment errors (diagonal views in images 5,7,8), while on profile, frontal and rear views, the score differences are smaller.

Figure 5.20 visualizes the improvement in camera pose estimation for the first 8 images from a diagonal and a top-down view; the camera positions and orientations as visualized by pyramidal cones, where the initial estimate is plotted in red, the improved estimate after registration in blue, and the groundtruth in green; the car position is indicated by a red point. Apart from the first test image, the registration procedure significantly reduces alignment error, in particular in terms of camera distance and elevation. Table 5.2 provides the improvement in orientation and position in absolute and relative numbers before and after the registration step; camera orientation is compared in terms of the angle between the camera directions (note that camera roll is not representable in the chosen camera model). On average, the initial position estimate is improved by more than 66% to within on average $8cm$ of the groundtruth position; the remaining difference is mostly due to underestimating the distance to the object which results from the difficulty of the detection step in correctly estimating camera distance, given a very generic representation of the 3D class geometry. The initial orientation estimation is improved by more than 41% to within on average 4° of the groundtruth orientation; in most test cases, the initial azimuth estimate is already close to groundtruth, while initial elevation estimates are less precise. Since the object detector provided only a pose relative to the virtual camera used during its training, no metric pose can be determined initially; instead, a virtual unit of one model diameter is assumed. When using the

calibration information of the test sequence, we can repeat the mapping procedure described in section 3.4.3.2 of chapter 3 to determine the metric 3D pose in real camera coordinates that corresponds to the detected synthetic 2D pose and compute metric errors in centroid position and box orientation between measured groundtruth and estimated object pose; instead of per-class bounding boxes, we use the exact boxes that completely contain the object instance and consequently obtain an exact solution to the mapping from synthetic to real pose.

Table 5.2: Comparison of pose estimation improvements for the first 8 images and average values over the entire test sequence. We provide the differences w.r.t. groundtruth for the orientation after initial detection (second column) and after optimization (third column) in rad as well as the position after initial detection (fourth column) and after optimization (fifth column) in cm; average values over the entire sequence achieved by the registration are given in the last rows.

Image	initial orient.	optim. orient.	initial position	optim. pos.
	vs. groundtruth			
1	0.2351 rad	0.3192 rad	20.0 cm	10.0 cm
2	0.0978 rad	0.0227 rad	12.0 cm	4.1 cm
3	0.0749 rad	0.0523 rad	16.0 cm	5.1 cm
4	0.0409 rad	0.0698 rad	7.0 cm	2.1 cm
5	0.2282 rad	0.0790 rad	40.0 cm	20.4 cm
6	0.0941 rad	0.0268 rad	30.0 cm	15.0 cm
7	0.0404 rad	0.0193 rad	30.0 cm	5.5 cm
8	0.0812 rad	0.0193 rad	35.0 cm	8.2 cm
Average absolute estimates (24 images)				
	0.12 rad	0.07 rad	24.5 cm	8.2 cm
Average relative improvement (24 images)				
		41.67%		66.54%

5.5 Conclusion

In this chapter, we have proposed a simple but efficient similarity measure derived from joint entropy estimation and perspective contour matching. In conjunction with a robust evolutionary Particle Swarm Optimization strategy and a kinetic energy based convergence criterion, we have been able to register 3D models to 2D sensor inputs of different modalities. The approach offers a suitable tradeoff as regards precision, speed and universality and completes the object class detection approaches described in the previous chapters towards a more precise 3D pose estimation and the identification of specific object instances.

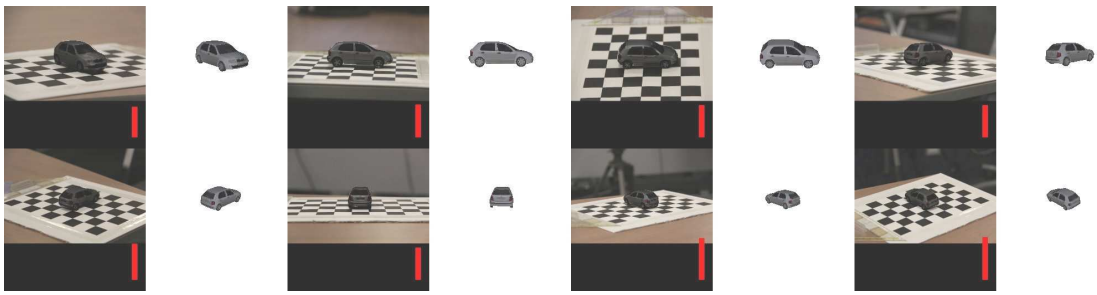

Figure 5.19: Convergence results of the best-scoring instance on the first 8 images (overlays and rendered visualizations).

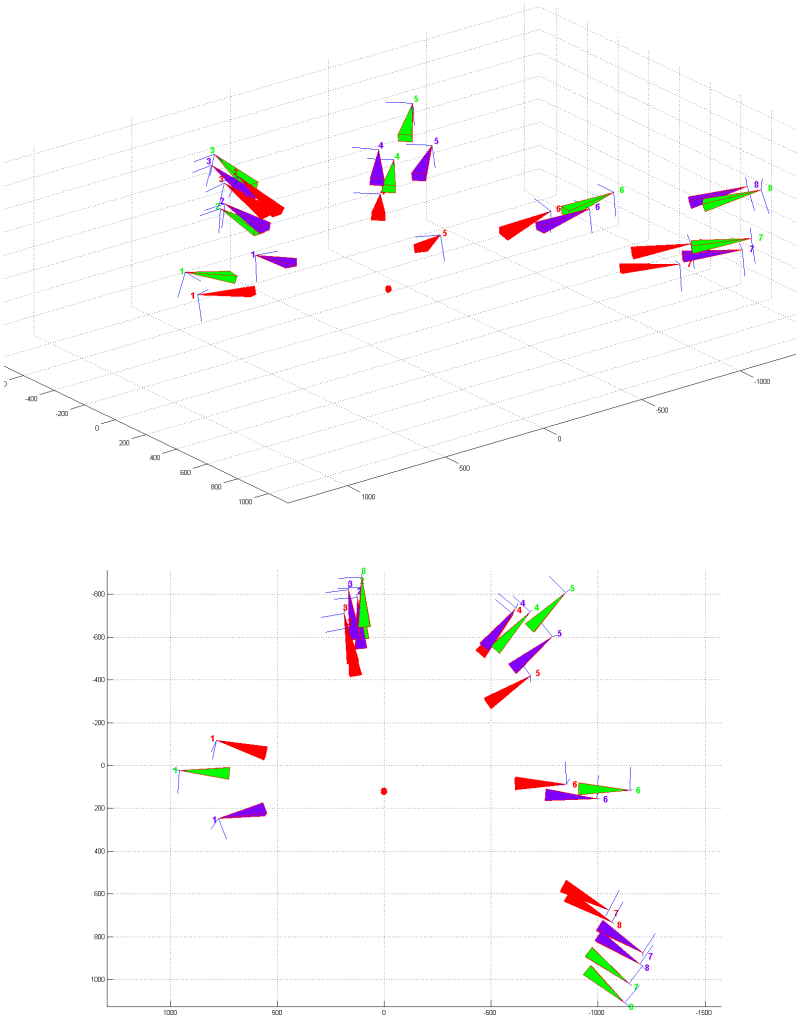

Figure 5.20: Visualization of the camera poses for the first 8 images of the sequence from a side view and a top-down view after mapping into a real metric coordinate system (section 3.4.3.2 of chapter 3). The camera positions and orientations are visualized by pyramidal cones, where the initial estimate is plotted in red, the improved estimate after registration in blue, and the groundtruth in green; the car position is indicated by a red point.

6

Conclusion

6.1 Summary

In this work, we made use of synthetic 3D CAD models for the computer vision tasks of object class detection and 3D pose estimation in the following three different ways:

- Textured 3D CAD models of an object class are rendered from different viewpoints. For each viewpoint, local features are computed and annotated with their 3D position on the model surface. A sequence of small viewpoint changes and different backgrounds allows to filter the local features according to their discriminativity and stability. The retained features after filtering and assignment to a codebook of clusters form a 3D feature map. During detection, local features from a test image are assigned to their nearest neighbors in the feature map according to a combined distance function which takes into account descriptor similarity and geometric projection error. Three different pose estimation methods are evaluated for the task of determining the pose which maps the 3D feature map of a given class onto the test image such that the combined distance is minimized. To initialize the feature map for the estimation task, a probabilistic voting scheme is introduced to identify the most likely viewpoint hypotheses. When applying the method on several benchmark image databases, the detection of 2D bounding boxes after reprojection and the 3D approximate pose estimation achieve precision comparable to state-of-the-art. During evaluation, several limitations of the approach are identified, notably a lower recall compared to pure 2D approaches due to local features computed from synthetic appearance instead of real images and a strict 3D pose estimation without 2D fallback which fails for feature constellations not seen in training examples.

- To further improve detection performance, a second approach to object class detection was developed which takes into account the shortcomings of the initial method, notably its limited extensibility and a reduced descriptor similarity due to training on pure synthetic data. In this approach, appearance and geometry learning are separated to avoid the dependency on synthetic textures. Appearance is learnt from a set of real 2D images with weak viewpoint annotations; for each viewpoint, the object area is subdivided into a regular grid of object part regions. Within each part region, spatial pyramids are computed to densely cover the respective region. The spatial pyramids are learnt discriminatively using SVM classifiers

for each part and viewpoint based on a minimum intersection kernel. Synthetic CAD models are rendered from the same viewpoints and the renderings are subdivided into the same grid used for the appearance learning step. The 3D surface points which project into the respective areas of the subdivision grid form the training data for the geometry learning; for each part and viewpoint, Gaussian Mixture Models are learnt generatively to describe the corresponding distribution of 3D surface points. The link between appearance and geometry is implicitly established via the use of the same regular grid structure in both learning steps. During detection, spatial pyramids are computed densely over the entire test image and classified according to their part and viewpoint membership. Given the 3D geometry and the most likely viewpoint hypotheses with their associated parts, a probabilistic pose estimation is performed which allows to evaluate the consistency of the detected 2D parts relative to a full 3D projective geometry. The method could potentially allow for a 2D fallback and it offers a more robust estimation than the closed-form methods. Evaluation on several benchmark datasets for 2D detection and 3D pose estimation shows superior results when compared to the initial approach, in particular in terms of recall; albeit at a slightly reduced precision.

- The object class detection approaches allow to locate an object of a certain category in an image and provide a 2D bounding box estimate and an approximate 3D pose estimation. However, in many applications the identification of a specific object instance, such as the make of a car, and its precise 3D pose are required. To address this task, we suggest a similarity measure allowing to evaluate the similarity between a specific rendered 3D CAD model and a 2D image region. The similarity measure combines an entropy-based measure to robustly compare intensity values without requiring exact color equivalence and a contour-based distance. Based on the proposed similarity measure, the suitability of an evolutionary optimization scheme is evaluated in order to robustly align a 3D CAD model onto a 2D image region while maximizing their mutual similarity. The optimization requires initialization with an object class and an approximate 3D pose, both of which can be provided by the object detection methods discussed previously. On a calibrated test sequence, the complete detection and precise pose estimation loop is evaluated successfully.

6.2 Future work

The present work describes an approach which offers efficient and flexible training and achieves state-of-the-art 2D detection precision/recall in addition to a 3D pose estimation. It is among the first to resort to synthetic CAD models for the training of object class detection methods. Consequently, many of the opportunities residing in the suggested approach could not yet be investigated. In the following, we provide a short outline of future lines of work which build on the results of the present thesis.

6.2.1 3D Scene Geometry

In the present work, individual pose estimations are made for each object detected in an image. While being flexible and allowing for simplified parameterizations, this does not exploit any ex-

isting priors on the 3D geometry of the scene in the image. These priors could be computed from the image by exploiting vanishing lines, regions with certain properties indicative of a coherent alignment, occlusions and boundaries as has been proposed by several authors, notably [41]. Alternatively, scene geometry may be known, as is the case for certain video surveillance applications. Assuming priors on the 3D scene geometry, the detection process can be adapted to restrict detections to a set of admissible viewpoints. Furthermore, the present work does not make use of the context of a detection. While most previous work on the use of context of pure 2D detections exploits co-occurrence relationships such as in [111], the availability of 3D geometry allows to make stronger assumptions which go beyond pure co-occurrence; only recently, Bao et al. [3] have suggested relying on 3D detections as reconstruction clues. Figure 6.1 shows an example of a set of detections suitable for such geometric co-occurrence priors; the present approach detects each object instance and its 3D pose individually and independently of other detections in the same image (6.1, left) with slight imprecisions in their poses which could be resolved when grouping consistent detections. It detects only one instance in figure 6.1, right, although the knowledge of the successful detection could be used to identify more instances with similar poses when assuming a common scene geometry. The detection of several car objects in a similar orientation can for example serve as a detection prior for further car objects in image positions coherent with the perspective geometry defined by the found 3D poses. The priors are therefore no longer limited to influencing the occurrence probability of other object categories, but can have an impact on the probability of viewpoints, scales and appearance of additional objects as well.

Figure 6.1: Example for a suitable testcase for 3D geometric co-occurrence priors: currently, all detections are independent of each other, but precision could be improved from co-occurrence priors (left). Only one instance is detected (right) which could contribute to detecting more instances with similar poses when assuming a common scene geometry.

6.2.2 3D Model-Driven Part Annotation

Both object class detection methods developed in this thesis essentially deploy 2D image-based techniques to identify suitable clues for appearance learning. In the first approach, local features are computed in 2D images rendered from different viewpoints and their stability over a sequence of 3D transformations is evaluated only with respect to their 2D appearance similarity. The second approach subdivides images into a regular grid of parts without considering the suitability of

certain object regions for representing a consistent part. Neither of the two approaches exploits the semantic and geometric information already residing in synthetic CAD models for the part selection process. This shortcoming could be alleviated in the following two ways.

6.2.2.1 Semantic Model Decomposition

The design procedure of CAD models usually requires the artist to build a model from geometric primitives in a successive way. Most of the object parts are assigned semantic descriptions in natural language as part of the assembly procedure. Previous work on combining object detection and natural language semantics such as [64] have shown promising results. While part annotations could be done manually in 3D, the procedure is tedious and potentially impossible for significant intra-class variation. However, one could attempt to analyze the semantic descriptions of the CAD designer to identify parts which belong to similar semantic groups, thereby allowing to consistently annotate part regions directly and automatically from the 3D model. Knowledge of the semantic hierarchy of natural language may be required in this process since no standardization exists for the design of CAD models. Once part annotations have been identified on the 3D model, an arbitrary amount of precisely annotated 2D renderings can be generated, thereby significantly simplifying and extending the training process and permitting to identify individual parts in new images by their natural descriptions and semantic context. Figure 6.2 shows examples of two natural candidates for semantic part annotations in a database of synthetic 3D car models.

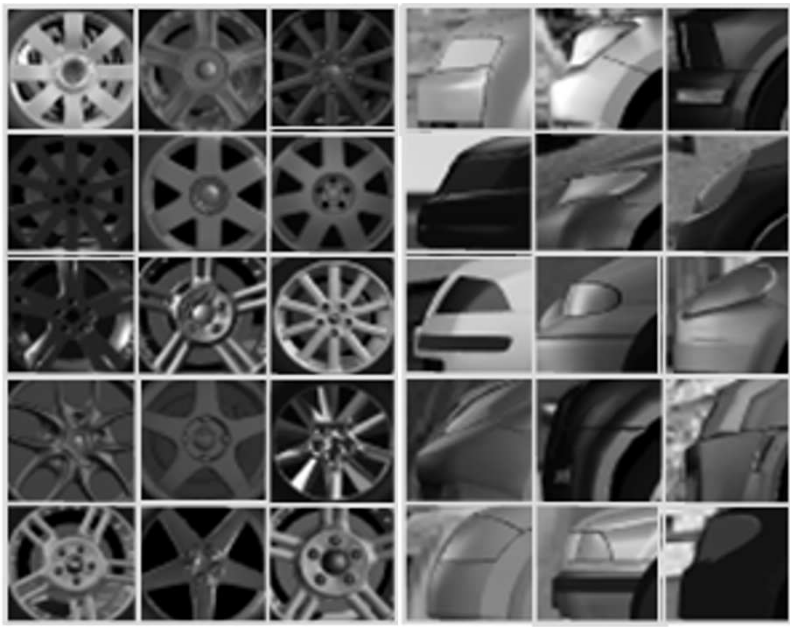

Figure 6.2: Two semantically chosen object parts (car wheel rim, front light) from synthetically generated training examples; illustration courtesy J. Schels.

6.2.2.2 Geometric Part Discovery

In the absence of semantic annotations of 3D CAD model parts, the surface geometry can be exploited to identify parts over a sequence of models of the same object category and even to group models into categories in an unsupervised manner. Several authors from the computer graphics domain, such as [31], have come forward with techniques to automatically segment 3D models into parts according to surface properties, including curvature, connectedness, smoothness and common texturing. Based on such segmentation maps of a set of 3D models, completely unsupervised category discovery and part assignment could be possible. Figure 6.3 shows example segmentations for the object class *chairs* from the work of [31].

Figure 6.3: Unsupervised 3D model segmentation for future use as part annotations; figure from [31].

6.2.3 Classes of Nonrigid Objects

All approaches developed as part of the present work require rigid object geometries. While this requirement is approximatively true for many categories of manmade objects, natural object categories usually have the capacity of being deformable. The best-suited example is the human body; no rigid part layout is capable of representing even the most frequent body part configurations. Animals or plants may display even more pronounced nonrigid geometries. Still, one of the strengths of computer graphics is animation and many algorithmic tools exist for building kinematic chains of model parts and performing surface smoothing in joint areas. Similarly, CAD databases contain deformable models for most object categories, although usually not obeying any standardized design process. Consequently, the generation of training data for nonrigid object categories is conceivable and geometric configurations of object parts can be brought to a normalized representation. Still, the detection of nonrigid objects would require integrating nonrigid deformations in

a suitable parameterization into the pose estimation process, a problem which has seen significant progress in recent years in the context of human body motion [97] and facial action recognition, where the 3D information available during the training process on synthetic CAD models can also be advantageous [56].

6.2.4 Semantic Separability of Object Classes

In the introduction, section 1.1, we defined the term *object class* by means of *semantic equivalence* and limited the evaluation of the present work to three exemplary and rather vast object classes. The limitation to a few exemplary object classes may be valid for a general assessment of the algorithmic concepts of the present work; however, it will cease to be applicable in real-world scenarios where it might be necessary to process large numbers of, potentially ill-defined or overlapping, object classes. Consequently, a systematic approach to dealing with the definition of object classes will have to be found and the training process will have to be adapted to using varying levels of detail for separating the classes. By aligning the training process with a comprehensive semantic hierarchy such as the one described in [69], training could account for overlap and semantic similarities in a more principled way. Moreover, through the use of synthetic data, it could become feasible to exhaustively evaluate with how much intra-class variation a given training method can reasonably deal with before a separation into different subclasses may be required.

6.2.5 Assessing Training Success with Synthetic Training and Validation Data

In the present work, discriminative learning constitutes an essential building block for feature and part detection. Certain methods exist to evaluate and improve the training success, notably cross-validation and bootstrapping as outlined in section 4.2.2.2 of chapter 4. Furthermore, previous work has been done [89] to assess the impact of training set selection on the performance of discriminative classifiers. Yet, many decisions on which training data characteristics to include in order to achieve a certain classifier behavior rely on heuristics and interpretation. For example, it is considered an advantageous property of a classifier to generalize to instances which are not part of the training set. Such a behavior can be induced by either increasing the number of training samples to include as many of the possible variations in intra-class appearance, noise, camera and lighting conditions as technically feasible and counting on the representational power of the chosen classifier, or one could choose to provide fewer, possibly simplified training instances and thereby achieve an implicit generalization on only those statistic properties which are common to the concept to be learnt. Interestingly, this area has not been investigated thoroughly in the domain of object class detection, possibly because neither of the two approaches can be realistically achieved when relying on real training image databases, since no real sample can be considered either a natural simplification or an exhaustive representation of all variations. We believe that in the present work, we have been able to provide a proof of concept of the transferability of synthetically generated data to real-world object class detection tasks; subsequently, the reliance on synthetic data could offer a possible solution. Synthetic training data can be generated and parameterized to cover the entire range from overly simplistic comic-like to photorealistic rendering, and arbitrary amounts of validation data can be created such that particular aspects of a specific

detection task can be covered and the results can be fed back to dynamically adapt the training set selection. Such a closed-loop self testing and training in addition to a quantifiable and meaningful assessment of training success would constitute an important step towards the deployment of learning-based methods in real-world application scenarios.

6.3 Concluding Remarks

We have shown in the present thesis that synthetic CAD models can serve as a readily available source for integrating 3D geometric information into object class detection approaches. Moreover, we demonstrated that the use of synthetic textures for appearance training is feasible and has certain advantages over the use of real training image datasets; however, we also conceded that on current large-scale benchmark datasets [17, 18], state-of-the-art results still cannot be achieved without including real images into the appearance training process. Consequently, we proposed a combined approach to flexibly fuse real appearance and synthetic geometry training data. The results obtained suggest that 3D geometric priors can help to improve precision and to indicate and discard failed detections which are inconsistent with the assumed geometry. Among the potential future lines of work, the possibility to create a systematic validation loop for object class detection approaches on synthetic training and test data seems particularly promising in order to prepare for the transition of object class detection from scientific benchmark datasets towards industrial application scenarios with stricter reliability requirements; we believe that vision applications in the domain of autonomous robotics may benefit the most from the approximate 3D pose estimations obtained with our approach.

A

Publications

A.1 Refereed Conferences

The work presented in this dissertation has been published in part in the following conference papers:

- J. Liebelt and K. Schertler. Precise registration of 3D models to images by swarming particles. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.
- J. Liebelt, C. Schmid, and K. Schertler. Viewpoint-independent object class detection using 3D feature maps. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.
- J. Liebelt and C. Schmid. Multi-view object class detection with a 3D geometric model. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.
- W. Ensinger, C. Stahl, P. Knappe, K. Schertler, and J. Liebelt. Toward a robust 3D-model-based ground target classification system for airborne platforms. In *SPIE Conference on Defense, Security and Sensing*, 2010.

A.2 Patents

The present work was part of an industry cooperation with EADS Innovation Works Germany; as part of this cooperation, the following patents have been filed:

- J. Liebelt, K. Schertler, and F. Schubert. Image processing device, 2010. Patent no. DE102008014381, accepted.
- K. Schertler and J. Liebelt. Learning apparatus for an object detection system, 2010. Patent no. DE200810057979, pending.
- K. Schertler and J. Liebelt. Automatable reconstruction method, 2010. Patent no. DE200810057176, pending.

B

3D Model Database

In the following, we visualize all 3D CAD models contained in the database that was used in the present work. They stem from different free and commercial CAD model databases, notably turbosquid.com, 3d02.com and doschdesign.com; see section 2.2.1 for details. For visualization, we rendered the models using the parameterization described in chapter 2.

Figure B.1: All 3D models used to represent the object class *motorbike*.

Figure B.2: All 3D models used to represent the object class *car* (1/2).

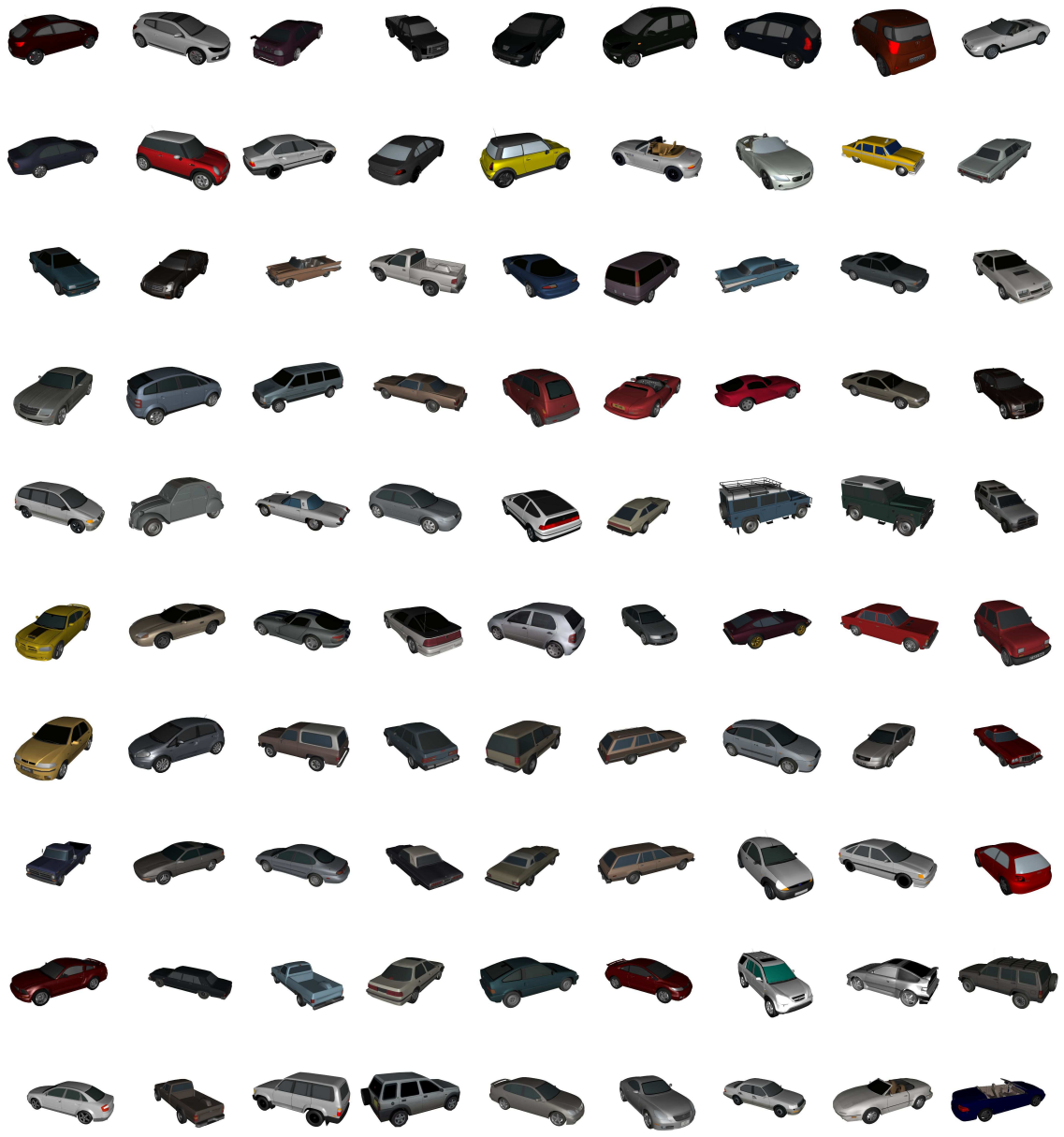

Figure B.3: 3D models used to represent the object class *car* (2/2).

Figure B.4: All 3D models used to represent the object class *bicycle*.

C

Support Vector Machine Classifiers

In chapters 3 and 4, we rely on *Support Vector Machine* (SVM) classifiers. A full derivation of *Support Vector Machines* is not in the scope of the present work; we provide an overview of the concept, see [108]. Suppose a classification problem where we are given a training set of N data vectors x_i, each associated with a target value of t_i with $t_i \in \{-1, 1\}$ indicating whether a data vector x_i belongs to the relevant class ($t_i = 1$) or to an unspecified background set ($t_i = -1$). The objective is to find a decision function y such that at least for all training vectors x_i,

$$y(x_i) \begin{cases} > 0, & t_i = 1 \\ < 0, & t_i = -1 \end{cases}. \tag{C.1}$$

In addition, a desirable behavior of y is to generalize this property to unseen test data vectors, assuming that they are drawn from a reasonably similar distribution as the training vectors themselves. *Support Vector Machines* (SVMs) have been suggested as an approach to this classification problem [108]. They rely on *kernel functions* $k(x_i, x_j) = \phi(x_i)^t \, \phi(x_j)$, a dot product between the data vectors mapped into a higher-dimensional vector space by means of a potentially non-linear mapping function ϕ, to compare pairs of data vectors; in the following, kernels are assumed to be *Mercer* kernels [108], which notably corresponds to their being positive semidefinite. The decision function for a test data vector x can then be written as a weighted sum of kernel evaluations between x and (a subset of) training data vectors x_i,

$$y(x) = \sum_{i=1}^{N} \alpha_i \, k(x_i, x) + \beta = \sum_{i=1}^{N} \alpha_i \, \phi(x_i)^t \, \phi(x) + \beta, \tag{C.2}$$

where the training data vectors x_i with nonzero weights α_i are called *support vectors*; they define a plane which separates the higher-dimensional vector space into subspaces according to their class affiliation. Efficient optimization methods for training large data sets have been proposed [86] by reformulating the task as a quadratic optimization under constraints such that the separating margin between the subspaces is maximized; we use the approach of [19]. The flexibility of the margin can be governed by an additional parameter $C > 0$. The classification problem then reduces to the definition and evaluation of a suitable kernel, the computation of the α_i and the choice of parameters β and C.

D

Rapport de Thèse

D.1 Introduction

Selon des études récentes, le cerveau humain est à même d'interpréter et de compléter des images en 2D par des formes en 3D apprises au cours de l'enfance. Cette faculté permet à l'être humain de résoudre des ambiguïtés et de comprendre des phénomènes liés à la projection perspective en 3D. Cependant, dans le domaine de la vision par ordinateur, la plupart des approches portant sur la détection de classes d'objets proposent des solutions décrivant l'apparence et la géométrie de classes d'objets purement dans l'espace 2D de l'image.

Le domaine de la détection de classes d'objets et de l'estimation de pose en utilisant des approches d'apprentissage s'est développé dans les années 1990. Avec la disponibilité croissante de bases d'images, il est devenu de plus en plus intéressant d'analyser et d'organiser de manière automatique ces quantités de données visuelles. Les travaux dans ce domaine visent à l'identification et à la localisation de classes d'objets dans des images ; la notion d'une classe d'objets peut être basée sur la structure sémantique de la langue : une classe d'objets comporte tous les objets similaires par rapport à un concept sémantique. Dans cette thèse, nous nous limitons à trois classes d'objets exemplaires possédant des géométries rigides : les voitures, les motocycles et les bicyclettes. La détection de telles classes d'objets dans une image revient à sélectionner toutes les régions de l'image contenant au moins une partie d'un objet appartenant à l'une de ces classes. Les méthodes de détection de classes d'objets peuvent être évaluées et comparées sur des bases d'images en déterminant leur précision, c.à.d. le nombre de détections correctement attribuées à une classe, par rapport à leur rappel, c.à.d. le nombre de détections parmi tous les objets de cette classe contenus dans la base d'images.

D.2 Contexte

Historiquement, des approches ont été proposées qui décrivent une classe d'objets à travers son apparence globale, par exemple en calculant l'espace propre d'une distribution de couleurs caractéristique pour cette classe. D'autres approches globales modélisent les constellations entre des structures marquantes dans l'image afin de devenir plus invariantes par rapport à des points de vue différents et à la variabilité intra-classe. L'apparition de descripteurs locaux d'apparence a permis

de résoudre plusieurs limitations des approches globales : l'apparence d'un objet ou d'une classe d'objets est décrite de manière locale et l'ensemble de ces descriptions locales est regroupé sous forme d'histogrammes d'occurrence, sous forme de modèles de constellations ou à l'aide de sacs-de-caractéristiques, c.à.d. des représentations localement non ordonnées de descripteurs. Quant à l'estimation de poses, différentes méthodes ont été proposées dans la littérature. Des ensembles de points en 2D et en 3D peuvent être utilisés afin de déterminer les paramètres permettant de projeter les points en 3D sur les points en 2D de manière optimale. D'autres méthodes se servent de structures plus complexes comme des lignes ou des objets à géométrie paramétrique.

D.3 Contributions

Dans le cadre de cette thèse, nous visons la détection de classes d'objets et l'estimation de leurs poses à partir d'une seule image en utilisant des modèles 3D. Nous décrivons des étapes d'apprentissage, de détection et d'estimation adaptées à l'utilisation de données synthétiques pour lesquelles la géométrie est connue. En utilisant des modèles CAO existants et des méthodes de rendu issues du domaine de la synthèse d'images, nous proposons de créer des représentations en 3D de classes d'objets permettant de gérer simultanément des points de vue différents et la variabilité intra-classe.

Pour obtenir ces représentations en 3D, deux méthodes différentes sont proposées. La première utilise des données d'entraînement purement synthétiques alors que la seconde approche est basée sur un modèle de parties combinant des images d'entraînement réelles avec des données géométriques synthétiques. Pour l'entraînement de la méthode purement synthétique, nous proposons une procédure non-supervisée de filtrage de descripteurs locaux afin de rendre les descripteurs discriminatifs pour leur pose et leur classe d'objet. Dans le cadre du modèle de parties, l'apparence d'une classe d'objets est apprise de manière discriminative à partir d'une base de données annotée et la géométrie en 3D est apprise de manière générative à partir d'une base de modèles CAO.

Pendant la détection, nous introduisons tout d'abord une méthode de vote en 3D qui renforce la cohérence géométrique en se servant d'une estimation robuste de la pose. Ensuite, nous décrivons une deuxième méthode d'estimation de pose qui permet d'évaluer la probabilité de constellations de parties détectées en 2D en utilisant une géométrie 3D entière. Les deux méthodes génèrent des détections en 2D ainsi que des estimations approximatives de poses en 3D ; ces estimations approximatives sont ensuite améliorées en se servant d'un alignement de modèles 3D CAO avec des images en 2D, ce qui permet de résoudre des ambiguïtés, d'effectuer un filtrage des détections en 2D et de gérer des occultations.

L'approche est évaluée sur plusieurs bases d'images de référence et nous montrons qu'elle est capable de fournir des estimations de pose en 3D à partir d'images 2D tout en générant des détections en 2D comparables à l'état de l'art.

D.4 Résumé par Chapitre

D.4.1 Chapitre 2 : Données Synthétiques d'Entraînement

Dans ce chapitre, nous résumons le rendu de données synthétiques d'entraînement en utilisant des méthodes issues du domaine de la synthèse d'images. Nous justifions le choix de la représentation des modèles 3D CAO, le paramétrage de la caméra synthétique et le modèle d'éclairage. Dans les chapitres suivants, nous décrivons des approches pour la détection de classes d'objets génériques, l'estimation approximative de poses en 3D, l'identification d'objets particuliers et l'alignement précise ; ces approches exploitent les avantages associés avec les données 3D synthétiques afin d'élargir l´état de l'art au-delà des détections purement en 2D.

D.4.2 Chapitre 3 : Feature Maps

Dans ce chapitre, nous présentons une première approche pour la détection de classes d'objets basée sur des données synthétiques d'entraînement et nous décrivons les avantages de ces données lors de l'entraînement et de la détection. Nous décrivons une approche pour la détection de classes d'objets à partir de points de vue différents. Cette approche est à même de fournir des informations portant sur la pose 3D de l'objet détecté. Au lieu d'apprendre un modèle 3D à partir d'un nombre d'images 2D en utilisant des contraintes géométriques, nous nous servons d'une base de modèles 3D CAO texturés afin de construire une représentation robuste en 3D pour chaque catégorie d'objets, ce qui facilitera la détection à partir de points de vue différents. En utilisant une méthode de filtrage, les descripteurs locaux calculés à partir d'images synthétiques sont choisis en fonction de leur caractère approprié pour la mise en correspondance avec des descripteurs calculés à partir d'images réelles. Lors de la détection, la mise en correspondance permet de voter pour la catégorie d'objets et la pose la plus probable d'un objet détecté. Les votes les plus prometteurs sont ensuite évalués et améliorés en fonction de leur cohérence géométrique avec la projection du modèle 3D en se servant d'une estimation robuste de la pose. Nous présentons des résultats obtenus sur plusieurs bases d'évaluation et nous analysons la précision de l'estimation de la pose dans le cadre d'une séquence d'images calibrées. Les contributions principales de ce chapitre résident dans la procédure de sélection de descripteurs locaux contenant des informations géométriques 3D à partir de modèles 3D CAO et dans l'élargissement de la méthode traditionnelle de vote probabiliste à l'espace 3D. La méthode génère des hypothèses approximatives de poses en 3D qui sont ensuite améliorées par une estimation de pose complète. Cependant, le rappel relativement faible de cette méthode et la lenteur de l'estimation suggèrent une autre approche plus souple. Dans le chapitre suivant, nous proposons une telle approche qui permettra de résoudre plusieurs de ces faiblesses.

D.4.3 Chapitre 4 : Détection avec un Modèle Géométrique

Dans ce chapitre, nous décrivons une approche pour la détection de classes d'objets basée sur la combinaison de deux représentations différentes pour la géométrie et l'apparence. Nous combinons un modèle de parties pour la représentation de l'apparence construit de manière discriminative à partir d'images réelles et une représentation générative de la géométrie 3D apprise à partir

de modèles CAO. L'apprentissage de détecteurs discriminatifs en 2D pour la localisation de parties d'un objet peut s'effectuer de manière efficace et fiable à partir d'images réelles annotées, ce qui permet de réduire le nombre de modèles CAO requis pour l'apprentissage de la géométrie à quelques modèles non texturés, représentatifs pour la variation géométrique intra-classe. Aucune annotation manuelle de parties d'objets n'est nécessaire. En outre, une estimation probabiliste de la pose en 3D d'un objet permet d'obtenir sa pose 3D approximative en même temps que sa localisation en 2D. Cette estimation représente une manière efficace d'évaluer la cohérence de la détection de parties d'un objet en 2D par rapport à la géométrie 3D entière de la classe d'objets. Nous démontrons que deux représentations pour la géométrie et l'apparence peuvent être apprises séparément et combinées seulement lors de la détection, ce qui permet d'utiliser des sources de données d'entraînement séparées et des représentations simplifiées. Les contributions principales de ce chapitre résident dans une méthode permettant d'intégrer la géométrie 3D de modèles CAO dans la détection en 2D de parties d'objets, ce qui rend possible de déterminer la pose approximative en 3D d'un objet en même temps que sa localisation en 2D. Bien que cette approche soit moins précise que la méthode décrite dans le chapitre précédent, cette approche parvient à un rappel plus important sur des bases de données difficiles et bénéficie d'une procédure d'apprentissage plus souple. Les deux méthodes de détection de classes d'objets se sont servies de modèles 3D CAO comme source intégrale ou partielle de données d'entraînement afin de parvenir à une performance comparable à l'état de l'art tout en fournissant des estimations approximatives de poses en 3D. Dans le chapitre suivant, nous allons décrire une méthode d'alignement de modèles CAO avec des images réelles afin d'améliorer les estimations approximatives des deux approches précédentes.

D.4.4 Chapitre 5 : Estimation Précise de Pose par Alignement

Dans les deux chapitres précédents, nous avons proposé des méthodes pour la détection de classes d'objets dans des images tout en estimant leurs poses approximatives en 3D. Etant donné qu'un modèle générique décrivant la géométrie d'une classe d'objets doit prendre en compte des variations intra-classe importantes, la précision de ces estimations approximatives est limitée. Cependant, une meilleure précision serait souhaitable pour de nombreuses applications. Dans ce chapitre, nous introduisons une méthode permettant d'aligner un seul modèle 3D avec une image réelle, supposant qu'une initialisation de la pose a été fournie par l'une des deux approches précédentes. Nous illustrons une méthode d'alignement basée sur la génération de représentations synthétiques à partir d'un modèle CAO fourni et sur une mesure de similarité combinant l'alignement de contours avec une distance entropique d'apparence. Cette mesure combinée peut ensuite être optimisée de manière efficace afin de déterminer la pose de l'objet dans l'image avec davantage de précision. Nous proposons d'effectuer cet alignement en utilisant la méthode de "l'optimisation par essaim de particules", une méthode d'optimisation génétique. Nous analysons l'aptitude et la précision de cette méthode d'optimisation sur des exemples synthétiques et des images réelles et nous introduisons un critère de convergence basé sur l'énergie cinétique de l'essaim. Nous démontrons sa robustesse par rapport au bruit d'image et à l'occultation partielle et nous décrivons sa parallélisation efficace sur la carte graphique. Ce chapitre présente une mesure de similarité simple mais efficace, dérivée de l'alignement de contours et de l'estimation d'entropie conjointe qui est optimisée en utilisant une méthode génétique. L'approche effectue l'alignement d'un mo-

dèle CAO avec une image réelle et permet d'améliorer les estimations de pose fournies par les deux précédentes méthodes de détection de classes d'objets.

D.5 Perspectives

Cette thèse propose l'utilisation de modèles 3D CAO et de méthodes de rendu issues du domaine de la synthèse d'images afin de détecter des classes d'objets dans des images. Nous croyons que le potentiel de cette idée est loin d'être épuisé. Il est envisageable que les détections en 3D obtenues dans une image peuvent servir à améliorer la détection d'autres classes d'objets en se servant du contexte géométrique 3D de la scène et de la modélisation d'interactions entre différentes classes d'objets. En outre, l'information contenue dans les modèles CAO peut apporter d'avantage à l'apprentissage, notamment à travers les annotations sémantiques de parties ou la segmentation géométrique non supervisée de ces modèles. Dans le domaine de la synthèse d'images, l'animation d'objets déformables est facilitée par de nombreux logiciels, ce qui pourrait permettre l'apprentissage et la détection d'objets non rigides à partir du rendu synthétique. Enfin, le rendu synthétique d'images pourrait être une manière efficace d'évaluation systématique de l'aptitude d'une méthode de détection de classes d'objets pour son utilisation dans le cadre d'applications industrielles qui exigent une fiabilité supérieure.

Bibliography

[1] N. Allezard, M. Dhome, and F. Jurie. Recognition of 3D textured objects by mixing view-based and model-based representations. In *International Conference on Pattern Recognition*, 2000.

[2] M. Arie-Nachmison and R. Basri. Constructing implicit 3D shape models for pose estimation. In *International Conference on Computer Vision*, 2009.

[3] Y. Bao, M. Sun, and S. Savarese. Toward coherent object detection and scene layout understanding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[4] A. Bardera, M. Feixas, and I. Boada. Normalized similarity measures for medical image registration. *SPIE Medical Imaging*, 5370:108 – 118, 2004.

[5] H. Bay, T. Tuytelaars, and L. V. Gool. SURF: Speeded Up Robust Features. In *European Conference on Computer Vision*, 2006.

[6] J. F. Blinn. Models of light reflection for computer synthesized pictures. In *Conference on Computer Graphics and Interactive Techniques*, 1977.

[7] L. Breiman. Bagging predictors. *Machine Learning*, 24:123 – 140, 1996.

[8] O. Chum and A. Zisserman. An exemplar model for learning object classes. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[9] J. Clerc, M. Kennedy. The particle swarm - explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6:58 – 73, 2002.

[10] D. Crandall, P. Felzenszwalb, and D. Huttenlocher. Spatial priors for part-based recognition using statistical models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

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

[12] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[13] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39:1–38, 1977.

[14] N. Dowson and R. Bowden. Metric mixtures for mutual information (m3i) tracking. In *Proceedings of the 17th International Conference on Pattern Recognition*, pages 1051 – 4651, 2004.

[15] A. Edelman and H. Murakami. Polynomial roots from companion matrix eigenvalues. *Math. Comp.*, 64:763–776, 1995.

[16] W. Ensinger, C. Stahl, P. Knappe, K. Schertler, and J. Liebelt. Toward a robust 3D-model-based ground target classification system for airborne platforms. In *SPIE Conference on Defense, Security and Sensing*, 2010.

[17] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html, 2007.

[18] M. Everingham, A. Zisserman, C. K. I. Williams, and L. Van Gool. The PASCAL Visual Object Classes Challenge 2006 (VOC2006) results. http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf, 2006.

[19] R.-E. Fan, P.-H. Chen, and C.-J. Lin. Working set selection using second order information for SVM training. *Journal of Machine Learning Research*, 6:1889–1918, 2005.

[20] Z.-G. Fan and B.-L. Lu. Fast recognition of multi-view faces with feature selection. In *International Conference on Computer Vision*, 2005.

[21] P. Felzenszwalb, R. Girshick, and D. McAllester. Cascade object detection with deformable part models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[22] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[23] M. A. Fischler and R. C. Bolles. Random sample consensus. *Communications of the ACM*, 24:381–395, 1981.

[24] M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, 22:67–92, 1973.

[25] P. Flynn and A. Jain. CAD-based computer vision: from CAD models to relational graphs. *Transactions on Pattern Analysis and Machine Intelligence*, 13:114 – 132, 1991.

[26] D. Forsyth, J. L. Mundy, A. Zisserman, C. Coelho, A. Heller, and C. Rothwell. Invariant descriptors for 3D object recognition and pose. *Transactions on Pattern Analysis and Machine Intelligence*, 13:971–991, 1991.

[27] J.-M. Frahm, M. Pollefeys, and M. Shah. CVGPU: Workshop computer vision on GPUs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[28] M. Fritz and B. Schiele. Towards unsupervised discovery of visual categories. In *DAGM Symp. on Pattern Recog.*, 2006.

[29] G. Gill and M. Levine. Multi-view object detection based on spatial consistency in a low dimensional space. In *Symposium of the German Association for Pattern Recognition (DAGM)*, 2009.

[30] S. Gold, A. Rangarajan, C.-P. Lu, and E. Mjolsness. New algorithms for 2D and 3D point matching: Pose estimation and correspondence. *Pattern Recognition*, 31:957–964, 1997.

[31] A. Golovinskiy and T. Funkhouser. Consistent segmentation of 3D models. In *Shape Modeling International*, 2009.

[32] K. Grauman and T. Darrell. The pyramid match kernel: Efficient learning with sets of features. *Journal of Machine Learning Research*, 8:725–760,, 2007.

[33] R. L. Gregory. *The Intelligent Eye.* McGraw-Hill, 1970.

[34] W. Grimson, T. Lozano-Perez, and D. Huttenlocher. *Object Recognition by Computer: The Role of Geometric Constraints*. MIT Press, 1990.

[35] C. Hansen and T. Henderson. *CAD Based Programming for Sensory Robots*, chapter CAD-based computer vision: The automatic generation of recognition strategies, pages 275 – 298. Springer-Verlag, 1988.

[36] R. M. Haralick, H. Joo, C.-N. Lee, X. Zhuang, V. G. Vaidya, and M. B. Kim. Pose estimation from corresponding point data. *IEEE Transactions on Systems, Man and Cybernetics*, 19:1426–1446, 1989.

[37] R. M. Haralick, C.-N. Lee, K. Ottenberg, and M. Noelle. Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, 13:331 – 356, 1994.

[38] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, 1988.

[39] B. Heisele, G. Kim, and A. J. Meyer. Object recognition with 3D models. In *British Machine Vision Conference*, 2009.

[40] T. C. Henderson, C. Hansen, A. Samal, C. Ho, and B. Bhanu. CAD based 3D visual recognition. In *International Conference on Pattern Recognition*, 1986.

[41] D. Hoiem, A. Efros, and M. Hebert. Putting objects into perspective. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[42] D. Hoiem, C. Rother, and J. Winn. 3D LayoutCRF for multi-view object class recognition and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[43] H. Hoppe. Progressive meshes. *Computer Graphics*, 30:99–108, 1996.

[44] B. K. P. Horn. Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America A*, 5:1127 – 1135, 1987.

[45] J. Kennedy. Probability and dynamics in the particle swarm. In *Congress on Evolutionary Computation*, volume 1, pages 340 – 347, 2004.

[46] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proc. IEEE International Conference on Neural Networks*, volume IV, pages 1942 – 1948, 1995.

[47] H. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

[48] A. Kushal, C. Schmid, and J. Ponce. Flexible object models for category-level 3D object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[49] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[50] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *European Conference on Computer Vision Workshop on Statistical Learning*, 2004.

[51] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *Transactions on Pattern Analysis and Machine Intelligence*, 28:1465 – 1479, 2006.

[52] J. Liebelt and K. Schertler. Precise registration of 3D models to images by swarming particles. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[53] J. Liebelt, K. Schertler, and F. Schubert. Image processing device, 2010. Patent no. DE102008014381, accepted.

[54] J. Liebelt and C. Schmid. Multi-view object class detection with a 3D geometric model. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[55] J. Liebelt, C. Schmid, and K. Schertler. Viewpoint-independent object class detection using 3D feature maps. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[56] J. Liebelt, J. Xiao, and J. Yang. Robust AAM fitting by fusion of images and disparity data. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[57] D. G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31:355 – 395, 1987.

[58] D. G. Lowe. Robust model-based motion tracking through the integration of search and estimation. *International Journal of Computer Vision*, 8:113 – 122, 1992.

[59] D. G. Lowe. Local feature view clustering for 3D object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[60] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal on Computer Vision*, 60:91 – 110, 2004.

[61] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[62] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens. Multimodality image registration by maximization of mutual information. *IEEE Transactions on Medical Imaging*, 16:187 – 198, 1997.

[63] M. Marszalek and C. Schmid. Spatial weighting for bag-of-features. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[64] M. Marszalek and C. Schmid. Semantic hierarchies for visual object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[65] J. Matas, O. Chum, U. Martin, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *British Machine Vision Conference*, 2002.

[66] J. Meyer-Hilberg. Pirdis: A new versatile tool for SAR/MTI systems simulation. In *Sixth European Conference on Synthetic Aperture Radar (EUSAR2006)*, 2006.

[67] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *Transactions on Pattern Analysis and Machine Intelligence*, 27:1615 – 1630, 2005.

[68] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65:43–72, 2005.

[69] G. A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38:39–41, 1995.

[70] H. P. Moravec. Towards automatic visual obstacle avoidance. In *International Joint Conference on Artificial Intelligence*, 1977.

[71] J. L. Mundy and C.-F. Chang. Fusion of intensity, texture, and color in video tracking based on mutual information. In *IEEE Proceedings of the 33rd Applied Imagery Pattern Recognition Workshop*, 2004.

[72] J. L. Mundy, A. Liu, N. Pillow, A. Zisserman, S. Abdallah, S. Utcke, S. K. Nayar, and C. Rothwell. An experimental comparison of appearance and geometric model based recognition. In *Object Representation in Computer Vision*, 1996.

[73] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society of Industrial and Applied Mathematics*, 5(1):32–38, 1957.

[74] H. Murase and S. K. Nayar. Visual learning and recognition of 3D objects from appearance. *International Journal of Computer Vision*, 14:5–24, 1995.

[75] H. Najafi, Y. Genc, and N. Navab. Fusion of 3D and appearance models for fast object detection and pose estimation. In *Asian Conference on Computer Vision*, 2006.

[76] N. Navab and O. Faugeras. Monocular pose determination from lines: critical sets and maximum number of solutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1993.

[77] J. Ng and S. Gong. Multi-view face detection and pose estimation using a composite SVM across the view sphere. In *International Workshop on Recogniton, Analysis and Tracking of Faces*, 1999.

[78] A. Ngan, F. Durand, and W. Matusik. Experimental analysis of BRDF models. In *Eurographics Symposium on Rendering*, 2005.

[79] C. F. Olson. Probabilistic indexing for object recognition. *Transactions on Pattern Analysis and Machine Intelligence*, 17:518 – 522, 1995.

[80] A. Opelt and A. Pinz. Object localization with boosting and weak supervision for generic object recognition. In *14th Scandinavian Conference on Image Analysis*, 2005.

[81] S.-H. Or, W. S. Luk, K. H. Wong, and I. King. An efficient iterative pose estimation algorithm. In *Asian Conference on Computer Vision*, 1998.

[82] J. Owens, D. Luebke, N. Govindaraju, M. Harris, J. Krüger, A. Lefohn, and T. Purcell. A survey of general-purpose computation on graphics hardware. *Eurographics State of the Art Reports*, 2005.

[83] M. Ozuysal, V. Lepetit, and P.Fua. Pose estimation for category specific multiview object localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[84] C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38:15–33, 2000.

[85] T. Phong, R. Horaud, A. Yassine, and D. Pham. Optimal estimation of object pose from a single perspective view. In *IEEE International Conference on Computer Vision*, 1993.

[86] J. C. Platt. *Advances in kernel methods: support vector learning*, chapter Fast training of support vector machines using sequential minimal optimization, pages 185 – 208. MIT Press, 1999.

[87] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever. Image registration by maximization of combined mutual information and gradient information. *IEEE Transactions on Medical Imaging*, 19(8):1 – 6, 2000.

[88] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever. Mutual-information-based registration of medical images: A survey. *IEEE Transactions on Medical Imaging*, 22(8):986 – 1004, 2003.

[89] J. Ponce, T. L. Berg, M. Everingham, D. A. Forsyth, M. Hebert, S. Lazebnik, M. Marszalek, C. Schmid, B. C. Russell, A. Torralba, C. K. I. Williams, J. Zhang, and A. Zisserman. *Toward Category-Level Object Recognition*, chapter Dataset Issues in Object Recognition. Lecture Notes in Computer Science. Springer-Verlag, 2006.

[90] H.-K. Pong and T.-J. Cham. Alignment of 3D models to images using region-based mutual information and neighborhood extended gaussian images. In *Proceedings of the Asian Conference for Computer Vision*, page 60 – 69, 2006.

[91] M. Pontil and A. Verri. Support vector machines for 3D object recognition. *Transactions on Pattern Analysis and Machine Intelligence*, 20:637–646, 1998.

[92] T. Quack, V. Ferrari, B. Leibe, and L. V. Gool. Efficient mining of frequent and distinctive feature configurations. In *IEEE International Conference on Computer Vision*, 2007.

[93] V. Raghavan, P. Bollmann, and G. S. Jung. A critical investigation of recall and precision as measures of retrieval system performance. *ACM Transactions on Information Systems*, 7:205 – 229, 1989.

[94] J. Rissanen. A universal prior for integers and estimation by minimum description length. *Annals of Statistics*, 11:417 – 431, 1983.

[95] G. Rohde, S. Pajevic, and C. Pierpaoli. Multi-channel registration of diffusion tensor images using directional information. In *IEEE International Symposium on Biomedical Imaging: Macro to Nano*, volume 1, pages 712 – 715, 2004.

[96] B. Rosenhahn. *Pose Estimation Revisited*. PhD thesis, Christian-Albrechts-Universitaet Kiel, 2003.

[97] B. Rosenhahn, T. Brox, U. Kersting, D. Smith, J. Gurney, and R. Klette. A system for marker-less human motion estimation. *Kuenstliche Intelligen*, 1:45–51, 2006.

[98] 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. *International Journal of Computer Vision*, 66:231 – 259, 2006.

[99] D. Rueckert, M. Clarkson, D. Hill, and D. Hawkes. Non-rigid registration using higher-order mutual information. In *Medical Imaging 2000: Image Processing*, volume 3979 of *SPIE Proceedings*, pages 438 – 447, 2000.

[100] D. Russakoff, C. Tomasi, T. Rohlfing, and C. Maurer. Image similarity using mutual information of regions. In *Proceedings of the European Conference on Computer Vision*, pages 596 – 607, 2004.

[101] S. Savarese and L. Fei-Fei. 3D generic object categorization, localization and pose estimation. In *IEEE International Conference on Computer Vision*, 2007.

[102] S. Savarese and L. Fei-Fei. View synthesis for recognizing unseen poses of object classes. In *European Conference on Computer Vision*, 2008.

[103] K. Schertler and J. Liebelt. Automatable reconstruction method, 2010. Patent no. DE200810057176, pending.

[104] K. Schertler and J. Liebelt. Learning apparatus for an object detection system, 2010. Patent no. DE200810057979, pending.

[105] B. Schiele and J. L. Crowley. Recognition without correspondence using multidimensional receptive field histograms. *International Journal of Computer Vision*, 36:31–52, 2000.

[106] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *Transactions on Pattern Analysis and Machine Intelligence*, 19:530–534, 1997.

[107] H. Schneiderman and T. Kanade. Object detection using the statistics of parts. *International Journal of Computer Vision*, 56:151–177, 2004.

[108] B. Schoelkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, 2002.

[109] Y. Shi and R. Eberhart. Empirical study of particle swarm optimization. In *Congress on Evolutionary Computing*, volume III, pages 1945 – 1950, 1999.

[110] P. Shirley and S. Marschner. *Fundamentals of Computer Graphics*. A K Peters, 2009.

[111] B. Siddiquie and A. Gupta. Beyond active noun tagging: Modeling contextual interactions for multi-class active learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[112] G. Stockman. Object recognition and localization via pose clustering. *Computer Vision Graphics and Image Processing*, 40:361 – 387, 1987.

[113] C. Studholme, D. Hill, and D. Hawkes. Incorporating connected region labelling into automated image registration using mutual information. In *Proceedings of the IEEE workshop on Mathematical Methods in Biomedical Image Analysis*, pages 23 – 31, 1996.

[114] H. Su, M. Sun, L. Fei-Fei, and S. Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *IEEE International Conference on Computer Vision*, 2009.

[115] K.-K. Sung and T. Poggio. Example-based learning for view-based human face detection. *Transactions on Pattern Analysis and Machine Intelligence*, 20:39–51, 1998.

[116] I. Suveg and G. Vosselman. Mutual information based evaluation of 3D building models. In *IEEE International Conference Pattern Recognition*, volume 3, pages 557 – 560, 2002.

[117] A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. V. Gool. Towards multi-view object class detection. In *Conf. on Computer Vision and Pattern Recognition*, 2006.

[118] E. Tola, V. Lepetit, and P. Fua. DAISY: An efficient dense descriptor applied to wide baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009.

[119] A. Toshev, A. Makadia, and K. Daniilidis. Shape-based object recognition in videos using 3D synthetic object model. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[120] M. Turk and A. Pentland. Face recognition using eigenfaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1991.

[121] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:376 – 380, 1991.

[122] M. Unser and P. Thévenaz. Stochastic sampling for computing the mutual information of two images. In *Proceedings of the Fifth International Workshop on Sampling Theory and Applications (SampTA'03)*, pages 102 – 109, May 2003.

[123] P. Viola and M. Jones. Robust real-time object detection. In *International Journal of Computer Vision*, 2001.

[124] P. Viola and W. M.Wells. Alignment by maximization of mutual information. *International Journal of Computer Vision*, 24(2):137 – 154, 1997.

[125] P. A. Viola. *Alignment by Maximization of Mutual Information*. Technical report no. 1548, MIT, AI Lab, 1995.

[126] S. Watt. *Introduction: The Theory of Forms (Plato: Republic, Books 5-7)*. Wordsworth Editions, 1997.

[127] M. Weber, W. Einhaeuser, M. Welling, and P. Perona. Viewpoint-invariant learning and detection of human heads. In *International Conference on Automatic Face and Gesture Recognition*, 2000.

[128] A. E. Welchman, A. Deubelius, V. Conrad, H. H. Bülthoff, and Z. Kourtzi. 3D shape perception from combined depth cues in human visual cortex. *Nature Neuroscienc*, 8:820 – 827, 2005.

[129] W. Wolfe, D. Mathis, C. Sklair, and M. Magee. The perspective view of three points. *IEEE Transactions on Pattern Analysis and Machine Intelligence,*, 13:66 – 73, 1991.

[130] H. Wolfson and I. Rigoutsos. Geometric hashing: An overview. *Comp. Science and Engineering*, 4:10 – 21, 1997.

[131] J. Wu. *Rotation Invariant Classification of 3D Surface Texture Using Photometric Stereo*. PhD thesis, School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, 2002.

[132] J. Xie, M. Hu, and M. Shah. Unfolding warping for object recognition. In *International Conference on Pattern Recognition*, 2008.

[133] S. Yamany and A. Farag. Free-form surface registration using surface signatures. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1098 – 1104, 1999.

[134] P. Yan, S. M. Khan, and M. Shah. 3D model based object class detection in an arbitrary view. In *International Conference on Computer Vision*, 2007.

[135] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *International Conference on Computer Vision*, 1999.