OBJECT REACQUISITION AND TRACKING IN LARGE-SCALE SMART CAMERA NETWORKS

Clemens Arth, Christian Leistner, Horst Bischof

Institute for Computer Graphics and Vision
Graz University of Technology
Inffeldgasse 16/2, 8010 Graz, AUSTRIA
{arth,leistner,bischof}@icg.tu-graz.ac.at

ABSTRACT

Object reacquisition or reidentification is the process of matching objects between images taken from separate cameras. In this paper, we present our work on feature based object reidentification performed on autonomous embedded smart cameras and applied to traffic scenarios. We present a novel approach based on PCA-SIFT features and a vocabulary tree. By building unique object signatures from visual features, reidentification can be done efficiently coevally minimizing the communication overhead between separate camera nodes. Applied to large-scale traffic scenarios, important parameters including travel time, travel time variability, section density, and partial dynamic origin/destination demands can be obtained.

The proposed approach works on spatially separated, uncalibrated, non-overlapping cameras, is highly scalable and solely based on appearance-based optical features. The entire system is implemented and evaluated with regard to a typical embedded smart camera platform featuring one single Texas Instruments ^{TM} fixed-point DSP.

Index Terms— Smart Camera Network, Object Recognition, Reacquisition, Vehicles, Vocabulary Tree

1. INTRODUCTION

In recent years, the development of algorithms for smart cameras has attracted immense attention in the Computer Vision community. This mainly resulted from the industrial interest in embedded systems in general and from the advantages of decentralized approaches over centralized image processing techniques. Nevertheless, the deployment of entire smart sensor networks and the implementation of algorithms for distributed computing yield many new problems. While energy efficiency and robustness to environmental stress are important factors for hardware design on the one hand, on the other hand the software developed for such systems has to be implemented for local autonomous processing and decision making. These requirements and more restrictions given by constrained resources place demanding challenges on software developers and system designers.

In this work we present a novel algorithm for object reacquisition in smart camera networks. The major contributions are the very efficient description of objects using local features, and the subsequent communication and matching algorithm proposed. First, objects are identified and described by robust local features. The object description is subsequently converted into an extremely compact representation by using a vocabulary tree, the so called *signature*. This signature might further be communicated to the neighboring camera nodes. The objects are again described and a specific signature for the object is created. The matching algorithm compares object signatures and allows for efficient reacquisition and tracking of the objects through entire camera networks.

One nice feature of our approach is that our algorithms minimize the amount of information per object to be transmitted between adjacent camera nodes while another one is that it has not to be re-trained for every single camera view. Additionally, one major goal of our work was to use uncalibrated setups since multi-camera calibration is still a tedious task. Finally, an important property of our system is that it is implemented with regard to a development board featuring a Texas Instruments^{TM}DSP, and that it runs fully autonomous coevally fulfilling real-time demands. The power of our approach is shown on the simulation of several traffic scenarios; first, a two-camera setup is considered and then a medium-scale camera network is simulated. The results presented proof our concept useful and motivate further research in the area of local features for object fingerprinting on embedded systems.

The paper is organized as follows. In section 2 we give an overview about applications and algorithms from the area of smart cameras. In section 4 we present our network topology specifications and our general framework design. A detailed description of the algorithms involved follows in section 3. Experimental results on two camera network setups are presented in section 5. Section 6 concludes the paper and gives an outlook on future work.

1-4244-1354-0/07/$25.00

2. RELATED WORK

In this section, we review some of the approaches for object reacquisition proposed in the past. We emphasize that most of the algorithms in this field come from the area of computer vision and do not take any embedded system related issues into account. Thus, in contrast to the work presented here, communication constraints or computational issues are usually hardly considered since all necessary computations are performed on a centralized engine. Additionally, most algorithms are designed for aerial imagery, thus capturing larger areas but, on the other hand, due to their lower resolutions are not able to benefit from the usage of local features.

In literature, many work concentrates on object instance recognition and fingerprinting, mostly for vehicles. For example, Guo *et al.* [1, 2] address the problem of matching vehicles across multiple sightings. They extract multiple features, *e.g.* line segments, of poor quality aerial images and hold these features in an integrated matching framework.

In one of most recent works Ali *et al.* [3] use both motion and appearance contexts for tracking of vehicles in aerial images. In contrast to Guo *et al.* they use a clustering scheme based on the Lyapunov Characteristic Exponent (LCE) to learn the motion context of multiple trajectories. Additionally, they use the motion of a car to interpret the behavior of neighbored cars. By using appearance information they are, furthermore, able to handle occlusions.

Coifman *et al.* [4] tackle the problem of single loop vehicle reidentification in order to reliably deviate the traffic flow as well as travel time data. Oh *et al.* [5] use a Bayesian approach to identify vehicles in freeway traffic. The probability of identity is derived from physical observations and events, *i.e.* trajectories of vehicles, and can be improved online. Additionally, they are able to derive appearance probabilities for each object.

Huang and Russel [6] use a probabilistic approach to reidentify vehicles on a traffic highway. Color appearance and transition times are modeled using Gaussian distributions. Kettnaker and Zabith [7] use a Bayesian formalization to track persons over multiple non-overlapping cameras. Yet, their system has to be calibrated and the number of possible objects has to be known.

Shan *et al.* in [8] present a system capable of reidentifying cars between two non-overlapping cameras formulated as same-different classification problem without direct feature matching. Their system, however, builds on a SVM classifier which has to be trained and uses edge-based object matching which does not achieve comparable discrimination rates compared to, e.g, DoG or MSER based approaches.

Sun *et al.* [9] perform vehicle reidentification using a multidetector fusion approach. While detection is performed using a nearest neighbor classifier and a linear fusion strategy the features are based on object color and inductive loop information.

Most former mentioned work is based on global appearance based methods while recent works have successfully demonstrated that local appearance approaches achieve high recognition results [10, 11]. However, due to the high computational effort and low image quality until now the use of local-features for object reidentification has been mostly avoided, a forteriori for resource constrained embedded systems.

Our proposed system differs from the former mentioned in (i) that the entire system is designed and implemented in order to run on resource constrained embedded systems, (ii) we successfully and solely employ local features (PCA-Sift) for object reidentification and (iii) concentrate our work on minimizing communication costs rather than local processing (iv) we avoid the necessity of tedious learning and (v) our approach works on spatially separated, uncalibrated, non-overlapping cameras (vi), and finally, no global optimization has to be performed.

3. OBJECT FINGERPRINTING AND REACQUISITION

As former mentioned, the entire object fingerprinting and reidentification approach presented in this paper is solely based on local appearance features. Although the usage of local features has some shortcomings, *i.e.* computational costs, minimum requirements on image quality, etc., and highly relies on robust matching methods, it has the advantage of being partially robust to image scale, rotation, affine distortion, addition of noise and illumination changes. By using a setup such as the proposed one, it is not necessary to tediously train classifiers for each camera or add information of additional cues or sensors. In the following, we shortly introduce our choice of local features, describe their organization in a hierarchical tree-like structure and describe the matching algorithm.

3.1. Local Features

In our approach we use David Lowe's famous *Difference of Gaussian* (DoG) detector to obtain rather accurate keypoints with high repeatability [10] as they have proven to achieve excellent repeatability and recall performance [12].

In short, the DoG-detector takes the differences of Gaussian blurred images as an approximation of the scale normalized Laplacian and uses the local maxima of the responses in scale space as an indicator for a keypoint. The DoG-detector mainly delivers keypoints which indicate the presence of blob-like (more less circular) structures in images.

A nice feature of the DoG detector is that it is almost purely based on image filtering and addition/subtraction operations. While a clever arrangement of filtering and search operations makes the algorithm also efficient in terms of memory usage, the algorithm is very well suited for DSP platforms, as they are mainly designed for fast filter operations.

Moreover, the filtering can be implemented in fixed-point which results in a significant performance increase.

Because the algorithm delivers keypoints with a given scale factor, for the calculation of a descriptor a circular region around the keypoint is cropped, whose size is dependent on the keypoint scale and its orientation depends on the major gradient orientation around the keypoint. Ke and Sukthankar [13] proposed to extract a compact representation, the so called PCA-SIFT descriptor. They have calculated a PCA eigenspace on the gradient images of a representative number of over 20000 image patches. The descriptor of a given patch is generated by projecting the gradients of the tile onto the precalculated eigenspace keeping only the d most significant eigenvectors.

This descriptor has several advantages, especially for our utilization. First, the descriptor is very compact, because Ke and Sukthankar have proven the $d = 36$ dimensional descriptor to exhibit the same discriminatory power as the original 128-dimensional SIFT descriptor of Lowe. A second big advantage is, that a further decrement of d results in only a slight loss in discriminatory power, thereby making the descriptor calculation itself scalable.

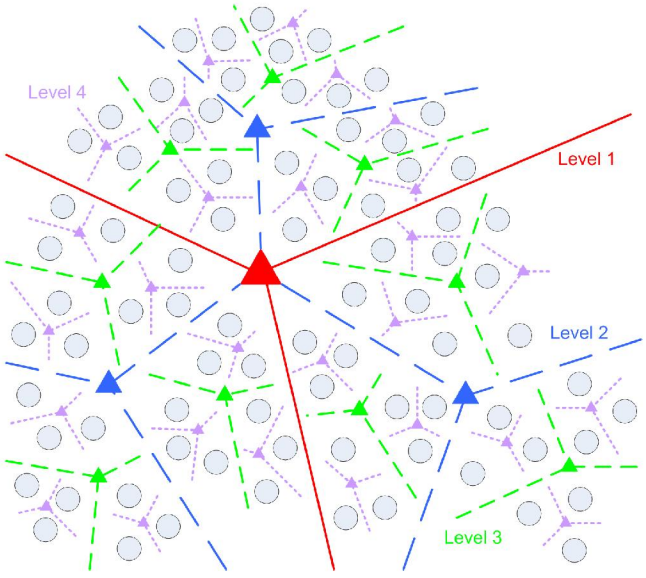

Fig. 1. Schematical illustration of hierarchical k-means clustering. Here, clustering is performed for $k = 3$ and levels 1 to 4, until all points in feature space remain as individual leaves.

3.2. Building the Vocabulary Tree

Recently encouraging recognition results have been achieved in the area of large-scale image databases, using tree-like representations of local descriptors [14, 15]. One approach is the usage of vocabulary trees and inverse voting, like the work of Nistér and Stewenius [15]. For our framework we also utilize the advantages of this basic principle to efficiently build a compact object fingerprint.

First we extract a large number m PCA-SIFT descriptors from training images containing instances of the object category to be dealt with. It is essential that the object domain is well covered by the sample images to guarantee for good performance. In other words, the descriptors extracted from training images should nicely populate the d-dimensional feature space to best possibly fit to the features extracted during testing. The large set of descriptors is quantized in a repeated k-means clustering with a fixed k down the levels of the hierarchical tree. At every node the set of descriptor vectors clustered by k-means is partitioned in k nodes and propagated to the next level until no further splitting is possible. The number of visual words which can be represented is k^L, where k is the branch factor and L is the deepest level of the tree. Finally, the output of the repeated clustering is our vocabulary tree containing m leaves. The clustering and tree building process is illustrated in figure 1.

The amount of data for storing and maintaining the tree is mainly dependent on the descriptor dimension d, the branching factor k and the number of descriptors m used for training. While a small descriptor dimension d is always preferable, the overall size of the tree can be massively reduced by collapsing multiple leaves at the end of branches into single nodes. This can be done either by removing leaves which are too close together or by simply cutting off branches from a predefined level. Needless to say that pruning is equivalent to making a dense sampling in feature space a little sparser. The vocabulary tree has to be built only once and is universally used throughout a given camera network.

3.3. Object Signature and Signature Matching

Whether leaves were collapsed or not, the tree contains a fixed number l leaves which are numbered in increasing order and (hopefully) populate the feature space as desired. For generating an object signature, the keypoints are detected on the object image and corresponding descriptors are calculated. For each of the t descriptors extracted, the nearest neighbor in feature space is determined by traversing the vocabulary tree from the root downwards to the leaves. The unique IDs of the corresponding leaves representing the nearest neighbors are increasingly sorted to form the final object signature (which is in fact a vector of length t numbers). This vector is a maximally compact representation given a fixed vocabulary tree (see figure 2 for illustration).

For matching of individual object signatures, the number of identical elements has to be determined. Though the object signatures are relatively short vectors (typically in the order of a few hundred elements) and can be matched very fast, sorting of the elements in increasing order still simplifies the algorithm complexity and makes more efficient matching possible.

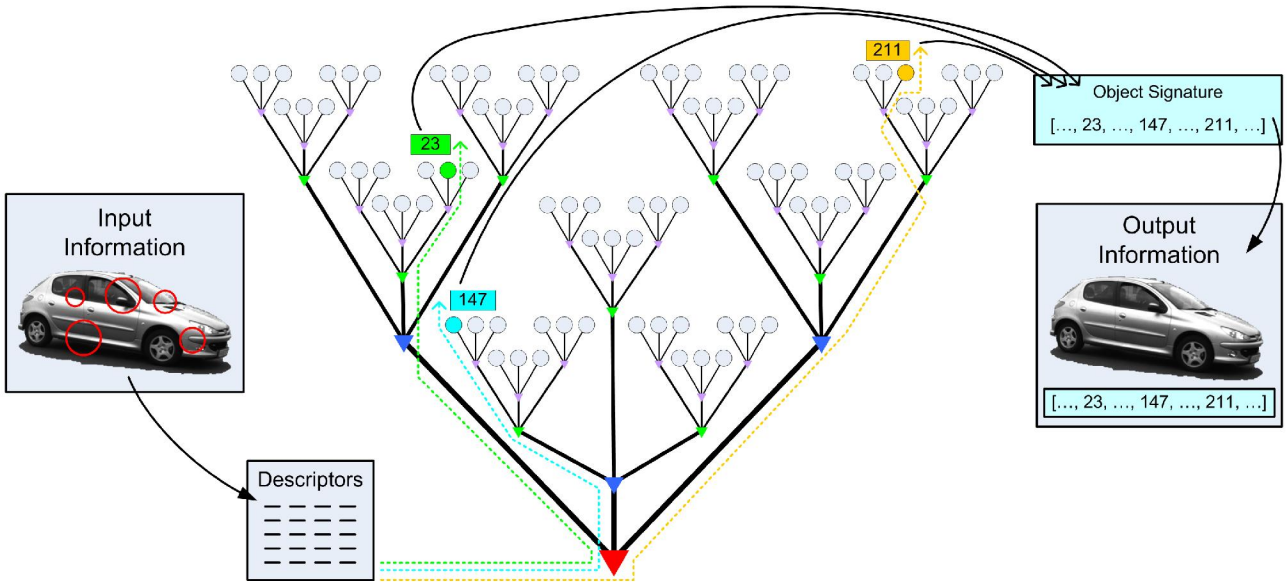

Fig. 2. Illustration of the object signature generation mechanism. After detection of keypoints and descriptor calculation, the leaves representing the nearest neighbors are determined. The unique indices of the leaves are sorted and stored as object signature.

3.4. Removal of Insufficiently Represented Features

Although sample images from the object domain are used for the construction of the vocabulary tree, populating the whole feature space equally well is almost impossible. Especially features from background noise or objects not belonging to the object category often can not be represented well by any leaf of the tree since they populate different areas of the feature space. We can leverage this to early discard features from being included into an object signature. If the Euclidean distance between a feature and its nearest leaf is above a given threshold θ_{dist} we simply treat it as noise and eliminate it. By doing so we can guarantee that only these features are used to build a signature which are not likely to change their affiliation to a specific leaf when being extracted from different object views.

4. CAMERA NETWORK SETUP AND FRAMEWORK OVERVIEW

In this section, we in short describe our target hardware platform and outline the proposed camera network set up. All design considerations are driven by the fact that the entire system has to be used in typical outdoor traffic scenarios as well as under harsh environmental conditions.

4.1. Target Platform

All experiments and simulations were performed to proof our framework applicable on a popular hardware setup for smart cameras. Our platform is similar to the one used by Arth *et al.* [16], hence consisting of a single Texas Instruments^{TM} digital signal processor (TMS320C6414) running at 600MHz and 1MB on chip cache as well as 16MB external SDRAM. The hardware prototype is designed to be applied as a fully integrated compact video server which can be easily integrated in existing analog/digital camera networks.

4.2. Camera Network Setup

In order to ensure an easy and scalable setup, our proposed network architecture is hierarchical organized in camera groups. A group simply consists of one or more single cameras and is defined as a set of neighboring video sensors. Each camera itself is uniquely identified by its group and camera ID, respectively. A complete camera network is temporally synchronized using NTP, constituting the only dependency of our system to some kind of local coordinator. Two types of communication paths are possible, as illustrated in figure 8 showing our multi-camera simulation setup, namely *intra-group* and *inter-group* communication. For each camera, the neighboring cameras are defined as internal neighbors if they are in the same group, or external neighbors if they belong to a different group.

For each new object passing a camera an object signature is created and multicasted together with a timestamp, a preliminary empty history, and the camera and group ID to all neighbors defined. The signature is also stored together with a predefined timeout in a temporary output buffer for later acknowledgment. All receiving cameras store the incoming message in a dedicated buffer. If objects are passing the indi-

vidual node, a signature is created and compared to all available signature messages in the buffer. The matching score is evaluated against a special threshold. If it is lower than the threshold it is considered as "new" (object having entered the scenario in between two neighboring camera nodes), and processing proceeds as described above. If it is higher than the threshold, the car is reidentified, the old signature is exchanged by the new one, the history is updated and the message is multicasted to this cameras neighbors again. Furthermore, the camera which had delivered the signature before is notified that the object has been reidentified.

For overall surveillance purposes two types of notifications to a supervisor might occur. Either a message is created if the object in the temporary output buffer was not acknowledged within the given timeout. This means that an object might has left the scene in between two neighboring cameras. Another message is created if an object is passing a boundary camera node having no more neighbors. Anyway, both types of messages passed to the supervisor contain the complete tracking history of the object making statistical flow analysis on a global level possible.

5. EXPERIMENTS

In this section we evaluate our approach in a real-world traffic scenario. First we describe the way how we collected images for our algorithm evaluations. Then we examine our methods on two different setups and give detailed results. A discussion of the results and further notes conclude the section. Note that all calculations have been simulated under MATLAB^{TM}and finally been examined in the context of our hardware platform.

5.1. Data Acquisition and Database Creation

A well-known problem with the evaluation of algorithms for sensor networks is the lack of training and test data, respectively images in our case. Clearly, collecting data from a multi-node sensor network is difficult due to time synchronization problems. Moreover, the amount of data to be stored and the lack of automatic video annotation tools complicate the creation of a useful database.

We cope with the problem by applying several tricks from the area of computer vision to collect a set of pictures with two cameras only, which we can still use to test our algorithms under maximally realistic conditions for an entire camera network.

Precisely, we recorded 171 vehicles with two different cameras in a roundabout, in order to get maximal viewpoint changes, and then segmented the objects using a simple background modeling. Additionally, we took 29 pictures of many different urban traffic scenarios. Having this basic data setup, we projected the segmented objects to the different backgrounds and varied the ratio between background and foreground ob-

Fig. 3. Projection of various cars onto different backgrounds. We simulate the various levels of background noise by cropping the vehicle out of the images again with a varying border added around the object.

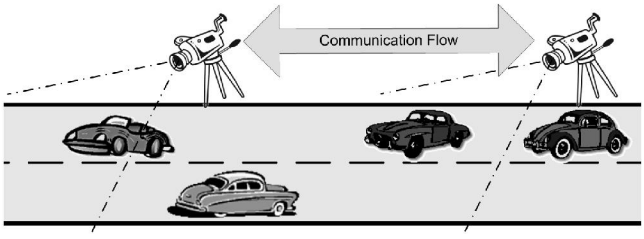

Fig. 4. 2-Camera setup. Vehicle signatures are communicated between the cameras.

ject, thus, allowing for arbitrary reidentification simulation under various amounts of background noise and various viewpoints (see figure 3 for illustration). For simplicity we assume that in a practical setup a preliminary detection result is available. In other words, the algorithm is only applied if an object is present in the image and the major part of an image is covered by only this single object in query. Partial occlusions or parts of other objects are sufficiently well simulated by background noise.

5.2. Two-Camera setup

In our first experiment we simulate a camera network consisting of two single cameras, as depicted in figure 4. For simplicity we assume that vehicles are simply passing the scenario in random order without leaving or entering the scenario in between the cameras. Furthermore, we simulate traffic in one direction of the road only, so cars are not allowed to turn around. Our simulation runs in a 10000 *ticks* long time loop and the vehicles pass the first camera in random order and random intervals. Each vehicle has 500 *ticks* at max to pass the second camera. Thus, the buffer timeout of the cameras is set to 500 *ticks*. To highlight the benefits of our approach we compare it with a reacquisition system based on pure matching of SIFT or PCA-SIFT descriptors.

In figure 5 the dependency of the recognition performance on the additional amount of background noise is depicted. To illustrate the importance of removing inadequately represented features, we have evaluated our approach for both strategies, with and without removal. In the first case the recognition performance significantly drops as the influence of noise on the signature generation increases. In the latter case the recognition performance only slightly decreases. This indicates that a rough object segmentation together with our removal strategy is sufficient to allow for satisfying performance. Note that we have not evaluated our approach for noise levels below 30% because using a simple bounding box around the segmented car already includes at least 25% of background clutter.

In table 1 the amount of data together with the recognition performance achieved for different types of strategies for fully segmented vehicles (no background noise) is summarized. While our strategy already reduces the amount of data to be sent by a dramatic factor, removing noisy features once again cuts the transmission costs by 2/3. As can easily be seen, our setup achieves satisfying results, coevally minimizing the amount of data to be transmitted between individual camera nodes.

In figure 6 the amount of data to be transmitted for various levels of background noise is depicted. While the transmission costs for our tree-based approach only slightly increase, the transmission costs for matching-based approaches explodes as there is no mechanism to early discard bad features.

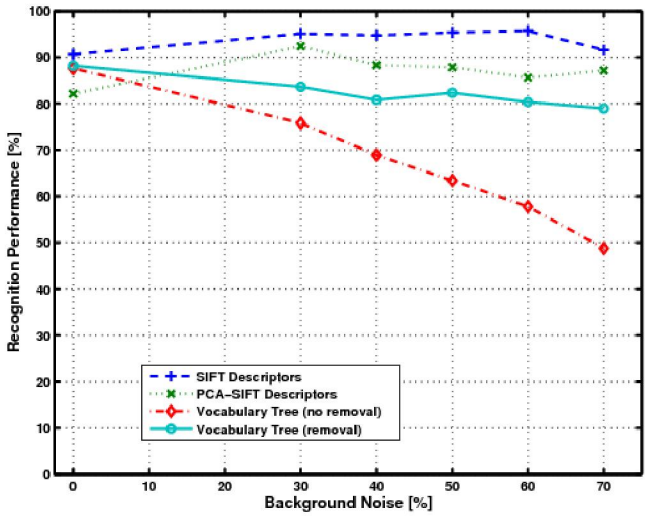

Fig. 5. Recognition performance for different levels of background noise. For example, 50% background noise indicates that the object only covers half of the total image area.

Method	Data to transmit [kB]	Recognition Performance
SIFT keys	19.060,8	90,64 %
PCA-SIFT keys	5.360,8	82,11 %
Object Signatures (no removal)	148,9	87,60 %
Object Signatures (removal)	49,5	88,18 %

Table 1. The amounts of data to be transmitted between individual nodes and the recognition performances for our two-camera setup. We assume that all information is encoded in *integer* or *float* units with 4 bytes each.

Figure 7 shows the amount of information contained in a single transmission unit ([kB]). It is easy to see that our approach nicely compresses the information necessary, while a high amount of data with low information content is transmitted in matching-based approaches.

5.3. Multi-Camera Setup

For this experiment the camera network setup is as depicted in figure 8. The conditions for vehicle movements are the same as described in the previous section, but additionally the paths of the cars through the scenario are determined randomly. For ease of illustration we allowed all cars in either case solely to enter the scenario from one group. However, note that we achieved similar performance rates with arbitrary entry groups.

In figure 9 we have depicted the traffic flows for our setup based on a random traffic setup and with four different entry groups. Though recognition rates in each case depend on the entering group, for all four starting conditions our framework achieved at least *87%* of successful vehicle tracking through

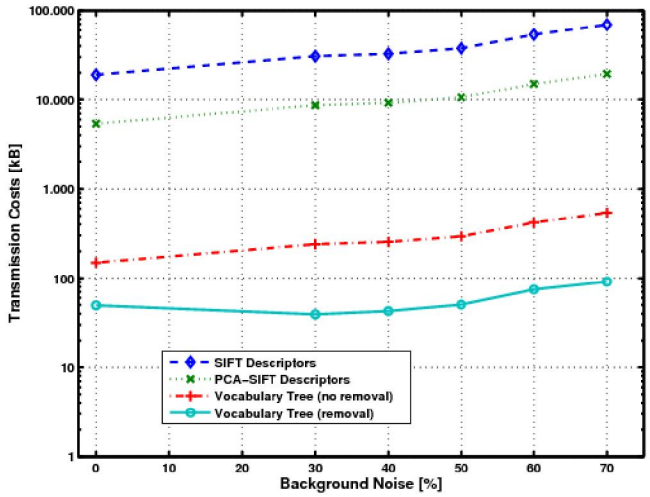

Fig. 6. Transmission Costs for different levels of background noise. As the difference in the amount of data is that severe, logarithmic scaling is chosen.

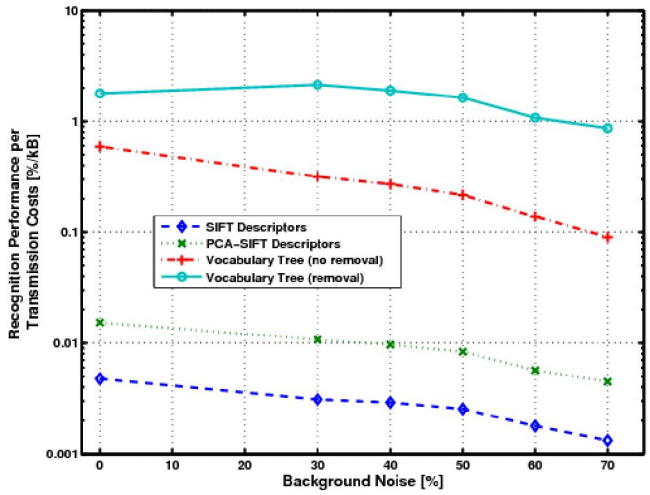

Fig. 7. Information Content for different levels of background noise. While the amount of information contained in a single transmission unit ([kB]) is low in matching-based approaches, it is high in our tree-based approach. Note that our vocabulary tree based method performs more than two powers of ten better than the original approach based on PCA-SIFT or SIFT feature matching.

the entire scenario. Note that, although compared to the two-camera setup one might expect a significant worse recognition rate, the relative high performance comes also from the fact that due to the higher amount of cameras, each camera has to handle a smaller signature message buffer.

5.4. Discussion and Notes

In the previous experiments, we showed that the use of a vocabulary tree is advantageous for this type of application. However, for deployment on our hardware some additional preconditions have to be met.

First we have to make sure that our implementation of the DoG detector and the PCA-SIFT descriptor is real-time capable. Currently, our implementation achieves a little less than 4 frames/second on 352x288 pixel images, the time taken for calculation of keypoints and descriptors is about $270ms$. The time for calculation of a signature using the tree is around $5ms$. This means that about 3 object detections per second can be processed, which is equivalent to a traffic flow of about 10000 cars/hour. Note that we have not fully optimized our implementation, so there is still room for improvement.

An important issue is the amount of memory needed to store a k-means-tree on an embedded device. About 5.1 MB are needed to store a tree with about 43.000 leaves as we used it in our first experiment. The amount of memory to store the temporary buffers is negligible.

Although we have a fully-embedded and working implementation of our approach we have not fully evaluated it yet. The main reason is that our framework is missing some more important features, such as the incorporation of color information. Additional features are indispensable to compensate for errors due to large viewpoint changes. Another shortcoming is that our algorithm uses relatively low image resolution and grayscale images, thus it is not possible to distinguish two similar cars, like cabs, as long as they have no remarkable unique feature, like stickers or paintwork damages. Lastly, to hit performance limits for series-production readiness incorporating license plate information is essential anyway and was not taken into account yet.

6. CONCLUSION AND FUTURE WORK

In this paper we presented a novel approach for object reacquisition and reidentification in networks of embedded smart cameras. The proposed algorithms are designed to run fully autonomous on an embedded development platform and are real-time capable. While communication overhead between adjacent camera nodes is notably reduced, we proof the application of high-level local features advantageous for object fingerprinting. While the application of the approach proposed is not limited to a special object category, our algorithms were tested on traffic scenarios and encouraging results were presented.

In the future we will concentrate our work on the optimization of our algorithms for special types of objects. One of our intentions is to incorporate color information, as color might be one of the most important additional cues for some object domains, *e.g.* vehicles. Furthermore we look forward to also introduce new visual features and other types of simple classifiers into our framework to better fit to the demands difficult object categories make, *e.g.* pedestrians.

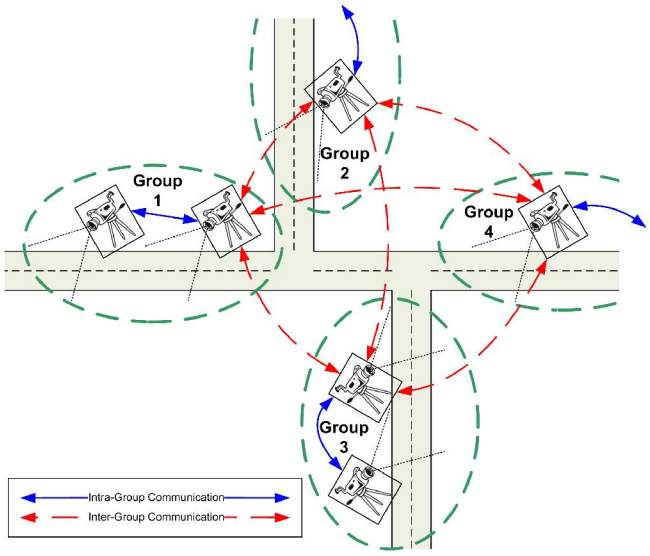

Fig. 8. Multi-Camera setup. Intra-group communication of signatures and inter-group communication occurs between separate camera groups.

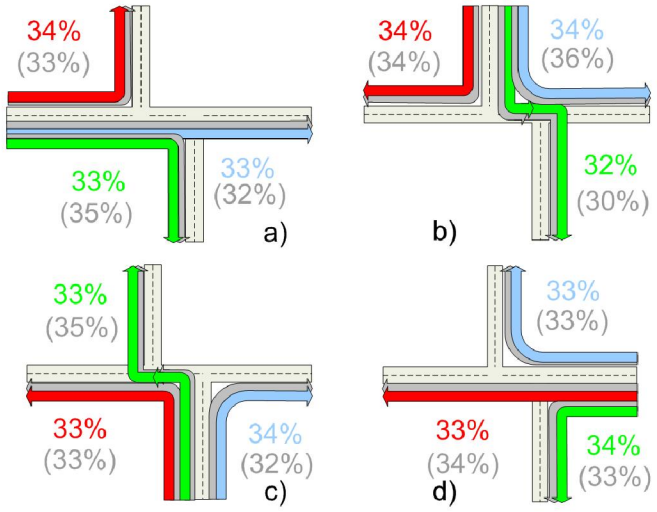

Fig. 9. Four different charts indicating the measured traffic flow in percent. In each illustration a different entry group is chosen. The groundtruth is illustrated in gray.

7. REFERENCES

[1] Yanlin Guo, Steven C. Hsu, Ying Shan, Harpreet S. Sawhney, and Rakesh Kumar, "Vehicle fingerprinting for reacquisition and tracking in videos.," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005, vol. 2, pp. 761–768.

[2] Yanlin Guo, Steven C. Hsu, Ying Shan, Harpreet S. Sawhney, and Rakesh Kumar, "Robust object matching for persistent tracking with heterogeneous features," *Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 29, no. 5, pp. 824–839, 2007.

[3] Saad Ali, Vladimir Reilly, and Mubarak Shah, "Motion and appearance for tracking and re-acquiring targets in aerial videos," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[4] B. Coifman, "Vehicle reidentification and travel time measurement in real-time on freeways using the existing loop detector infrastructure," in *Transportation Research Board*, 1998.

[5] S. Oh, S. Russell, and S. Sastry, "Markov chain monte carlo data association for general multiple-target tracking problems," in *Proc. of the 43rd IEEE Conference on Decision and Control, Paradise Island, Bahamas*, 2004.

[6] Timothy Huang and Stuart J. Russell, "Object identification in a bayesian context," in *Int. Joint Conference on Artificial Intelligence (IJCAI)*, 1997, pp. 1276–1283.

[7] V. Kettnaker and R. Zabih, "Bayesian multi-camera surveillance," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999, pp. 253–259.

[8] Y. Shan, H. S. Sawhney, and R. Kumar, "Vehicle identification between non-overlapping cameras without direct feature matching," in *Int. Conference on Computer Vision (ICCV)*, 2005, vol. 1, pp. 378–385.

[9] C. C. Sun, G. S. Arr, R. P. Ramachandran, and S. G. Ritchie, "Vehicle reidentification using multidetector fusion," in *IEEE Transactions on Intelligent Transportation Systems*, 2004, vol. 5,3, pp. 155–165.

[10] David G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. Journal of Computer Vision (IJCV)*, vol. 60:2, pp. 91–110, 2004.

[11] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool, "A comparison of affine region detectors," *Int. Journal of Computer Vision (IJCV)*, vol. 65, no. 1-2, pp. 43–72, 2005.

[12] Krystian Mikolajczyk and Cordelia Schmid, "A performance evaluation of local descriptors," *Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 27, no. 10, pp. 1615–1630, 2005.

[13] Yan Ke and Rahul Sukthankar, "PCA-SIFT: A more distinctive representation for local image descriptors.," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004, vol. 2, pp. 506–513.

[14] V. Lepetit, Pascal Lagger, and Pascal Fua., "Randomized trees for real-time keypoint recognition.," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005, vol. 2, pp. 775–781.

[15] David Nister and Henrik Stewenius., "Scalable recognition with a vocabulary tree.," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006, vol. 2, pp. 2161–2168.

[16] Clemens Arth, Christian Leistner, and Horst Bischof, "TRICam - An Embedded Platform for Remote Traffic Surveillance," in *CVPRW '06: Proc. of the CVPR Embedded Computer Vision Workshop*, Washington, DC, USA, 2006, p. 125, IEEE Computer Society.