Using Multi-view Recognition and Meta-data Annotation to Guide a Robot's Attention

Alexander Thomas
KU Leuven, Belgium

Vittorio Ferrari
ETH Zürich, Switzerland

Bastian Leibe
RWTH Aachen, Germany

Tinne Tuytelaars
KU Leuven, Belgium

Luc Van Gool
ETH Zürich, Switzerland, and KU Leuven, Belgium

June 10, 2009

Abstract

In the transition from industrial to service robotics, robots will have to deal with increasingly unpredictable and variable environments. We present a system that is able to recognize objects of a certain class in an image and to identify their parts for potential interactions. The method can recognize objects from arbitrary viewpoints and generalizes to instances that have never been observed during training, even if they are partially occluded and appear against cluttered backgrounds. Our approach builds on the Implicit Shape Model of Leibe et al. (2008). We extend it to couple recognition to the provision of meta-data useful for a task and to the case of multiple viewpoints by integrating it with the dense multi-view correspondence finder of Ferrari et al. (2006). Meta-data can be part labels but also depth estimates, information on material types, or any other pixelwise annotation. We present experimental results on wheelchairs, cars, and motorbikes.

Keywords: object class recognition, computer vision

1 Introduction

People can very quickly understand scenes and assess situations. In particular, we can deal with the substantial variability which we are bound to be confronted with in our daily lives. The human ability to recognize object classes and their functional parts is a vital component in this. If a new type of car hits the market, we immediately recognize it as yet another car, without needing any kind of extra training. Lots of qualitative information can be derived through mechanisms of generalization, based on previous exposure to other members of the same object class. Coming back to the issue of functional parts, their relative positions tend to be quite similar indeed and we won't be hard-pressed to identify them. Similarly, we can judge 3D shape from a single image, not very precisely but at a qualitative level. This is often enough to allow interaction with an object, possibly in an iterative way. Qualitative information can serve as a starting point to obtain more accurate data if needed.

This *qualitative rather than quantitative* type of scene analysis is a natural outcome of our need to interact with the world at a high semantic level. A need for prior, quantitatively precise models of the surroundings would put heavy constraints on the applicability and robustness of a system. An increasing number of robotic applications call for similar, semantic capabilities. Yet, much of robotics so far has been geared towards navigation in precisely modeled worlds and interactions with precisely modeled objects. Object class recognition or fast matching of information (e.g. images) against massive datasets to find topological similarities was not possible before. But that is rapidly changing now.

Let us take visual navigation as a case in point for qualitative scene analysis. People are known to mainly determine their trajectory *relative* to landmarks. Robots, on the other hand, are typically programmed to navigate via precisely defined paths, calculated in *absolute* terms,

based on a precise 3D world model. New results on very fast comparisons of images taken by a mobile platform against masses of reference images, can provide for the aforementioned relative trajectory planning. Indeed, using such technologies, the first such implementations for robot navigation have already been published (Goedemé et al., 2004, 2007; Fraundorfer et al., 2007; Segvic et al., 2007). Object class recognition from 2D images still has not quite put its mark onto robotics to the same degree, but can be expected to have an even bigger impact. Efforts to classify parts of scenes as trees, buildings, etc. from mobile platforms have been made, but by taking 3D point clouds as input, most of this work is still very much grounded in the quantitative line of thinking (Pantofaru et al., 2003; Munoz et al., 2008; Brostow et al., 2008). Even though visual information is gaining interest (Posner et al., 2007), it is mostly used only to augment the point clouds. Meger et al. (2008) do use visual information in their Curious George platform to augment online scene mapping with semantically useful information, i.e. the presence of specific objects. It would be interesting to extend their approach to object classes and enable interaction with the objects.

With this paper, we contribute to this general shift towards more qualitative, but semantically enriched information. Our proposed approach recognizes object classes from single images, regardless of their viewpoint. As an integral component, our approach detects individual, semantically meaningful object parts and crudely localizes them on the object. This should allow a robot to approach these parts in order to engage in an interaction with the objects. The experiments show results for object classes like cars, wheelchairs, and motorbikes. In terms of applications, an automated carwash station could better adapt to the particular car at hand (Figure 1), or a service robot could approach a wheelchair, grasp it at the handles, and bring it to a specified location (Figure 13). Moreover, we demonstrate that expectations about object classes allow for the estimation of the overall 3D shape of members of the same class. All this works for class members that have never been seen before, and from single images.

The presented work builds on earlier object class recognition work. In particular, we use the Implicit Shape Model approach of Leibe and Schiele (Leibe et al., 2008), which is briefly reviewed in Section 3. It models new instances of an object category as a jigsaw puzzle of parts

Figure 1: *Humans can quickly analyze a scene from a single image. Recognizing subparts of an object helps to recognize the object as a whole, but recognizing the object in turn helps to gather more detailed information about its subparts. Knowledge about these parts can then be used to guide actions. For instance, in the context of a car wash, a decomposition of the car in its subparts can be used to apply optimized washing methods to the different parts. This figure shows such decomposition obtained with our system.*

from the training instances. A codebook of typical appearances is constructed from interest points, and their occurrences on the training images are recorded, allowing to detect novel objects by means of generalized Hough voting. As already argued, dealing with higher intra-class variability implies that robots can no longer rely on rigid, predefined 3D transformations to interact with those objects. Instead, we propose a meta-data transfer method which helps a robot to localize the relevant part for interaction based on the actual image observations, rather than relying on a fixed rigid 3D structure. But other types of meta-data can be handled as well, such as crude 3D shape and surface orientation. As will be described, our meta-data transfer is tightly interwoven with the object class recognition procedure itself. We attach meta-data to the votes cast in the Hough space. By collecting the votes that contributed to an object hypothesis, we can combine the meta-data fragments into an annotation for the recognized object. Moreover, we also extend the recognition procedure to handle multiple viewpoints. Instead of running a separate detector for each view, we establish and exploit relations between the views. Indeed, traditional object class recognition methods, including the Implicit Shape Model, work with a preferred viewpoint only (e.g. frontal faces or cars seen from the side). The same multi-viewpoint capabilities are inherited by our meta-

data transfer.

In this paper, we highlight the use of object class recognition and meta-data transfer for tasks that would require object-robot interactions. The impact of these procedures can be expected to be much larger, however. Even 'low-level' processes like motion extraction or ground plane determination can benefit greatly. As a matter of fact, this is a major breakthrough that we can expect to happen over the coming years. When high-level, semantic information like object class membership (a car) or material type (a windshield) can be fed back into lower levels, these can function more reliably. Cars tend to move on the ground plane in specific ways, and windshields are smooth and made of glass, therefore shiny, and stereo information obtained there can better be discarded. As the performance of the lower levels improves because of this feedback from higher levels, they can then also support these higher levels more effectively. One gets processing loops that are closed over semantic levels. These observations are mirrored by neurophysiological findings (Mumford, 1994; Rockland and Hoesen, 1994). In the brain, 'low-level' areas do not only feed into the 'high-level' ones, but invariably the latter channel their output into the former. The resulting feedback loops over the semantic level are key for successful scene understanding. The brain seems keen to bring all levels into unison, from basic perception up to cognition. It relies on these *cognitive loops* for this to happen.

The paper is organized as follows. After discussion of related work (Section 2), we recapitulate the Implicit Shape Model of Leibe et al. (2008) for simultaneous object recognition and segmentation (Section 3). Then follows the first contribution of this paper, as we explain how we transfer meta-data from training images to a previously unseen image (Section 4) for both discrete and real-valued meta-data. Next, as the second contribution, we show how to efficiently extend the recognition and annotation procedure to the multi-view case (Section 5) by integrating it with Ferrari et al. (2006). We demonstrate the viability of our approach by transferring object part labels for wheelchairs, cars and motorbikes, as well as depth maps and surface orientations for cars (Section 6). Section 7 concludes the paper.

2 Related Work

The first examples of cognitive feedback in vision have already been implemented. Hoiem et al. (2006) and Cornelis et al. (2006) proposed frameworks which embed the separate mechanisms of object detection and scene geometry estimation into a cognitive loop. Objects can be detected more reliably and false-positive detections in improbable locations (e.g. people on trees) are filtered out based on the automatically estimated geometry of the scene. In turn, object detections allow to improve scene geometry estimation. In Leibe et al. (2007), a similar idea is applied to images taken from a moving vehicle, using car and pedestrian detections to improve ground-plane and scene depth estimation in a city environment. However, these systems only couple recognition and crude 3D scene information (the position of the groundplane). Here we set out to demonstrate the wider applicability of cognitive feedback, by inferring 'meta-data' such as material characteristics, the location and extent of object parts, or even 3D object shape, based on object class recognition. Given a set of annotated training images of a particular object class, we transfer these annotations to new images containing previously unseen object instances of the same class.

A general framework that allows such inference is the work on *image analogies*, where a mapping between two given images A and A' is transferred to an image B to get an 'analogous' image B'. As shown in work by Hertzmann et al. (2001) and Cheng et al. (2008), mappings can include texture synthesis, superresolution and image transformations like blurring and artistic filters. Most closely related to our work is the mapping that is called 'texture-by-numbers', where A is a parts annotation of a textured image A'. This allows to generate a plausible textured image from a new annotation B. Even though no example is shown in the cited works, it should be possible to do the inverse mapping, i.e. annotate an unseen image. However, the image analogies framework is limited to local image statistics, and does not involve a deeper understanding of the structure of the image.

Related to image analogies is SIFT Flow by Liu et al. (2008), where the best matching image in a database of training images is warped to match the structure of a query image. If the training images are annotated with a type of meta-data (e.g. motion vectors), an annotation for the

query image can be inferred. The success of the method depends on the presence of a sufficiently similar image in the training set. It cannot integrate information from multiple images to annotate a given query image.

Other approaches focus on inferring 3D shape from single images. Hoiem et al. (2005) estimate the coarse geometric properties of a scene by learning appearance-based models of surfaces at various orientations. In the same vein, Saxena et al. (2005) are able to reconstruct coarse depth maps from a single image of an entire scene by means of a Markov Random Field. Both these methods focus purely on geometry estimation, without incorporating an object recognition process. Like image analogies, they rely solely on the statistics of small image patches. There are methods which focus on more detailed 3D shape estimation of separate objects from a monocular image, like Han and Zhu (2003). Their method uses graph representations for both the geometry of the objects and their relations to the scene. To extract the graph representation from the image and estimate the geometry, a sketch representation of the objects is generated. This limits the method to objects that can be represented by a set of lines or that have prominent edges, like trees or polyhedra. Hassner and Basri (2006) infer 3D shape of an object in a single image from known 3D shapes of other members of the object's class. Their method is specific to 3D meta-data though, and the object is assumed to be recognized and segmented beforehand. Their analysis is not integrated with the detection and recognition of the objects, as is ours.

Recently, there has been a growing interest in extending object category recognition to the multi-view case, mirroring a similar evolution in the older field of specific object detection (e.g. Rothganger et al. (2006); Ferrari et al. (2006)). Aside from Thomas et al. (2006) which will form the basis of the multi-view recognition and annotation extension in this paper (Section 5), other approaches have been proposed to handle multi-view object class recognition. Hoiem et al. (2007) have extended their Layout Conditional Random Field framework to input a 3D model. They demonstrate recognition on cars from multiple viewpoints. Although the aspect of meta-data is not explored, their method could potentially be applied to estimate 3D-related meta-data like a depth map for the recognized object. Other methods for viewpoint-independent object recognition have been proposed as well, e.g. based on Partial Surface Models (Kushal et al., 2007), canonical object parts (Savarese and Fei-Fei, 2007), a rigid 3D model with features from different object instances attached to it (Yan et al., 2007) and a set of CAD models (Liebelt et al., 2008). However, all of these focus on recognition only and are not suited for deriving meta-data, in that there is no obvious mechanism for adapting the metadata to the appearance of the new object instance.

Preliminary versions of the two main components of this work appeared in Thomas et al. (2008, 2009) (meta-data transfer) and Thomas et al. (2006) (multi-view). This paper for the first time discusses their full integration into a single system and shows experimental results obtained with this integrated method.

3 Object Class Detection with an Implicit Shape Model

In this section we briefly summarize the *Implicit Shape Model* (ISM) approach proposed by Leibe et al. (2008), which we use as the object class detection technique at the basis of our approach (see also Figure 2).

Given a training set containing images of several instances of a certain category (e.g. side views of cars) as well as their segmentations, the ISM approach builds a model that generalizes over intra-class variability and scale. The modeling stage constructs a codebook of local appearances, i.e. of local structures that occur repeatedly across the training images. Codebook entries are obtained by clustering image features sampled at interest point locations. Instead of searching for correspondences between a novel test image and model views, the ISM approach maps sampled image features onto this codebook representation. We refer to all features in every training image that are mapped to a single codebook entry as *occurrences* of that entry. The spatial intra-class variability is captured by modeling spatial occurrence distributions for each codebook entry. Those distributions are estimated by recording all locations of codebook entry occurrences, relative to the object centers (which are given as training annotation). Together with each occurrence, the approach stores a local segmentation mask, which is later used to infer top-down segmentations.

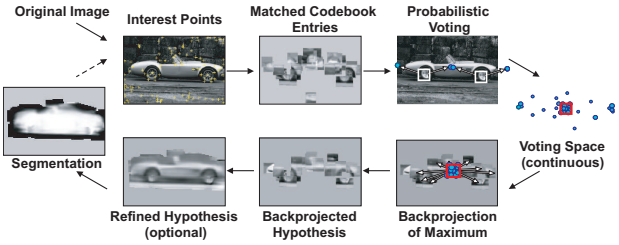

Figure 2: *The recognition procedure of the ISM system.*

3.1 ISM Recognition

The ISM recognition procedure is formulated as a probabilistic extension of the Hough transform (Leibe et al., 2008). Let e be an image patch observed at location $\boldsymbol{\ell}$. The probability that e matches to codebook entry c_i can be expressed as $p(c_i|e)$. Patches and codebook entries are represented by feature descriptors. In our implementation, two descriptors match if their distance or similarity (Euclidean or correlation, depending on the descriptor type), respectively, is below or exceeds a fixed threshold. Each matched codebook entry c_i casts votes for instances of the object category o_n at different locations and scales $\boldsymbol{\lambda} = (\lambda_x, \lambda_y, \lambda_s)$ according to its spatial occurrence distribution $P(o_n, \boldsymbol{\lambda}|c_i, \boldsymbol{\ell})$. The votes are weighted by $P(o_n, \boldsymbol{\lambda}|c_i, \boldsymbol{\ell})p(c_i|e)$, and the total contribution of a patch to an object hypothesis $(o_n, \boldsymbol{\lambda})$ is expressed by the following marginalization:

$$p(o_n, \boldsymbol{\lambda}|e, \boldsymbol{\ell}) \quad = \quad \sum_i p(o_n, \boldsymbol{\lambda}|c_i, \boldsymbol{\ell})p(c_i|e) \qquad (1)$$

where the summation is over all entries c_i in the codebook. The votes are collected in a continuous 3D voting space (translation and scale). Maxima are found using Mean Shift Mode Estimation with a kernel K with scale-adaptive bandwidth h and a uniform profile (Cheng, 1995; Leibe and Schiele, 2005):

$$\hat{p}(o_n, \boldsymbol{\lambda}) = \frac{1}{h(\boldsymbol{\lambda})^3} \sum_k \sum_j p(o_n, \boldsymbol{\lambda}_j|e_k, \boldsymbol{\ell}_k) K\left(\frac{\boldsymbol{\lambda} - \boldsymbol{\lambda}_j}{h(\boldsymbol{\lambda})}\right) \qquad (2)$$

In this equation, $\boldsymbol{\lambda}_j$ are the locations of the votes, stemming from image patches e_k. Each local maximum in this voting space yields an hypothesis for an object instance at a certain location and scale in the image.

3.2 Top-Down Segmentation

After the voting stage, the ISM approach computes a probabilistic top-down segmentation for each hypothesis, in order to determine its spatial support in the image. This is achieved by backprojecting to the image the votes contributing to the hypothesis and using the stored local segmentation masks to infer the probability that each pixel $\boldsymbol{p}$ is *figure* or *ground* given the hypothesis at location $\boldsymbol{\lambda}$ (Leibe et al., 2008). More precisely, the *figure* probablity for $\boldsymbol{p}$ is only affected by codebook entries c_i that match to a patch e containing $\boldsymbol{p}$, and only by their occurrences that contribute to the hypothesis at location $\boldsymbol{\lambda}$. The probability is calculated as a weighted average over the corresponding pixels in these occurrences' segmentation masks. The weights correspond to the contribution of each occurrence to the hypothesis:

$$\begin{aligned} & p(\boldsymbol{p} = figure|o_n, \boldsymbol{\lambda}) \\ & = \sum_{e:\boldsymbol{p}\in e} \sum_{c_i} p(\boldsymbol{p} = figure|e, c_i, o_n, \boldsymbol{\lambda}) p(e, c_i|o_n, \boldsymbol{\lambda}) \\ & = \sum_{e:\boldsymbol{p}\in e} \sum_{c_i} p(\boldsymbol{p} = figure|c_i, o_n, \boldsymbol{\lambda}) \frac{p(o_n, \boldsymbol{\lambda}|c_i)p(c_i|e)p(e)}{p(o_n, \boldsymbol{\lambda})} \end{aligned} \qquad (3)$$

We underline here that a separate local segmentation mask is kept for every occurrence of each codebook entry. Different occurrences of the same codebook entry in a test image will thus contribute different local segmentations, based on their relative location with respect to the hypothesized object center.

In early versions of their work, Leibe et al. (2008) included an optional processing step, which refines the hypothesis by a guided search for additional matches (Figure 2). This improves the quality of the segmentations, but at a high computational cost. Uniform sampling was used in Leibe and Schiele (2003), which became untractable once scale-invariance was later introduced into the system. Instead, in this paper we propose a more efficient refinement algorithm (Section 4.3).

3.3 MDL Verification

In a last processing stage of the ISM system, the computed segmentations are exploited to refine the object detection

scores, by taking only *figure* pixels into account. Moreover, this last stage also disambiguates overlapping hypotheses. This is done by a hypothesis verification stage based on Minimum Description Length (MDL), which searches for the combination of hypotheses that together best explain the image. This step prevents the same local image structure to be assigned to multiple detections (e.g. a wheel-like image patch cannot belong to multiple cars). For details, we again refer to Leibe et al. (2008).

4 Transferring Meta-data

The power of the ISM approach lies in its ability to recognize novel object instances as approximate jigsaw puzzles built out of pieces from different training instances. In this paper, we follow the same spirit to achieve the new functionality of transferring meta-data to new test images.

Example meta-data is provided as annotations to the training images. Notice how segmentation masks can be considered as a special case of meta-data. Hence, we transfer meta-data with a mechanism inspired by that used above to segment objects in test images. The training meta-data annotations are attached to the occurrences of codebook entries, and are transferred to a test image along with each matched feature that contributed to an hypothesis (Figure 3). This strategy allows us to generate novel annotations tailored to the new test image, while explicitly accommodating for the intra-class variability.

Unlike segmentations, which are always binary, meta-data annotations can be either binary (e.g. for delineating a particular object part or material type), discrete multi-valued (e.g. for identifying *all* object parts), real-valued (e.g. depth values), or even vector-valued (e.g. surface orientations). We first explain how to transfer discrete meta-data (Section 4.1), and then extend the method to the real- and vector-valued cases (Section 4.2).

4.1 Transferring Discrete Meta-data

In case of discrete meta-data, the goal is to assign to each pixel $\boldsymbol{p}$ of the detected object a label $a \in \{a_j\}_{j=1:N}$. We first compute the probability $p(\boldsymbol{p} = a_j)$ for each label a_j separately. This is achieved by extending eq. (3) for $p(\boldsymbol{p} = figure)$ to the more general case of discrete meta-data:

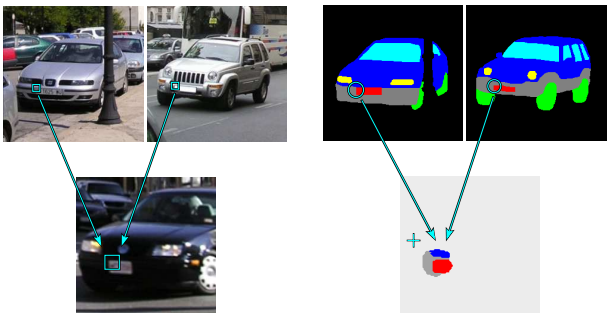

Figure 3: *Transferring (discrete) meta-data. Left: two training images and a test image. Right: the annotations for the training images, and the partial output annotation. The corner of the license plate matches with a codebook entry which has occurrences on similar locations in the training images. The annotation patches for those locations are combined and instantiated in the output annotation.*

$$
\begin{aligned}
&p(\boldsymbol{p} = a_j | o_n, \boldsymbol{\lambda}) \\
&\quad = \sum_{\boldsymbol{p} \in N(e)} \sum_i p(\boldsymbol{p} = a_j | c_i, o_n, \boldsymbol{\lambda}) \\
&\quad \times p\big(\hat{a}(\boldsymbol{p}) = a_e(\boldsymbol{p}) | e\big) p(e, c_i | o_n, \boldsymbol{\lambda}) \qquad (4)
\end{aligned}
$$

The components of this equation will be explained in detail next. The first and last factors are generalizations of their counterparts in eq. (3). They represent the annotations stored in the codebook, and the voting procedure, respectively. One extension consists in transferring annotations also from image patches *near* the pixel $\boldsymbol{p}$, and not only from those *containing* it. With the original version, it is often difficult to obtain full coverage of the object, especially when the number of training images is limited. By extending the neighbourhood of the patches, this problem is reduced. This is an important feature, because producing the training annotations can be labor-intensive. Our notion of proximity is defined relative to the size of the image patch, and parameterized by a scale-factor s_N. More precisely, let an image patch $e = (e_x, e_y, e_s)$ be defined by the three-dimensional coordinates of its center (e_x, e_y) and scale e_s obtained from the interest point

detector. The neighbourhood $N(e)$ of e is defined as:

$$N(e) = \{\boldsymbol{p} | \boldsymbol{p} \in (e_x, e_y, s_N \cdot e_s)\} \quad (5)$$

A potential disadvantage of the above procedure is that for $\boldsymbol{p} = (p_x, p_y)$ outside the actual image patch, the transferred annotation is less reliable. Indeed, the pixel may lie on an occluded image area, or small misalignment errors may get magnified. Moreover, some differences between the object instances shown in the training and test images that were not noticeable at the local scale can now affect the results. To compensate for this, we add the second factor to eq. (4), which indicates how probable it is that the transferred annotation $a_e(\boldsymbol{p})$ still corresponds to the 'true' annotation $\hat{a}(\boldsymbol{p})$. This probability is modeled by a Gaussian, decaying smoothly with the distance from the center of the patch e, and with variance related to the size of e by a scale factor s_G:

$$\begin{aligned} p\big(\hat{a}(\boldsymbol{p}) = a_e(\boldsymbol{p}) \,|\, e\big) &= \frac{1}{\sigma\sqrt{2\pi}} exp\left(-\frac{{d_x}^2 + {d_y}^2}{2\sigma^2}\right) \\ \text{with} \qquad \sigma &= s_G \cdot e_s \\ (d_x, d_y) &= (p_x - e_x, p_y - e_y) \end{aligned} \quad (6)$$

Once we have computed the probabilities $p(\boldsymbol{p} = a_j)$ for all possible labels $\{a_j\}_{j=1:N}$, we come to the actual assignment: we select the most likely label for each pixel. Note how for some applications, it might be better to keep the whole probability distribution $\{p(\boldsymbol{p} = a_j)\}_{j=1:N}$ rather than a hard assignment, e.g. when feeding back the information as prior probabilities to low-level image processing.

An interesting possible extension is to enforce spatial continuity between labels of neighboring pixels, e.g. by relaxation or by representing the image pixels as a Markov Random Field. In our experiments (Section 6), we achieved good results already without enforcing spatial continuity.

The practical implementation of this algorithm requires rescaling the annotation patches. In the original ISM system, bilinear interpolation is used for rescaling operations, which is justified because segmentation data can be treated as continuous probability values between 0 and 1. However, interpolating over discrete labels such as 'windshield' or 'bumper', which in practice are numerical values too, does not make sense. Therefore, rescaling must be carried out without interpolation.

4.2 Transferring Real- or Vector-valued Meta-data

In many cases, the meta-data is not discrete, but real-valued (e.g. 3D depth) or vector-valued (e.g. surface orientation). We will first explain how we obtain a real-valued annotation from quantized training data, and then how fully continuous meta-data is processed.

4.2.1 Quantized Meta-data

If the available training meta-data is quantized, we can use the discrete system as in the previous section, but still obtain a continuous estimate for the output by means of interpolation. Treating the quantized values as a fixed set of 'value labels', we infer for each pixel a probability for each discrete value (4). Next, we select the discrete value label with the highest probability, as before. To refine this value, a parabola is fitted to the probability scores for the maximum value label and the two immediate neighbouring value labels. The value corresponding to the maximum of the parabola yields the final estimate. This is a similar method as used in interest point detectors (Lowe, 2004; Bay et al., 2006) to determine continuous scale coordinates and orientations from discrete values. Thanks to this interpolation procedure, we obtain real-valued output even though the training meta-data is quantized.

4.2.2 Continuous and Vector-valued Meta-data

Processing fully real-valued or vector-valued meta-data requires a different approach. Instead of building probability maps for discrete labels, we store for each pixel all values that have been voted for, together with their vote weights. We again use Eq. 6 to decrease the influence of votes with increasing distance from their patch location. By storing all votes, we obtain a sampling of the probability distribution for each pixel. To extract a final value for each pixel, we estimate the mode of this distribution, using a Mean Shift Procedure. This is more robust to outliers than e.g. taking the value with the heighest weight or the average.

We use a Mean Shift procedure (Cheng, 1995) with a fixed window radius to estimate the mode for each pixel. This method works for 1-dimensional as well as vector-valued data. The mode estimation procedure uses a set

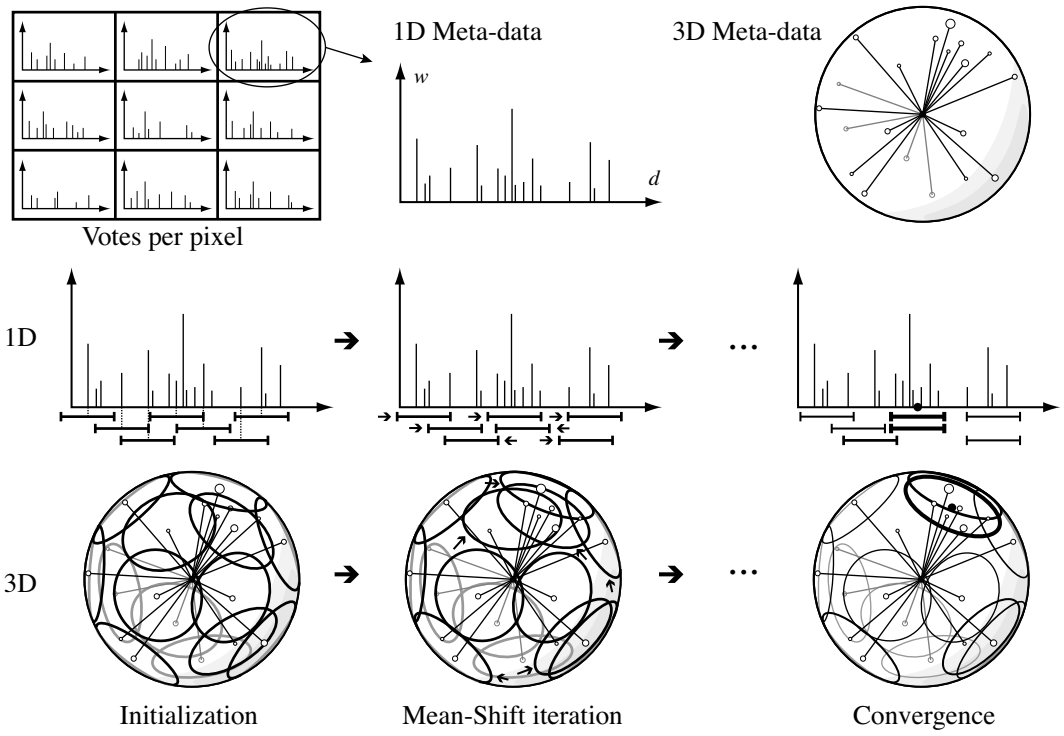

Figure 4: *Mean-Shift mode estimation for continuous and vector-valued meta-data. The top left shows a 3x3 pixel fragment from an image, with 1D vote distributions for each pixel. The top right shows another possible distribution where each vote is a 3D normal vector (the size of the circles indicates the vote weights). The middle and bottom row show the Mean-Shift mode estimation procedure for both types of data. In the rightmost figures, the line width of the windows corresponds to their scores and the black dot is the final value.*

of candidate windows, which are iteratively shifted towards regions of higher density until convergence occurs. Because the number of votes is small, in the order of one hundred, there is no need to initialize the windows through random sampling as done in other works (Cheng, 1995). Instead, we cover the entire distribution with candidate windows by considering the location of each vote as a candidate window, and removing all overlapping windows. Two windows overlap if their distance is less than the window radius. Depending on the type of data, distance can be defined as Euclidean distance, or as the angle between vectors. Next, we iterate over all windows by moving each window to the weighted mean of all votes within its radius, until convergence occurs. The score of a window is the sum of the weights of all its votes. The coordinates of the window with the highest score yield the position $\hat{\boldsymbol{a}}$ of the mode. The estimate for the final value for $\boldsymbol{p}$ can be formulated as:

$$\hat{\boldsymbol{a}}(\boldsymbol{p}) = \underset{\boldsymbol{a}}{\operatorname{argmax}} \sum_{d(\boldsymbol{a},\boldsymbol{a}_i(\boldsymbol{p}))<\theta} w\big(\boldsymbol{a}_i(\boldsymbol{p})\big) \qquad (7)$$

The scalar or vector value $\boldsymbol{a}_i(\boldsymbol{p})$ expresses the i-th vote for the value of pixel $\boldsymbol{p}$. The function $d(\boldsymbol{x},\boldsymbol{y})$ is a distance measure between meta-data values, θ is the mean-shift window radius, and $w\big(\boldsymbol{a}_i(\boldsymbol{p})\big)$ is the weight of the i-th vote. In case there are multiple modes with the same score, we take the average position (this occurs rarely in our experiments). The label 'background' is assigned if the score of the window around $\hat{\boldsymbol{a}}$ is smaller than the sum of the weights of background votes.

Figure 4 illustrates the mode estimation procedure for both 1-dimensional meta-data (e.g. depth values) and 3-dimensional normal vectors. In the latter case, the windows are circles on a unit sphere, and the distance measure between the votes and windows is the angle between their vectors. When updating the window positions, care must be taken to keep the resulting vectors normalized. When the meta-data consists of vectors that need to be compared using Euclidean distance (e.g. 3D points), the windows are (hyper)spheres of the same dimension as the vectors.

4.3 Refining Hypotheses

When large areas of the object are insufficiently covered by interest points, no meta-data can be assigned to them. Using a large value for s_N will only partially solve this problem, because there is a limit as to how far information from neighboring points can be reliably extrapolated. Too large an s_N may cause the annotation to 'leak' into the background and small details to be drowned out. A better solution is to actively search for additional codebook matches in these areas. The refinement procedure in early, fixed-scale versions of the ISM system (Leibe and Schiele, 2003) achieved this by means of uniform sampling. A dense 2D grid of candidate points was generated around the hypothesis, which is intractable in the scale-invariant (3D) case. Therefore we have developed a more efficient refinement algorithm which only searches for matches at promising locations.

For each hypothesis, new candidate points are generated by backprojecting all occurrences in the codebook, excluding points nearby existing interest points. When the feature descriptor for a new point matches with the codebook cluster(s) that backprojected it, an additional hypothesis vote is cast. The confidence for this new vote is reduced by a penalty factor to reflect the fact that it was not generated by an actual interest point. This penalty factor is 0.5 in all our experiments. The additional votes enable the meta-data transfer to cover those areas that were initially missed by the interest point detector. This procedure is illustrated in Figure 5. As can be seen from figure 5, this refinement step is a vital part to obtain a good coverage of the object.

This refinement step can either be performed on the final hypotheses that result from the MDL verification, or on all hypotheses that result from the initial voting. In the latter case, it will improve MDL verification by enabling it to obtain better figure area estimates of each hypothesis (Leibe et al., 2008). Therefore, we perform refinement on the initial hypotheses in all our experiments.

5 Multi-View extension of the ISM

In this section, we describe how we can achieve multi-view object class detection and annotation in a more efficient and higher performance way than simply running a battery of single-view detectors. We establish relations between different views of the same object. There are several approaches to this problem, e.g. the view clustering method by Lowe (2001) and the *image exploration* algo-

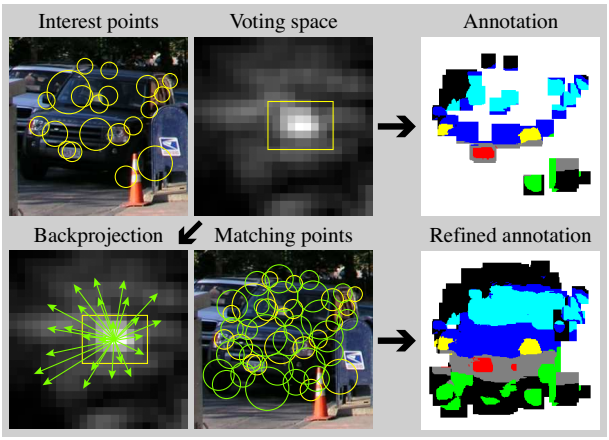

Figure 5: *Refining a hypothesis. An image with poor contrast (top left) produces insufficient interest points to cover the whole object (top right). By backprojecting the occurrence locations from the detected peak in the Hough space (bottom left), additional points can be found (bottom center), and a more complete annotation can be constructed (bottom right).*

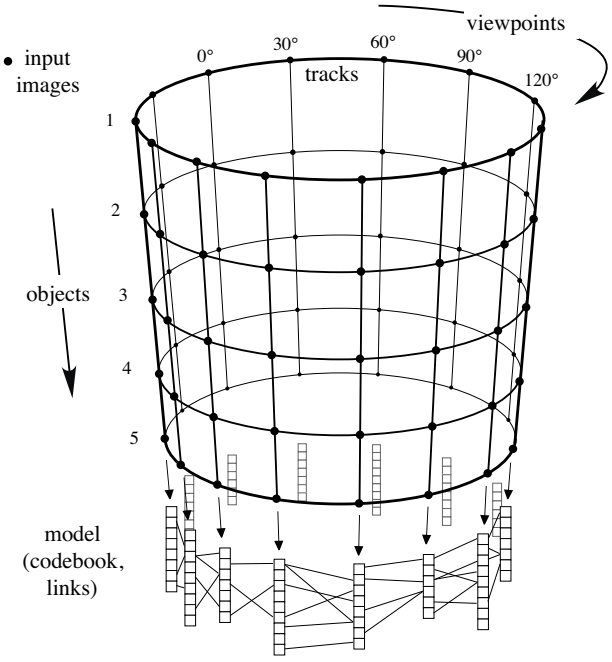

Figure 6: *Visualization of our multi-view model. Only viewpoints lying on a circle around the object are shown. However, the proposed method supports the general case of viewpoints distributed over the whole viewing sphere.*

rithm proposed by Ferrari et al. (2004, 2006). We build upon the latter method, which is designed for establishing dense correspondences among multiple model views of a specific object. In this work, we apply image exploration in the following fashion: for each specific training object, a set of *region tracks* is produced, densely connecting its model views. Each such track is composed of the image regions of a single physical surface patch along the model views in which it is visible.

The global scheme of the multi-view system is as follows. Initially, both a set of ISM models and exploration systems are trained separately on the same dataset. This dataset consists of images of M object instances, taken from N viewpoints. The viewpoints should approximately correspond to a fixed set of poses, but each instance does not need to have all viewpoints. In practice, it is sufficient to walk around each of the objects with a camera, and take images at approximately corresponding viewpoints. The total set of training images can be considered as an $M \times N$ matrix, with each row corresponding to an object instance and each column to a viewpoint (figure 6). A set of N ISMs are then trained independently (one ISM for each column), and M sets of region tracks are extracted (one set for each row). The next step is to establish relations between the single-view ISM models, consisting of so-called *activation links*.

In Section 5.1, we first summarize how to obtain multi-view tracks with the method of Ferrari et al. (2004, 2006). Next, we explain in Section 5.2 how the tracks are integrated in the ISM system, to construct activation links during training (Section 5.2.1) and to use these for improving recognition of a test image (Section 5.2.3).

5.1 Dense Multi-View Correspondences by Image Exploration

Finding relations between the different views of an object instance is a two-stage process. First, dense two-view matches are produced between each model image and all other images within a limited neighborhood on the view-

Figure 7: *Top: some of the region-tracks found across 3 views of a motorbike; bottom: all of them.*

ing sphere. Next, all pairwise sets of matches are integrated into a single multi-view model.

Region correspondences between two model views v_i and v_j are obtained via Ferrari et al. (2004). The method first generates a large set of low confidence, initial region matches, and then gradually *explores* the surrounding areas, trying to generate more and more matches, increasingly farther from the initial ones. The exploration process exploits the geometric transformations of existing matches to construct correspondences in view v_j, for a number of overlapping circular regions, arranged on a grid completely covering view v_i (*coverage regions*). This is achieved by iteratively alternating expansion phases, which construct new matching regions in v_j, with contraction phases that remove mismatches. With each iteration, the correct matches cover more and more of the object, while the ratio of mismatches progressively decreases. The result is a large set of reliable region correspondences, densely covering the parts of the object visible in both views.

Pairs of model views are matched within a limited neighborhood around each view. Next, the resulting two-view correspondences are organized into multi-view region tracks (Ferrari et al., 2006). The crucial point is to use always the same coverage regions when matching a certain view to any of the other model views. As a consequence, each region-track is directly defined by a coverage region together with all regions it matches in the other views (figure 7).

5.2 Integrating the Multi-View Correspondences with the ISM

With the tracks learnt from the image exploration algorithm, we can make the different single-view codebooks communicate with each other by means of *activation links*. This results in additional votes being inserted into a codebook's voting space, based on activations in the other codebooks. Section 5.2.1 explains how to generate the activation links. Sections 5.2.2 and 5.2.3 explain how the multi-view model is used during recognition.

5.2.1 Training: Establishing Activation Links

The image exploration system (Section 5.1) produces a set of tracks per training object, each containing regions corresponding across the object's model views. These regions are described by ellipses, i.e. affine transformations of the unit circle (figure 7). Regions are constructed so that the affine transformation between two regions in a track approximates the transformation between the image patches they cover. The goal of the linking stage is to establish connections between the different ISMs. These connections consist of *activation links* between the occurrences, indicating which occurrences in different ISMs correspond to the same object part. Because the ISM and image exploration systems have different goals, they use different features, so there is no one-to-one correspondence between regions and occurrences.

Before explaining how to use multi-view tracks to produce activation links, we first report on a subproblem: how to find the region R_i closest to an occurrence O_i. This problem boils down to finding in a set of ellipses (all regions in an image) the one nearest to a point (the center of O_i). An analytical solution for this problem exists, but is computationally expensive. Therefore, we use as an approximation the distance to a line segment of length $\|\mathbf{l}\| - \|\mathbf{s}\|$, aligned with the major axis of the ellipse, with $\mathbf{l}$ and $\mathbf{s}$ the major and minor axes respectively.

Occurrences are assigned to the nearest region only if they are within a distance $2 \cdot \|\mathbf{s}\|$. This assumes that the affine transformation of a region is typically valid within a small area around it (figure 8).

With this approximate distance measure, we are now ready to link the different ISMs together, by creating activation links between occurrences in different training

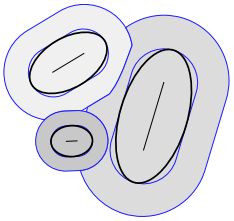

Figure 8: *Attraction zones for regions. The figure shows the areas in which occurrences would be assigned to one of three elliptical regions, using the distance to a line segment as an approximation for the distance to an ellipse.*

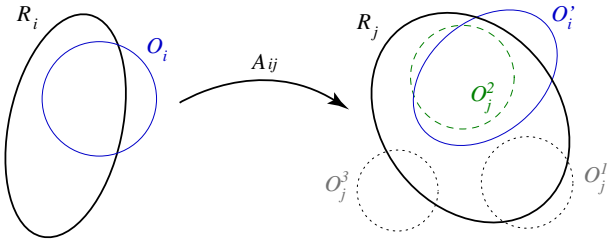

Figure 9: *Establishing links between occurrences. A_{ij} is the affine transformation between the region R_i in view i and R_j in view j. O_i' is obtained by mapping occurrence O_i from view i to view j using A_{ij}. In this example, a link between occurrences O_i and O_j^2 is created, because O_j^2 is sufficiently similar to O_i'.*

views. Activation links are created per object instance, i.e. one link only connects occurrences belonging to a specific training object. The algorithm iterates over all occurrences O_i in all training views of this object. For each O_i, it looks for the nearest region R_i, using the approximate distance measure described above. Then, we treat every other view v_j in the region's track as follows (figure 9). The circular region corresponding to O_i is first transformed with the affine transformation A_{ij} between R_i and R_j, i.e. $O_i' = A_{ij} \cdot O_i$. Next, we look for occurrences O_j^k in view v_j whose geometry is sufficiently similar to O_i'. All $O_i \rightarrow O_j^k$ are then stored as activation links.

Again, matching the occurrences O_j^k to O_i' involves the comparison between circles and an ellipse. However, this time we do not look for the nearest circle to the ellipse, but for all circles sufficiently similar to the ellipse. We

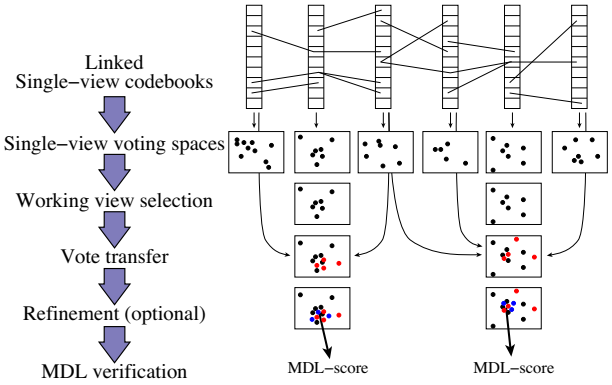

Figure 10: *Overview of the multi-view recognition scheme. After performing voting in all views, the most promising views are selected (Section 5.2.2). Evidence is transferred from all other views towards each working view, using the activation links (Section 5.2.3). After an optional refinement step (Section 4.3), the MDL procedure (Section 3.3) produces the final detection scores.*

use the following heuristics to determine whether a circle with center $\mathbf{p_c}$ and radius R matches an ellipse with center $\mathbf{p_e}$ and major/minor axis lengths $\|\mathbf{l}\|, \|\mathbf{s}\|$:

$$\|\mathbf{p_c} - \mathbf{p_e}\| < a \cdot R \quad (8)$$

$$\left|1 - (\|\mathbf{s}\| \cdot \|\mathbf{l}\|)/R^2\right| < b \quad (9)$$

$$\|\mathbf{s}\|/R > 1/c \quad (10)$$

$$\|\mathbf{l}\|/R < d \quad (11)$$

with a, b, c, d parameters, set to $a = 0.35$, $b = 0.25$, $c = d = 3.0$ in all reported experiments. These formulas put constraints on the distance between the centers, the ratio between the areas, the ratio between the minor axis and the radius, and the ratio between the major axis and the radius, respectively.

5.2.2 Recognition: Selecting Working Views

The early processing stages for detecting an instance of the object class in a novel image are similar to those of the original ISM framework (Section 3). Features are extracted from the image, and matched to all the codebooks of the different ISMs. Next, votes are cast in the Hough spaces of each ISM separately, and initial hypotheses are

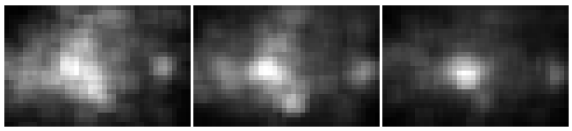

Figure 11: *Voting spaces for three neighbouring viewpoints at a certain scale. Note how strong hypotheses appear at similar locations.*

detected as local density maxima in these spaces. Up to this point, our system works in a similar fashion as a bank of independent single-view detectors.

Figure 10 illustrates the different steps in the multi-view recognition procedure, which will be explained in detail next. After initial hypotheses are found in each view separately, our system estimates which views are likely to match the actual pose(s) of the object(s) in the test image. We will refer to these views as *working views*.

We observed that a correct strong hypothesis is often corroborated by other strong hypotheses at similar locations in the voting spaces of neighbouring views (figure 11). This can be explained by the fact that there is some continuity in the voting spaces from one viewpoint to the next. Moreover, the pose of an object in a test image may fall in between the canonical poses of two training views. We tested a few different criteria to select working views. We assumed that by detecting clusters of nearby hypotheses across views, a more stable estimation of the correct pose may be possible. Surprisingly however, this is not the case: the most straightforward criterion proves to be the best performing. Instead of clustering nearby hypotheses across views like in Thomas et al. (2006), we pick the strongest hypothesis across all views, and define a threshold $\tau = T \cdot s_{max}$, with s_{max} the score of the strongest hypothesis and $T = 0.7$ in our experiments. The set of working views is defined as all views that contain at least one hypothesis whose score is above τ.

5.2.3 Recognition: Transferring Votes Across Views

The next stage is to *augment* the Hough spaces of each working view, by inserting additional votes that stem from codebook matches in other views. This is where the activation links come into play. Since working views are candidates for the actual pose of the object to be detected, the following process is repeated for each working view. After augmenting the Hough space of a working view, local peaks are detected again, and the MDL stage of Section 3 is performed on the resulting hypotheses. Detections after the MDL stage are the output of our system.

The key idea for augmenting the Hough spaces is the following. If a feature matches to a codebook entry in view v_i, we look if that entry has occurrences linking to our working view v_j. If we find such activation links, we cast additional votes in view v_j. We call this process *transferring votes* (see Figure 12). In other words, if we detect an object part in the codebook of view v_i, but we have found view v_j to be a more likely pose for the object, we transfer the evidence of the part to view v_j. Therefore, to cast the transferred vote we use information from both v_i's and v_j's ISMs. Remember that during the original voting stage, votes are cast for possible object positions. These are computed as the sum of the position where a codebook entry matches in the test image, and the relative positions of the occurrences to the center of the object in the training images. To determine the position of a transferred vote, we assume that when detecting a part in view v_i, the same part may be present in view v_j at approximately the same position. Therefore, the position of the transferred vote is calculated as the sum of the coordinates where the codebook entry matched in view v_i, and the relative coordinates of the occurrence in view v_j. Since the estimate for the object center is inevitably less accurate than in the single-view case, we use a larger kernel size when detecting peaks in the augmented Hough spaces. This compensates for the larger variance in the votes' positions.

The weight of the transferred votes is determined by extending eq. (1) to the multi-view system. This formula expresses the contribution of a patch e to an object hypothesis (o_n, λ):

$$\begin{aligned} p(o_n, \boldsymbol{\lambda}|e, \boldsymbol{\ell}) = \\ \sum_k p(o_n, \boldsymbol{\lambda}|c_k^j, \boldsymbol{\ell}) p(c_k^j|e) + \\ \sum_k \sum_l P(o_n, \boldsymbol{\lambda}|c_k^j, c_l^i, \boldsymbol{\ell}) p(c_l^i|e) \end{aligned} \quad (12)$$

with v_j the current working view. The first term is as in eq. (1). The summation over k runs over all codebook entries for view v_j. The summation over l runs over all

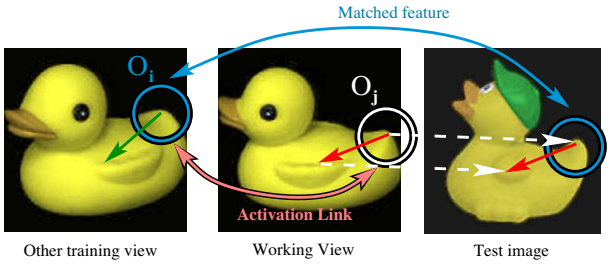

Figure 12: *Vote tranfer. The codebook entry containing occurrence O_i matches to the test image, but another view is selected as working view. Therefore, a vote for O_j is cast.*

other codebooks' entries, i.e. for views $v_i \neq v_j$. In this summation, the factor $p(c_l^i|e)$ is the probability that entry c_l^i is a correct interpretation for patch e. Just like in the original ISM system, we assume a uniform distribution here. $P(o_n, \boldsymbol{\lambda}|c_k^j, c_l^i, \boldsymbol{\ell})$ is non-zero only if there exists an activation link between c_l^i and c_k^j. It expresses the spatial distribution of transferred votes from occurrences in codebook entry c_l^i to occurrences in codebook entry c_k^j. This distribution consists of a set of weighted Dirac-impulses in the 3D Hough space at locations as described above. The weights of these impulses are derived as follows. Each of the K occurrences in codebook entry c_l^i has probability $1/K$ to yield the correct vote for the object center (under the uniform distribution assumption). If this occurrence has L links towards view v_j, the probability for each link to be valid is $1/L$. Therefore, each impulse in the transferred vote distribution should be weighted by $1/(KL)$. Note that, compared to the weights of the *direct* votes, which originate from view v_j itself, there is an additional factor of $1/L$. The weights of transferred votes are lower than direct ones, which adequately mirrors the fact that they are more numerous and less reliable individually.

5.2.4 Multi-view Meta-data Transfer

Transferring meta-data in the multi-view case is analogous to the single-view case from Section 4. Naturally, votes originating from within each working view are used to construct the output annotation. Moreover, transferred votes contribute as well, as if they were regular votes inside a working view. Transferred votes are treated as if they would originate directly from the interest point that triggered the vote transfer (see Figure 12). Thanks to this mechanism, even if there was no direct match from the working view for that point, the patch can still be annotated, leading to a more complete meta-data annotation.

6 Experimental evaluation

We evaluate our approach with several experiments on three different object classes: wheelchairs, cars, and motorbikes. Each experiment is designed to test a specific aspect of the system. We start with two experiments in a controlled scenario to assess the annotation ability of the system. In the first experiment we perform part decomposition for wheelchairs, which is a discrete labeling problem (6.1). The second experiment shows discrete, continuous, and vector-valued meta-data transfer on cars, where in addition to part decomposition (6.2.1) we also recover depth and 3D orientation information (6.2.2). Next, we demonstrate simultaneous recognition and annotation on challenging real-world images (6.3). Finally, in Section 6.4 we use the class of motorbikes to demonstrate the multi-view extension from Section 5.2.

These experiments demonstrate how the recognition of previously unseen object class instances can be coupled with the inference of additional information about the recognized object. The possibilities are not limited to the examples shown. Any type of information that can be attached to images in a pixel-wise fashion can be used as meta-data. Other possible examples include the expected motion vector field or a temperature map for a recognized object. The inferred data can be used for action planning or can be compared with actual measurements to detect unusual events, e.g. in a surveillance application.

6.1 Wheelchairs: Indicating Areas of Interest for a Service Robot

In the first experiment, the goal is to delineate certain areas of interest on the objects, which is a discrete annotation task. For the class of wheelchairs, a possible application is a service robot. This robot's task could be to retrieve a wheelchair, for instance in a hospital or to help a disabled person at home. In order to retrieve the

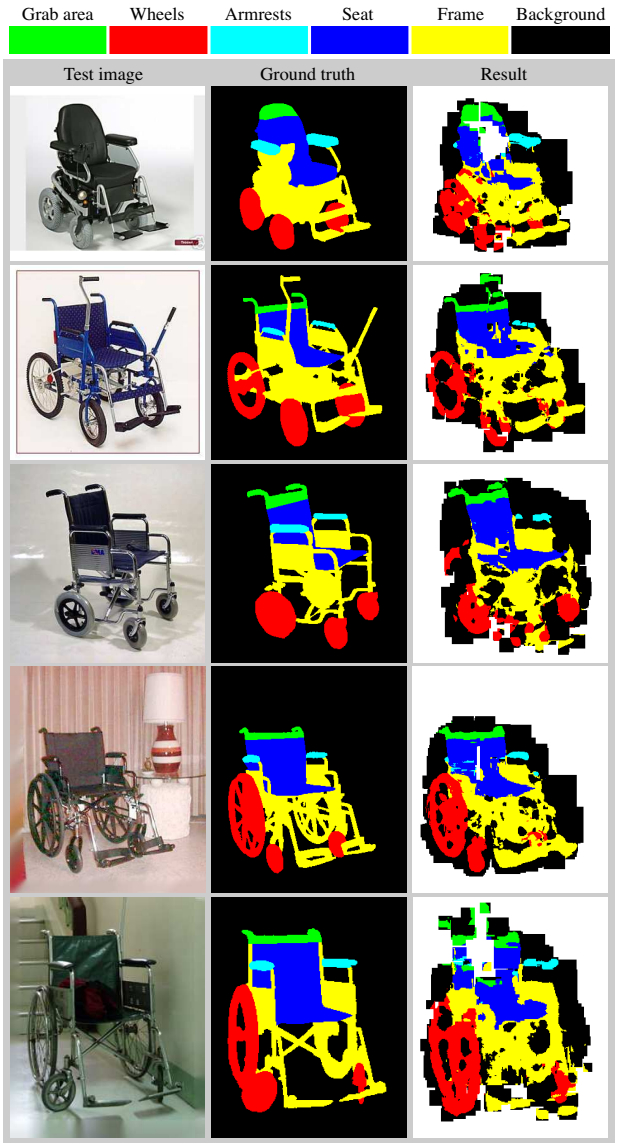

Figure 13: *Results for the annotation experiment on wheelchair images. From left to right: test image, ground-truth, and output of our system. White areas are unlabeled and can be considered background.*

wheelchair, the robot must be able to both detect it and determine where to grab it. Our method will help the robot to approach the grabbing position, after which a detailed analysis of the scene geometry in a small region can be used to finetune the grasp (e.g. Saxena et al., 2006).

We collected 141 images of wheelchairs from Google Image Search. We chose semi-profile views because they were the most widely available. All images were annotated with ground-truth part segmentations of the grab area, wheels, armrests, seat, and frame. In our assistive robot scenario, the grab area is the most important one. A few representative images and their ground-truth annotations can be seen in the left and middle columns of Figure 13.

The images are randomly split into training and test set. We train an ISM on 80 images using the Hessian-Laplace interest point detector (Mikolajczyk and Schmid, 2004) and local Shape Context descriptors (Belongie et al., 2000), because this combination has been shown to perform best in Seemann et al. (2005). Next, we test the system on the remaining 61 images, using the method from Section 4.1. In this experiment and all the following, $s_N = 3$ and $s_G = 1.4$. The goal of this first experiment is to assess the quality of the annotations only, not the recognition performance, which will be demonstrated in Section 6.3. Because each image only contains one object, we select the detection with the highest score for meta-data transfer. Some of the annotations produced by our system can be seen in the third column of Figure 13. The grab area is accurately localized.

To evaluate this experiment quantitatively, we use the ground-truth annotations to calculate the following error measures. We define *leakage* as the percentage of background pixels in the ground-truth annotation that were labeled as non-background by the system. The leakage for this experiment, averaged over all test images, is 3.75%. We also define a *coverage* measure, as the percentage of non-background pixels in the ground-truth images labeled non-background by the system. The coverage obtained by our algorithm is 95.1%. This means our method is able to accurately segment the wheelchair from the background.

We evaluate the annotation quality of the individual parts with a confusion matrix. For each test image, we count how many pixels of each part a_j in the ground-truth are labeled by our system as each of the possible parts (grasp, wheels, etc.), or remain unlabeled. Unla-

	backgrnd	frame	seat	armrest	wheels	**grab-area**	unlabeled
backgrnd	**32.58**	1.90	0.24	0.14	1.10	0.37	**63.67**
frame	15.29	**66.68**	6.47	0.46	6.90	0.10	4.10
seat	2.17	15.95	**74.28**	0.97	0.33	1.55	4.75
armrest	11.22	5.62	29.64	**49.32**	1.25	0.63	2.32
wheels	13.06	9.45	0.36	0.07	**71.39**	0.00	5.67
grab-area	6.48	1.28	9.77	0.11	0.00	**76.75**	5.62

Table 1: *Confusion matrix for the wheelchair part annotation experiment. The rows represent the annotation parts in the ground-truth maps, the columns the output of our system. The last column shows how much of each class was left unlabeled. For most evaluations, those areas can be considered as "background".*

beled areas can be due to lack of codebook matches at that location, or because they are too far away from any detected object. This pixel count is normalized by the total number of pixels of that label in the ground-truth $\hat{a}_j$. We average the confusion table entries over all images, resulting in Table 1. The diagonal elements show how well each part was recovered in the test images. Not considering the armrests, the system performs well as it labels correctly between 67% and 77% of the pixels, with the highest score being for the part we are the most interested in, i.e. the grab area. Performance is lower for the armrests because they are the smallest parts in most images. Small parts have a higher risk of being confused with the larger parts in their neighborhood.

6.2 Cars: Indicating Areas of Interest for an Automated Car Wash

To show the versatility of our system, we present results on the object class 'car' for three different types of metadata. The first is a part decomposition as before. Next, we infer 3D properties, consisting of a depth map and surface orientations, from a *single* image of a previously unseen car. A possible application is the automated car wash from Figure 1. A decomposition into parts can be used to apply optimized washing methods to the different car parts. The 3D information would allow to optimize the washing process beforehand, based on the car's shape inferred by our system (both depth and orientations). This is an improvement over existing systems which are in most cases based on sensors to measure distances to the car, and they are only used locally while the machine is already running.

As a dataset we adopt a subset of the one used in Leibe et al. (2007). It was obtained from the LabelMe website (Russell et al., 2005) by extracting images labeled as 'car' and sorting them according to their pose. There are generally no images of the same car from different viewpoints. Therefore, we only use the 'az300deg' pose, which is a semi-profile view. In this pose both the car's front (windscreen, headlights, license plate) and side (wheels, windows) are visible. This allows for more interesting depth/orientation maps and part annotations compared to pure frontal or side views. The dataset contains a total of 139 images. We randomly picked 79 for training and 60 for testing.

6.2.1 Parts Decomposition

In a similar fashion as in Section 6.1, we annotated our car dataset with ground-truth part segmentations for body, windshield/windows, wheels, bumper, lights and license plate. The ISM is trained using the same interest point detector and descriptor as in Section 6.1. The testing phase is again performed with the method presented in Section 4.1. Results are shown in Figure 14. The leakage for this experiment is 6.83% and coverage is 95.2%.

Table 2 shows the confusion matrix for this experiment. Labeling performance is good, except for the headlights. Similarly to the armrests in the wheelchair experiment, this is as expected because the headlights are very small. They are therefore easily confused with the larger parts (body, bumper) surrounding them.

6.2.2 Inferring 3D Shape

To obtain ground-truth data for training and testing, for both depth and orientation, we manually align a 3D model

	bkgnd	body	bumper	headlt	window	wheels	license	unlabeled
bkgnd	**23.56**	2.49	1.03	0.14	1.25	1.88	0.04	**69.61**
body	4.47	**72.15**	4.64	1.81	8.78	1.86	0.24	6.05
bumper	7.20	4.54	**73.76**	1.57	0.00	7.85	2.43	2.64
headlt	1.51	36.90	23.54	**34.75**	0.01	0.65	0.23	2.41
window	3.15	13.55	0.00	0.00	**80.47**	0.00	0.00	2.82
wheels	11.38	6.85	8.51	0.00	0.00	**63.59**	0.01	9.65
license	2.57	1.07	39.07	0.00	0.00	1.04	**56.25**	0.00

Table 2: *Confusion matrix for the car parts annotation experiment. (cfr. Table 1)*

on top of each training image. The most suitable 3D model for each image is selected from a freely available collection[1]. Depth is extracted from the OpenGL Z-buffer. In general, any 3D scanner or active lighting setup could be used to automatically obtain 3D shape annotations during training. We normalize the depths based on the dimensions of the 3D models by assuming that the width of a car is approximately constant. Orientations are encoded by mapping each surface normal vector $\boldsymbol{n} = (x, y, z)$ to a 24 bit color $\boldsymbol{c} = (r, g, b)$ (e.g. with a fragment shader):

$$\boldsymbol{c} = 255 \cdot \big(\boldsymbol{n}/2 + (0.5, 0.5, 0.5)\big) \tag{13}$$

We train an ISM in a similar fashion as for the discrete annotation experiments, but with the real-valued (depth) and vector-valued (orientation) meta-data above. The system is tested on the 60 test images, using the method from Section 4.2.2. For the Mean Shift mode estimation, we use a window radius θ of 24% of the total depth range, and 60 degrees for the orientations. Some of the resulting annotations can be seen in the third and fifth columns of figure 15. For both the depthmap and orientation experiment, leakage is 5.7% and coverage is 94.6%, hence the segmentation performance is again very good.

It is possible to estimate the depth error in real-world units by scaling the normalized depth maps by a factor based on the average width of a real car, which we found to be approximately 1.80m. All depth maps are scaled to the interval $[0, 1]$ such that their depth range is 3.5 times the width of the car. In this scale, the average depth error is 3.94%. In order to eliminate bias from the background, this is only measured inside areas labeled as non-background both by the ground-truth and by our method. A plausible real-world depth error can therefore be calculated by multiplying this measure by $3.5 \cdot 1.80$m, which yields a mean error of 24.8cm. To better visualize how the output compares to the ground-truth, Figure 16 shows a few horizontal slices through two depth maps of Figure 15. Moreover, Figure 17 shows some views of a 3D model created by mapping the image of the recognized car onto the depth map produced by our system.

For the surface orientation experiment, we can calculate the average angular error over the area labeled as foreground both by the ground-truth and test image. The average error over all test images is 21.6 degrees. Part of this error is inherent to the dataset, because there is quite a large variability in the rotation of both training and test instances. Because our system combines the information from several different training examples, the orientations derived for a test image will be subject to a certain degree of averaging.

6.3 Combined recognition and annotation in cluttered images

To illustrate our system's ability to simultaneously detect objects in cluttered scenes and infer meta-data annotation, we report here another part decomposition experiment for wheelchairs and cars, this time on challenging, uncontrolled real-world images.

For wheelchairs, we collected 34 test images with considerable clutter and/or occlusion. The same ISM trained in Section 6.1 was used to detect and annotate wheelchairs in these images. Example results are shown in Figure 18. We consider a detection as correct when its bounding box overlaps at least 50% with the ground-truth bounding box. Out of the 39 wheelchairs present in the images, 30 were

[1] http://dmi.chez-alice.fr/models1.html

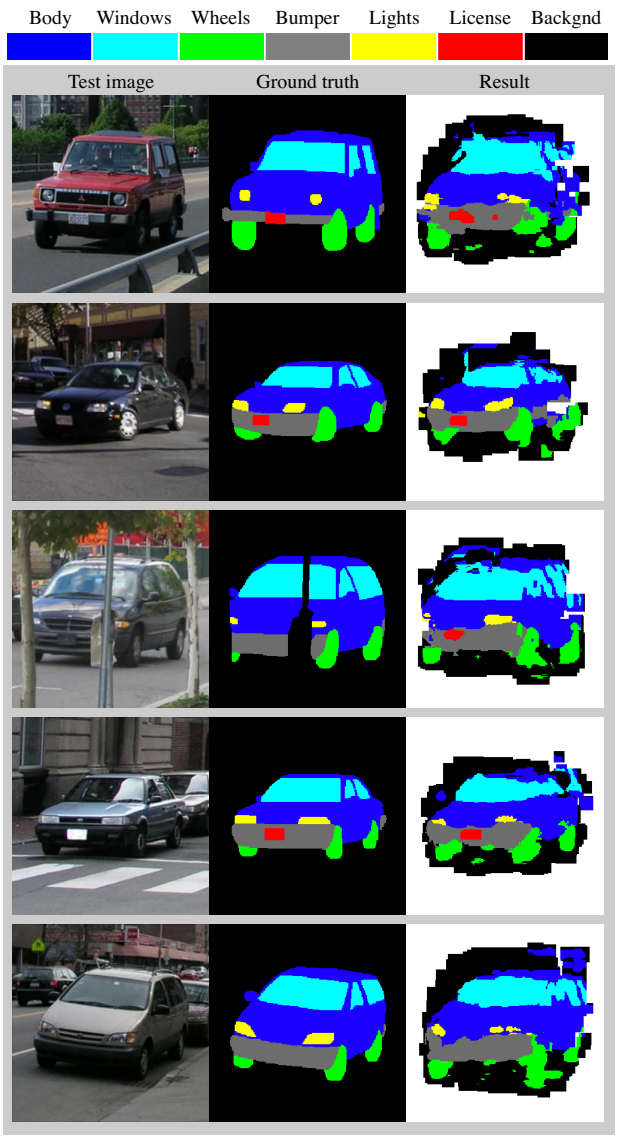

Figure 14: *Results for the car parts annotation experiment. From left to right: test image, ground-truth, and output of our system. White areas are unlabeled and can be considered background.*

detected, and there were 7 false positives. This corresponds to a recall of 77% and a precision of 81%. Results for the cars on real-world images are shown in Figures 1 and 19.

6.4 Multi-View Meta-data Annotation

In the last experiment, we demonstrate the multi-view extension to the ISM system as described in Section 5.2, to achieve multi-view annotation of meta-data. As a test case, we use a part decomposition experiment on motorbikes. Possible applications are in the field of traffic surveillance, or e.g. an automated parking system where the motorbike could be lifted by a robot and stored in a warehouse.

As training set we use the one of Thomas et al. (2006), and add 11 extra motorbikes. This amounts to a total of 41 instances, photographed from 16 positions on a circle around the object, approximately 22.5 degrees apart. Because it was impossible to collect all 16 viewpoints for every motorbike, there are 11 views on average for each motorbike, and only 4 bikes have all 16 images. As a result, there are on average 29 object instances available to train the ISM for a single viewpoint. This is a rather small number for training ISMs, but the refinement procedure (Section 4.3) compensates for the reduced number of occurrences. Each image was annotated with a part decomposition as in 6.1, discerning between the front, wheels, seat, body and background, as shown in Figure 20.

First, we perform recognition and parts annotation on the same motorbike testset as in Thomas et al. (2006). This set consists of the 'motorbikes-test2' subpart of the PASCAL Visual Object Classes (VOC) 2005 Challenge (Everingham et al., 2006), with duplicates removed. We again use the Hessian-Laplace interest point detector and Shape Context descriptor to train our multi-view system. We use the discrete meta-data transfer method of Section 4.1 combined with the multi-view method of Section 5.2 to achieve multi-view recognition and annotation. Using the same overlap removal and evaluation method as in Thomas et al. (2006), we obtain the precision-recall curve from Figure 21. Although this curve outperforms all other VOC2005 contestants on the same challenge, it cannot be immediately compared to those results. The other systems reported in Everingham et al. (2006) were trained on different instances, and evaluated on the en-

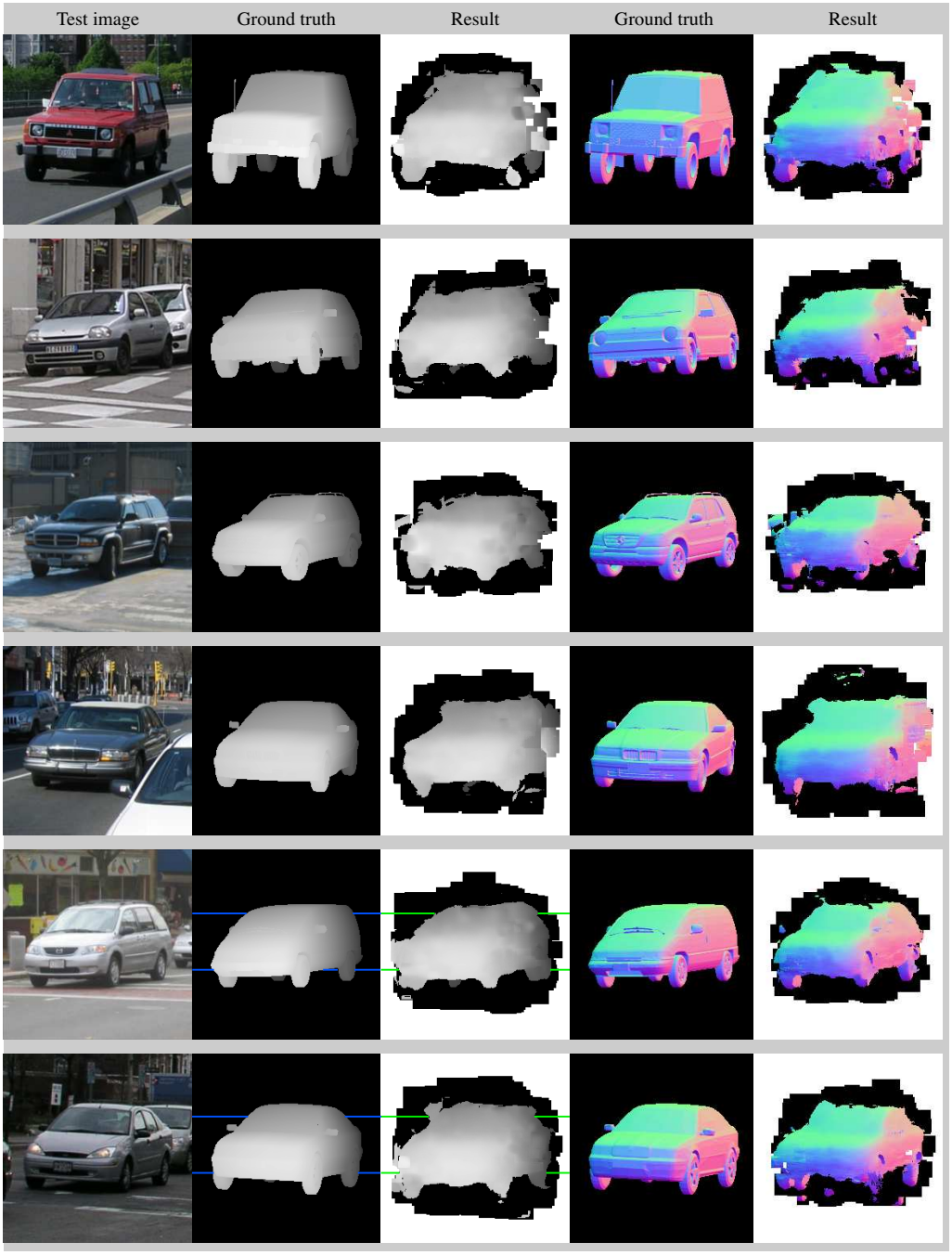

Figure 15: *Results for the car depth map and surface orientation experiments. From left to right: test image, ground-truth and output of our system for the depth map experiment, and ground-truth and output for the surface orientation experiment. The R,G,B colors represent the components of the surface normal according to Eq. 13. White areas are unlabeled and can be considered background. The lines in the last two rows indicate the slices from Figure 16.*

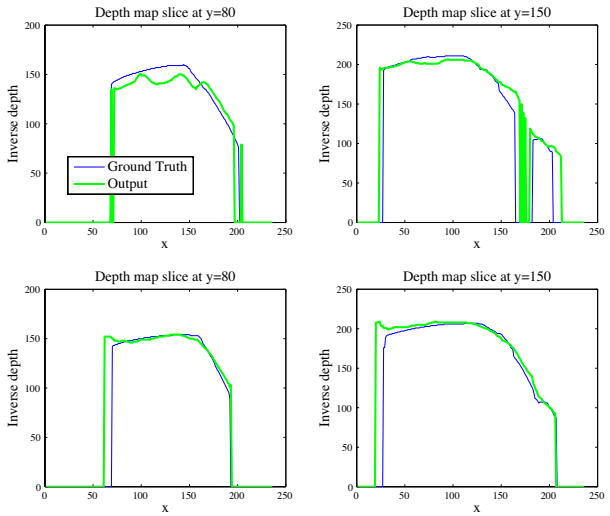

Figure 16: *Horizontal slices through the ground-truth and output depth maps of the fifth car (top) and sixth car (bottom) in Figure 15.*

Figure 17: *Some views of a texture mapped 3D model, generated from the depth map of the recognized car in the top left corner.*

tire VOC2005 set. The equal error rate (EER) is 71.8% and the average precision (AP) 73.8%. The improvement over Thomas et al. (2006), where the EER was 55.9% and the AP 56.0%, is due to the increased number of training images, the higher performance working view selection (Section 5.2.2), and the refinement step. Figure 21 also shows a confusion matrix for the estimated pose of the detections. Allowing a deviation of 22.5 degrees, our system estimated the correct pose for 86.9% of the correctly detected motorbikes. Figure 22 shows some detections together with their annotations. Despite the relatively small number of training instances, the quality of the annotations is good, and occlusions are handled correctly.

We report some additional qualitative results on the more recent VOC2007 Challenge (Everingham et al., 2007), using the same set-up as in the previous experiment. We test on the images marked as containing one or more motorbikes that can be recognized without the need of using context information. This amounts to a total of 222 images. The images contain various motorbikes in very diverse environments and poses, making this a very difficult test set. Figure 23 shows some detections with corresponding annotations. Again, our technique successfully localizes the motorbikes and delivers rather complete part decompositions over a wide range of viewpoints. Figure 24 shows a few images for which either detection, annotation or both failed. Many of the failures are due to objects that are in a pose that deviates too much from the set of training poses.

6.5 Processing Speed

Because our main focus is on proving the good detection and annotation ability of our system, our current implementation does not yet include any optimizations for speed nor memory usage. Although some parts of the algorithm have been multi-threaded, there is a lot of unexploited potential for parallel processing. Given the increasing trend towards multi-core computing, this means a substantial speed-up is certainly achievable. The refinement step (section 4.3) is the most computationally expensive. With refinement disabled, the average processing time per image for the multi-view algorithm (16 views) is about 2 minutes on an Intel Core 2 Quad Q6600 CPU. When refinement is enabled, the average time is about 8 minutes. Within the refinement step, calculating the de-

Figure 18: *Wheelchair detection and annotation results on challenging real-world test images. All detections are correct except for the two topmost ones in the center left image. Note how one wheelchair in the middle right image was missed because it is not in the pose used for training.*

Figure 19: *Car detection and annotation results on real-world test images. See also Figure 1.*

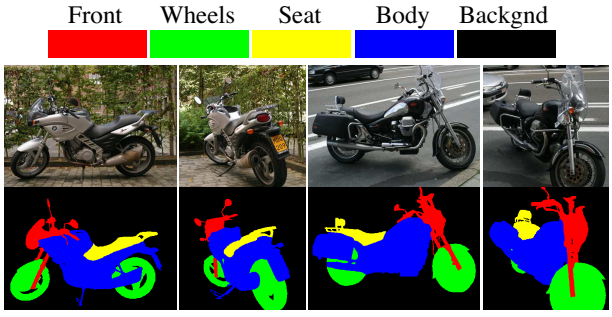

Figure 20: *Some of the multi-view motorbike images used for training, and their ground-truth part annotations.*

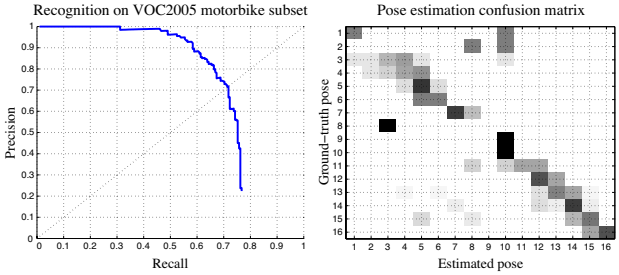

Figure 21: *Left: precision-recall curve for detection performance on the VOC2005 motorbikes test set. Right: confusion matrix for pose estimation within correct detections. The poses go counter-clockwise around the object starting at the frontal view. The best possible result would be a black main diagonal, and white everywhere else.*

scriptors for all points is the largest bottleneck, followed by the generating of the candidate points. For calculating the descriptors, we use a general purpose binary[2] which was not constructed with efficiency in mind. An optimized implementation (e.g. on a graphics card) could be an order of magnitude faster. In general, cluttered images require more processing time because they generate more hypotheses, and processing time is approximately linear in the number of hypotheses. By placing bounds on the number of working views and hypotheses per working view, the processing time and memory requirements per image can be bounded.

7 Conclusions

We have developed a method to transfer meta-data annotations from training images to test images containing previously unseen objects in arbitrary poses, based on object class recognition. Multi-view recognition is achieved by embedding relations between different views in our model. We have proposed a natural extension of the ISM method for the inference of meta-data directly as a result of the recognition process. The low-level cues that can lead to the detection of an object class instance in an image, are enriched with part labels, depths, orientations or any other type of meta-data. During recognition, our

[2] Available at http://www.robots.ox.ac.uk/~vgg/research/affine/

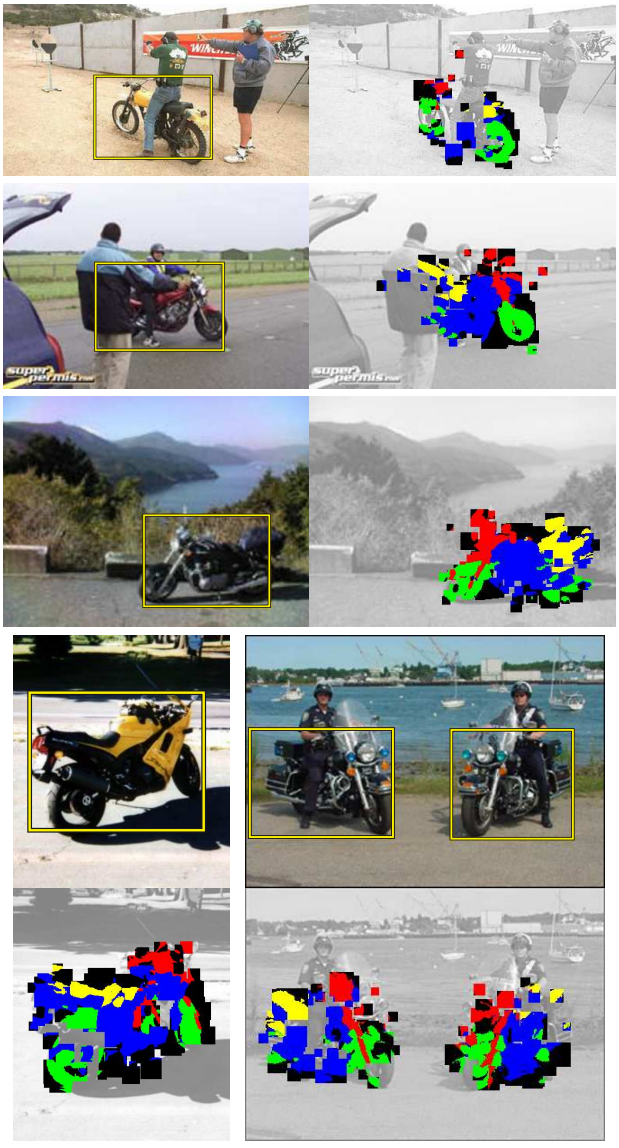

Figure 22: *Multi-view motorbike detection and annotation results on the VOC2005 dataset.*

Figure 23: *Multi-view motorbike detection and annotation results on the VOC2007 dataset.*

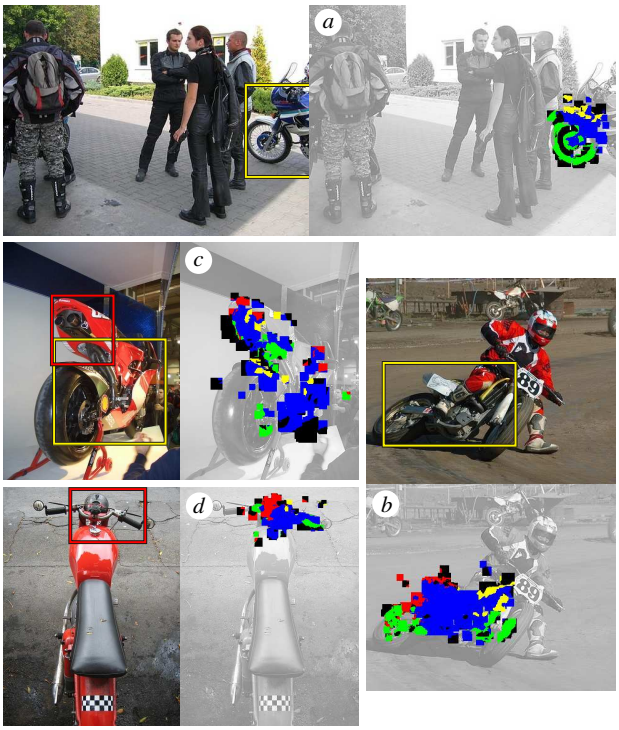

Figure 24: *Some failed detections and/or annotations on the VOC2007 dataset. Detections with a correct position are marked with a yellow (light) box, false positives with a red (dark) box. In (a) and (b) the motorbike is correctly detected, but the estimated pose is the reverse of the true pose. In (c) the scale is too small. The pose of (d) was not in the training set.*

system employs these cues to infer a similar meta-data annotation for previously unseen instances. Most other systems that allow automatic annotation of images, rely on local image statistics relative to an entire scene. The scene geometry needs to be consistent between the test and training images. This often prohibits reliable detection and/or accurate annotation of separate objects within the scene. Our system on the other hand cannot label an entire scene, but allows to focus on specific object classes without posing constraints on the scene. By coupling the detection and annotation processes, a reasonably accurate annotation can be inferred for detected objects. The allowed types of meta-data offer a wide range of possible applications. For instance, part, depth or surface normal annotations can be used to help a robot manipulate objects, or as input for other systems, forming a cognitive loop (e.g. priors for a 3D reconstruction algorithm).

Future work includes integration of these results in a real robotics application scenario, which will require optimizing the implementation and investigating ways to place bounds on the number of views and hypotheses without sacrificing performance. Also worth investigating is whether adding relaxation or Markov Random Fields can further improve the quality of the results.

Acknowledgment

The authors gratefully acknowledge support by IWT-Flanders, Fund for Scientific Research Flanders and European Project CLASS (3E060206).

References

H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *Proceedings ECCV, Springer LNCS*, volume 3951, pages 404–417, 2006.

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

G. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. In *Eur. Conf. on Computer Vision*, 2008.

Li Cheng, S. V. N. Vishwanathan, and X. Zhang. Consistent image analogies using semi-supervised learning. In *CVPR*, pages 1–8, 2008.

Y. Cheng. Mean shift, mode seeking, and clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 790–799, 1995.

N. Cornelis, B. Leibe, K. Cornelis, and L. J. Van Gool. 3D city modeling using cognitive loops. In *3DPVT*, pages 9–16, 2006.

M. Everingham, A. Zisserman, C. Williams, L. Van Gool, M. Allan, C. Bishop, et al. The 2005 PASCAL visual

object classes challenge. In *Selected Proceedings of the First PASCAL Challenges Workshop, LNAI*. Springer-Verlag, 2006.

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.

V. Ferrari, T. Tuytelaars, and L. Van Gool. Simultaneous object recognition and segmentation by image exploration. In *ECCV*, 2004.

V. Ferrari, T. Tuytelaars, and L. Van Gool. Simultaneous object recognition and segmentation from single or multiple model views. *International Journal of Computer Vision*, April 2006.

F. Fraundorfer, C. Engels, and D. Nister. Topological mapping, localization and navigation using image collections. In *IROS*, 2007.

T. Goedemé, T. Tuytelaars, and L. J. Van Gool. Fast wide baseline matching for visual navigation. In *CVPR*, volume 1, pages 24–29, 2004.

T. Goedemé, M. Nuttin, T. Tuytelaars, and L. Van Gool. Omnidirectional vision based topological navigation. *Int. Journal on Computer Vision and Int. Journal on Robotics Research*, 74(3):219–236, 2007.

F. Han and S.-C. Zhu. Bayesian reconstruction of 3D shapes and scenes from a single image. In *Workshop Higher-Level Knowledge in 3D Modeling Motion Analysis*, pages 12–20, 2003.

T. Hassner and R. Basri. Example based 3D reconstruction from single 2d images. In *Beyond Patches Workshop at IEEE CVPR06*, page 15, June 2006. ISBN 0-7695-2646-2.

A. Hertzmann, C. Jacobs, N. Oliver, B. Curless, and D. Salesin. Image analogies. In *SIGGRAPH Conference Proceedings*, pages 327–340, 2001.

D. Hoiem, A. A. Efros, and M. Hebert. Geometric context from a single image. In *ICCV*, pages 654–661, 2005.

D. Hoiem, A. Efros, and M. Hebert. Putting objects in perspective. In *CVPR*, pages 2137–2144, 2006.

D. Hoiem, Carsten Rother, and J. Winn. 3D LayoutCRF for multi-view object class recognition and segmentation. In *CVPR*, pages 1–8, 2007.

A. Kushal, C. Schmid, and J. Ponce. Flexible object models for category-level 3D object recognition. In *CVPR*, 2007.

B. Leibe and B. Schiele. Interleaved object categorization and segmentation. In *BMVC*, pages 759–768, 2003.

B. Leibe and B. Schiele. Pedestrian detection in crowded scenes. In *CVPR*, pages 878–885, 2005.

B. Leibe, N. Cornelis, K. Cornelis, and L. Van Gool. Dynamic 3D scene analysis from a moving vehicle. In *CVPR*, pages 1–8, 2007.

B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision*, 77 (1-3):259–289, 2008.

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

C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman. Sift flow: dense correspondence across different scenes. In *European Conference on Computer Vision*, pages 28–42, 2008.

D. G. Lowe. Local feature view clustering for 3D object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 682–688, December 2001.

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

D. Meger, P. Forssén, K. Lai, S. Helmer, S. McCann, T. Southey, M. Baumann, J. J. Little, and D. G. Lowe. Curious george: An attentive semantic robot. *Robot. Auton. Syst.*, 56:503–511, June 2008.

K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004.

D. Mumford. Neuronal architectures for pattern-theoretic problems. *Large-Scale Neuronal Theories of the Brain*, pages 125–152, 1994.

D. Munoz, N. Vandapel, and M. Hebert. Directional associative markov network for 3-d point cloud classification. In *Fourth International Symposium on 3-D Data Processing, Visualization, and Transmission (3DPVT)*, 2008.

C. Pantofaru, R. Unnikrishnan, and M. Hebert. Toward generating labeled maps from color and range data for robot navigation. In *Proc. 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 1314–1321, October 2003.

I. Posner, D. Schroeter, and P. Newman. Describing composite urban workspaces. In *International Conference on Robotics and Automation*, 2007.

K.S. Rockland and G.W. Van Hoesen. Direct temporal-occipital feedback connections to striate cortex (V1) in the macaque monkey. *Cereb. Cortex*, 4:300–313, 1994.

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. *IJCV*, 66(3):231–259, 2006.

B.C. Russell, A. Torralba, K.P. Murphy, and W.T. Freeman. LabelMe: a database and web-based tool for image annotation. *MIT AI Lab Memo AIM-2005-025*, 2005.

S. Savarese and Li Fei-Fei. 3D generic object categorization, localization and pose estimation. In *ICCV*, pages 1–8, 2007.

A. Saxena, J. Schulte, and A. Y. Ng. Learning depth from single monocular images. In *NIPS*, volume 18, pages 1–8, 2005.

A. Saxena, J. Driemeyer, J. Kearns, and A. Y. Ng. Robotic grasping of novel objects. In *NIPS*, volume 19, pages 1209–1216, 2006.

E. Seemann, B. Leibe, K. Mikolajczyk, and B. Schiele. An evaluation of local shape-based features for pedestrian detection. In *Proceedings of the 16th British Machine Vision Conference, Oxford, UK*, pages 11–20, September 2005.

S. Segvic, A. Remazeilles, A. Diosi, and F. Chaumette. Large scale vision-based navigation without an accurate global reconstruction. In *CVPR*, 2007.

A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. Van Gool. Towards multi-view object class detection. In *CVPR*, volume 2, pages 1589–1596, 2006.

A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. Van Gool. Using recognition to guide a robot's attention. In *Robotics: Science and Systems*, volume IV, 2008.

A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. Van Gool. Shape-from recognition: Recognition enables meta-data transfer. *Comput. Vis. Image Understand.*, 2009. doi: 10.1016/j.cviu.2009.03.010.

P. Yan, S. M. Khan, and M. Shah. 3D model based object class detection in an arbitrary view. In *ICCV*, 2007.