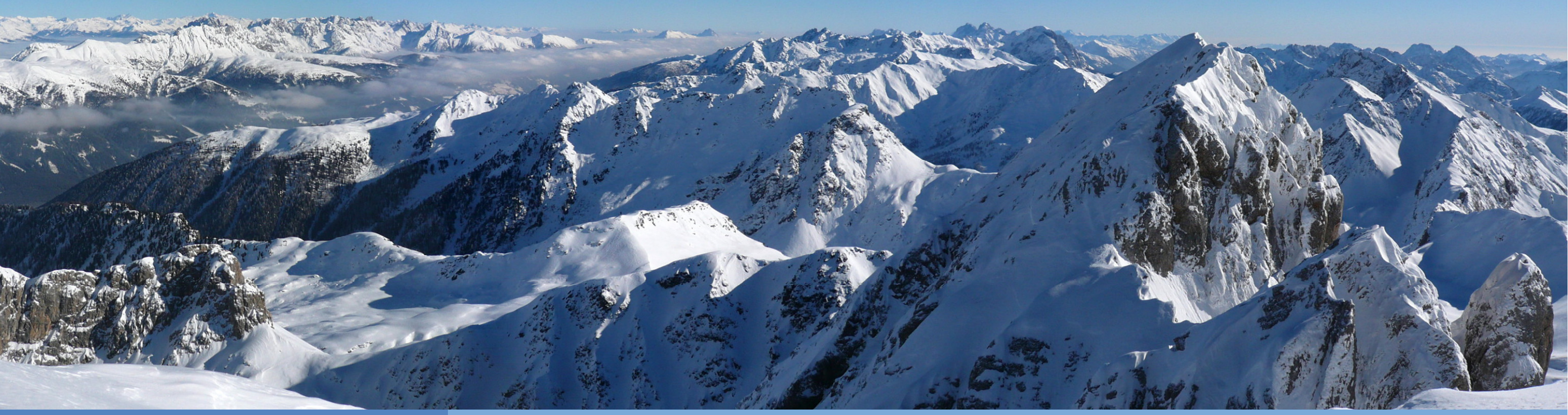
Andreas Wendel
Sabine Sternig
Martin Godec (eds.)

Proceedings of the

16th Computer Vision Winter Workshop



Mitterberg, Austria
February 2 - 4, 2011

Volume Editors

Andreas Wendel
Sabine Sternig
Martin Godec
Graz University of Technology
Institute for Computer Graphics and Vision
Inffeldgasse 16/II, 8010 Graz, Austria
Email: {wendel, sternig, godec}@icg.tugraz.at

Verlag der Technischen Universität Graz
www.ub.tugraz.at/Verlag

Print ISBN
978-3-85125-129-6

e-ISBN
978-3-85125-141-8

DOI
10.3217/978-3-85125-141-8

Bibliographic information published by Die Deutsche Bibliothek.
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available at http://dnb.ddb.de.

Preface

The 16th Computer Vision Winter Workshop (CVWW) was organized by the Institute for Computer Graphics and Vision at Graz University of Technology. It took place from 2nd to 4th of February 2011 in Mitterberg, Austria. The Computer Vision Winter Workshop is the annual meeting of several computer vision research groups located in Graz, Ljubljana, Prague, and Vienna. The basic goal of this workshop is to communicate new ideas within the groups and to provide conference experience to PhD students. In this spirit the topics of the workshop were not explicitly limited to a specific topic but include computer vision, image analysis, pattern recognition, medical imaging, 3D vision, human computer interaction, vision for robotics, as well as applications.

We received 30 paper submissions from six countries. Each paper was reviewed by three members of our international program committee. Among these 30 papers, 24 papers were accepted for presentation at the workshop (18 oral and 6 poster presentations). 12 papers were presented at the workshop but are not published in the proceedings so that no restrictions on submitting the work to other conferences and journals are imposed.

Besides papers selected in the review process, two invited talks were included in the program. We would like to express our thanks to Dr. Jürgen Gall (Swiss Federal Institute of Technology Zürich) and Dr. Christoph Lampert (Institute of Science and Technology Austria). We extend our thanks to the members of the program committee for their time and their detailed and helpful feedback to the authors. We are grateful to Peter M. Roth for providing several hints for organizing this workshop and to our secretary Renate Hönel, who has overtaken many of the "administrative struggles". We also want to thank the sponsors of the workshop for their support: *The Federal Government of Styria* and *Vexcel Imaging - a Microsoft company*.

Andreas Wendel, Sabine Sternig, Martin Godec
CVWW 2011 Workshop Chairs
Graz, Austria, January 2011

Das Land Steiermark
➔ Wissenschaft

VEXCEL IMAGING GmbH
a Microsoft company

Workshop Chairs

Andreas Wendel (Graz University of Technology)
Sabine Sternig (Graz University of Technology)
Martin Godec (Graz University of Technology)

Program Committee

Csaba Beleznai (Austrian Research Centers)
Horst Bischof (Graz University of Technology)
Michael Donoser (Graz University of Technology)
Boris Flach (Czech Technical University Prague)
Friedrich Fraundorfer (Swiss Federal Institute of Technology Zürich)
Yll Haxhimusa (Vienna University of Technology)
Vaclav Hlaváč (Czech Technical University Prague)
Martin Kampel (Vienna University of Technology)
Stanislav Kovacic (University of Ljubljana)
Walter G. Kropatsch (Vienna University of Technology)
Christoph Lampert (Institute of Science and Technology Austria)
Georg Langs (Massachusetts Institute of Technology)
Jiří Matas (Czech Technical University Prague)
Tomáš Pajdla (Czech Technical University Prague)
Franjo Pernus (University of Ljubljana)
Axel Pinz (Graz University of Technology)
Thomas Pock (Graz University of Technology)
Gerhard Reitmayr (Graz University of Technology)
Peter M. Roth (Graz University of Technology)
Matthias Rüther (Graz University of Technology)
Robert Sablatnig (Vienna University of Technology)
Radim Sara (Czech Technical University Prague)
Danijel Skočaj (University of Ljubljana)
Tomáš Svoboda (Czech Technical University Prague)
Martin Urschler (Graz University of Technology)

Contents

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

Structured Learning and Prediction in Computer Vision

Christoph Lampert
Institute of Science and Technology, Klosterneuburg, Austria
chl@ist.ac.at

Abstract. *Powerful statistical models that can be learned efficiently from large amounts of data are currently revolutionizing computer vision. These models possess rich internal structure reflecting task-specific relations and constraints. In my talk I will give an introduction to the most popular classes of structured models in computer vision, concentrating on discrete graphical models and the challenges and opportunities of applying them in a computer vision context. Special emphasis lies on the question how we can efficiently learn the parameters of these models. I will also present examples of successful application of structured prediction techniques to computer vision tasks from my own work and other groups, e.g. in object localization, image segmentation, graph matching and pose estimation.*

16^{th} *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

Vision-based Human Motion Capture: State-of-the-Art and Applications

Jürgen Gall
Computer Vision Laboratory
Swiss Federal Institute of Technology, Zürich, Switzerland
gall@vision.ee.ethz.ch

Abstract. *In this talk, I will present a vision-based human motion capture engine based on interacting simulated annealing and its extensions to various applications like in-house monitoring, video editing, or avatar acquisition. The first extension performs both pose estimation and segmentation. It achieves state-of-the-art results on the HumanEva-II benchmark without imposing restrictions on the dynamics. It can also be used for human motion capture with off-the-shelf handheld video cameras. When shape parameters are estimated in addition to the pose, the shape parameters of the tracked human can be furthermore modified for movie editing. The second extension does not only estimate skeleton motion or the shape of the body, it also estimates detailed time-varying surface geometry. To acquire a realistic avatar from video data, it automatically identifies non-rigidly deforming pieces of apparel and learns a physically-based cloth simulation model for it. Using this approach, real-time animations of humans captured in general apparel can be created. The third extension estimates the human pose and the performed action. To this end, the space of human poses is subdivided into action-specific subspaces. A variant of interacting simulated annealing is used to optimize jointly over the set of subspaces. The approach is promising for monitoring applications where both detailed human pose and performed actions are relevant.*

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

Addressing false alarms and localization inaccuracy in traffic sign detection and recognition[*]

Igor Bonači
Faculty of electrical engineering and computing
Unska 3, Zagreb, Croatia
igor.bonaci@fer.hr

Ivan Kusalić
ivan.kusalic@fer.hr

Ivan Kovaček
ivan.kovacek@fer.hr

Zoran Kalafatić
zoran.kalafatic@fer.hr

Siniša Šegvić
sinisa.segvic@fer.hr

Abstract. *We present a study on applying Viola-Jones detection and SVM classification for recognizing traffic signs in video. Extensive experimentation has shown that this combination suffers from high incidence of false alarms and low tolerance to localization inaccuracy of the true positive detection responses. We report on three improvements which effectively alleviate these problems. Firstly, we confirm the previous result that raw detection performance of Viola-Jones detector can be improved by exploiting color. Additionally, we propose a solution for filtering false positive detection responses, based on a properly trained artificial neural network classifier in the last stage of the detection cascade. Finally, we propose a novel approach for alleviating the degradation of the classification performance due to localization inaccuracy. Experiments have been performed on several video sequences acquired from a moving vehicle, containing several hundred triangular warning signs. The results indicate a dramatic improvement in detection precision, as well as significant improvements in classification performance. At the system level, the proposed system correctly classified more than 97% of triangular warning signs, while producing only a few false alarms in more than 130000 image frames.*

1. Introduction

The ability to detect and classify objects is a key component of many computer vision applications. This paper considers a framework based on combining a boosted Haar cascade detection (the Viola-Jones algorithm) with support vector machine (SVM) classification. Although we address issues of general interest in object detection and recognition, our focus is on studying the considered framework in the context of ideogram-based traffic signs.

There are many exciting application fields of traffic sign recognition in video such as driving assistance systems, automated traffic inventories, and autonomus intelligent vehicles. These applications are important for the society since their main goal is to increase the traffic safety. Consequently, the challenges towards achieving human-like performance (e.g. illumination and color variance or motion blur) are actively researched. Recent high-class car models already come with optional traffic recognition systems, but only limited technical information about the employed algorithms and their performance is available. These recognition systems usually detect only speed limit signs and assume highway conditions, which significantly simplifes the problem.

In early experiments with the proposed framework we experienced two major problems: i) large number of false alarms, and ii) poor classification of the detection responses. This paper reports on several improvements which effectively alleviate these problems. We first report that color sensitive detection can reduce the false positive detection rate while improving the recall for large signs. The false detection rate is additionally reduced by a novel method consisting of adding an artificial neural network classifier as an additional level of a boosted Haar cascade. We present experiments which suggest that the poor classification performance is caused by the localization error in the detection responses. To solve this problem we propose an additional novelty, which is to modify the classifier training set according to the empirically determined properties of the localization error. The presented methods significantly improve the classification performance on standalone images, while the performance in video experiments approaches 100%

[*]This research has been jointly funded by the Croatian National Foundation for Science, Higher Education and Technological Development, and the Institute of Traffic and Communications under programme Partnership in Basic research, project number #04/20. The project web page is at http://www.zemris.fer.hr/ ssegvic/mastif

correct detection and classification.

2. Related work

Automated traffic sign detection and recognition has been an active problem for many years, and there is a vast number of related publications. The detection procedure solves the problem of locating traffic signs in input images, while the classification procedure determines the types of the detected traffic signs.

There are different approaches to detection. Some of the methods [6],[16] use color based segmentation, and model matching in order to detect the traffic sign. There are also researchers that rely only on the shape, using Hough transform [9][7], radial basis transform [11] etc. The other approach is to use a general purpose object detector. A popular algorithm for general object detection has been proposed by Viola and Jones [20]. The algorithm has been applied for traffic sign detection by several researchers [2, 18, 4]. A disadvantage of the original algorithm is that it disregards the color information, which might be valuable for detection. Bahlman et al. [1] use the Viola-Jones detector with extended feature set in order to use color information. That paper reports better detection performance using color, especially in reducing the false positive rate. This result encouraged us to use color information in Viola-Jones detector as well.

Munder and Gavrila [12] compared object detection methods performance on pedestrian classification. Their experiments show that the combination of Support Vector Machines with Local Receptive Field features performs best, while boosted Haar cascades can, however, reach quite competitive results, at a fraction of computational cost. We took advantage of both the Viola-Jones detector speed and the performance of a slower classifier by building a heterogeneous cascade of boosted Haar-like features with Artificial Neural Network as the final level of cascade. This approach significantly lowered the number of false detections.

For the classification task, most of the previous approaches used one of well studied classification schemes, such as SVM [3], multiple discriminant analysis [17], neural networks [14] etc. A detailed report on current research in sign detection can be found in a recently published review paper by Nguwi and Kouzani [13].

3. The Dataset

We used two datasets, labeled as dataset A and dataset B. The dataset A was used for learning and validation, while dataset B was used to test the performance. Both datasets were extracted from video sequences recorded with camera mounted on top of a moving vehicle. Video sequences were recorded at daytime, at different weather conditions. The dataset A corresponds to video material containing about 450 physical triangular warning signs, in which 1802 occurences have been manually annotated. The dataset B contains 265 physical triangular warning signs. Figure 1 shows examples of annotations.

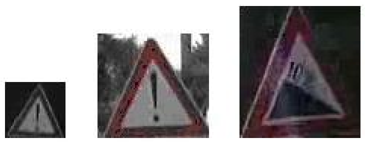
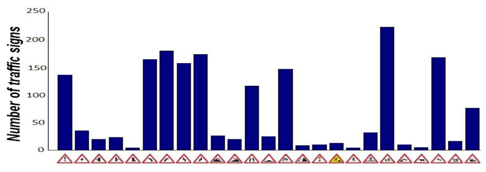
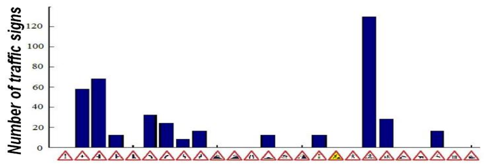


Figure 1. Examples of extracted images

Traffic sign images were annotated manually in video sequences, while background images are extracted randomly from video sequences in dataset A. Altogether, 25 classes of traffic signs are represented in the dataset. Figures 2a and 2b show the distributions of the traffic sign classes present in datasets A and B.



(a)



(b)

Figure 2. Distribution of samples with respect to the sign class for dataset A (2a) and dataset B (2b).

4. Detection

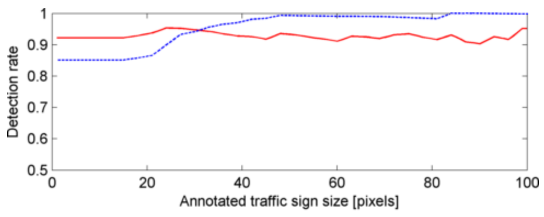
Our detection scheme is based on Viola and Jones' algorithm [20], a very popular method for real-time object detection.

In the next sections we will show the results of a standard Viola-Jones detector on our dataset and the modifications that were made to further improve the detection rate and the false positive rate.

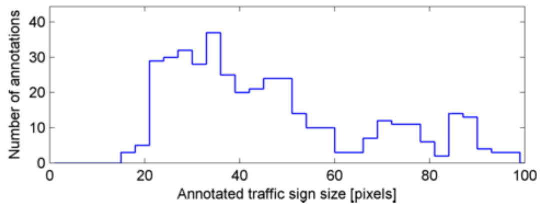
4.1. Viola and Jones' algorithm

Viola-Jones detector uses a cascade of boosted Haar-like features calculated on a gray-scale image.

For a human observer color is of great importance in traffic sign detection, so that by intuition we expect that color information should useful in machine detection as well. Bahlmann et al. [1] suggest computing the Haar-like features from multiple color channels (R, G, B, normalized r, g, b and grayscale).



(a)



(b)

Figure 3. Comparison of the Viola Jones detection with and without color information. 3a shows the detection rates with respect to the traffic sign size. The y-axis represents the detection rate for traffic sign that have an area larger than the value plotted on the x-axis. The dotted blue line corresponds to the color-based cascade, while the solid red line represents the grayscale cascade. 3b shows the distribution of the test dataset with respect to the traffic sign size.

We developed our own implementation of the algorithm which enables us to evaluate the impact of color information to the detection performance. The implementation employs the channels from the Lab color space, with which we obtained best results. Figure 3a compares the detection rates obtained by our color-enabled implementation and the corresponding grayscale version. The y-axis represents the detection rate for traffic signs that have an area larger than the value plotted on the x-axis[1]. The results show that color information has a positive impact when detecting larger traffic signs, but it has a negative impact when detecting small traffic signs (smaller than 30 pixels in size). The reason is that images of distant traffic signs are very small and contain very little color information, while larger images contain enough color (cf. Fig. 1).

In this work, we focus on the detection rate of larger traffic signs because our system will be used with video sequences and we expect that every traffic sign will become large enough for the system to detect. The problem with the system described so far is the false positive rate. When using the Lab cascade we get the false positive rate[2] of 68.7%, while with the grayscale cascade we obtain the false positive rate of 109.24%. Better detection rate for larger traffic signs and smaller false positive rate was the reason for choosing the color cascade. We still need to drastically reduce the false positive rate, because we use the system on video sequences.

[1]Detailed results and parameters used are presented in the results sections.

[2]False positive rate is defined as the number of false detections divided by the number of existing traffic signs.

4.2. Decreasing the false positive rate

In order to reduce the false positive rate we have added an additional stage to the detector cascade. The new stage is a binary classifier based on an artificial neural network[3]. The negative examples for ANN training have been collected as false positives of the Viola-Jones detector applied to the images from the learning dataset A. The positive training images are exactly the same as for the preceding stages of the cascade. The feature set for the neural network is based on the HOG (Histogram of Oriented Gradients) descriptor [5]. Figure 4 shows the arrangement of the HOG cells.

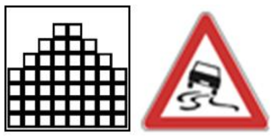


Figure 4. Arrangement of HOG cells in the detection window. The cell size is 6x6 pixels.

The Viola-Jones detector is used because it enables real-time detection, but in order to reduce the false positive rate it is better to use a heterogeneous cascade. Munder et al. [12]report that adding more stages to the VJ cascade further reduces the training set error, but the validation and test sets were observed to run into saturation. Using a stronger and less efficient classifier as the last stage of a VJ classifier does not have a negative impact on detection speed because only a small fraction of image patches passes the VJ cascade.

There are two possible ways of integrating ANN classifier with the Viola-Jones cascade. In the first arrangement the ANN is applied after the integration of multiple detections. That scheme drastically lowers the detection rate because of small errors in localization introduced by the integration of multiple detections. The neural network discards almost all traffic signs which are not aligned perfectly as the annotations used in the learning process. In the second arrangement the ANN is placed before the integration step, which proved to be much more effective. Usually there are several detections of a single traffic sign produced by the Viola-Jones detector, and some of these detections are perfectly aligned. Those detections are accepted by the ANN.

Figure 5 evaluates the impact of using the described combination. It is important to note that the detection rate is lowered only for traffic signs smaller

[3]SVM could be used instead of ANN as they yield almost identical results.

than 45×45 pixels. The false positive rate on a test dataset is reduced from 65.74% to 7.04%.

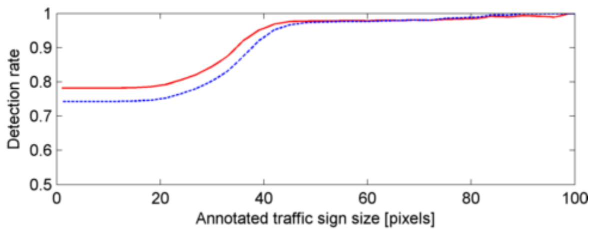


Figure 5. Detection rate with (solid red line) and without (dotted blue line) the ANN stage with respect to traffic sign size in pixels. The y-axis represents the detection rate for traffic signs that have an area larger than the value plotted on the x-axis.

Additionally, it is interesting to note that localization has improved after adding the additional level of cascade. We define localization as the percentage of overlap between an annotated traffic sign and the detection response. Figure 6 evaluates this impact, showing the distribution of traffic sign detections with respect to the localization error. We can see that the ANN stage removes some of the most inaccurately localized detection responses.

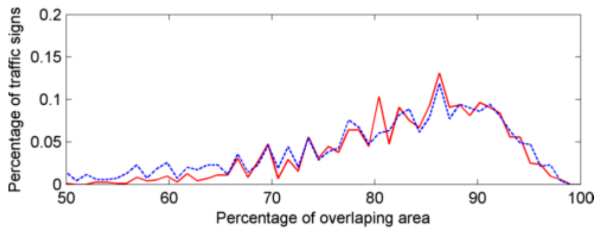


Figure 6. Localization error with (solid red line) and without (dashed blue line) adding the ANN stage. The x-axis represents the percentage of overlap between an annotated traffic sign and the detection response (localization quality).

5. Classification

When a traffic sign is detected, the next step is to determine the class it belongs to. In this section we describe the problems which arise due to localization inaccuracy of the detection responses and propose the solution.

5.1. Feature set and classifier

The first step in solving the classification problem is to choose which features to extract from the resized image patches corresponding to the detection responses. For that purpose we chose HOG descriptors [5] since they performed better than the raw pixels in early experiments with an ANN classifier. Before calculating the HOG descriptor the resized grayscale patches are first resized to 48× pixels, and then contrast-normalized and smoothed with the Gaussian filter. Figure 7 shows arrangement of HOG cells in a resized patch. Figure 7a shows cells of 6×6 pixels, while figure 7b shows cells of 4×4 pixels.

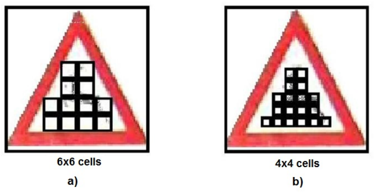


Figure 7. Arrangement of HOG cells over the detection window. Both sets of histograms are used for classification.

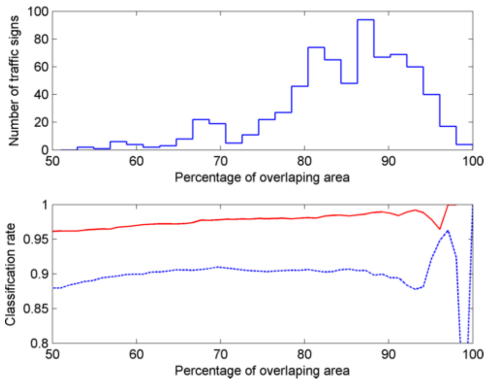


Figure 8. Classification performance of ANN (dotted blue line) and SVM (solid red line) with regard to the percentage of overlapping area between calculated area and annotation, after integration with the detection process. This graph represents the classification rate for all detections that have percentage of overlapping area with annotation larger than value plotted on the x-axis. Classification rate of the SVM classifier is consistently higher than the classification rate of the ANN. The decrease of classification rate at 98% overlap is a result of a single error in classification and therefore falls within the limits of a statistical error. The distribution of traffic signs (top image) is a coarsely discretized distribution from Fig. 6 (solid red line).

For each cell, a histogram of gradient orientations is calculated, and added to the feature vector. For the cells shown in figure 7a histograms have 4 bins and cover $(0, \pi)$ radians, while cells shown in figure 7b have histograms with 7 bins which cover $(0, 2\pi)$ radians. Both sets of cells shown in figure 7 are used in calculation of the feature vector. The dimension of the resulting feature vector is 174.

Having decided on the features that we will use, next we needed to choose a classifier, for which ANN and SVM were considered. After integration of both classifiers with the detection process, results shown in figure 8 were obtained. Dataset B was used as a test set, while the dataset A was used for learning (cf. Fig. 2). The figure clearly shows that SVM performs better then ANN, so that we chose SVM as our classifier.

Initial testing results showed that SVM with HOG

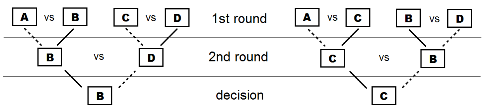


Figure 9. SVM as multi-class classifier. Two DAGSVM trees are shown, both of which use the same binary classifiers (A vs B, B wins; A vs C, C wins; A vs D, A wins; B vs C, C wins; B vs D, B wins; C vs D, D wins). To build DAGSVM tree, all available classes are divided into pairs. On the left side, pairs are: (A,B), (C,D) and on the right, pairs are (A,C) and (B,D). In each round, all pairs are evaluated using binary SVMs, and the winners advance to the next round (solid lines), which are in turn again divided into pairs. Losers are simply discarded (dashed lines). This process continues until only one class remains. Different arrangement of initial pairs can end up with different decisions, as is the case with the illustrated DAGSVM trees.

performs good enough without using kernels. There is no need to use kernels, because the results show that the classes are linearly separable in the feature space.

Because SVM is a binary classifier, we needed to decide on a strategy for using SVM as multi-class classifier. We decided against standard one-vs-one and one-vs-all methods, as they can both produce ambiguity errors in voting process if classes significantly vary in number of training examples. Instead, we used a method similar to the DAGSVM [15, 8]. Our method consists of building a directed acyclic graph which looks like upside-down binary tree with all available classes as leaves. In each step, all classes that are not eliminated are divided into pairs which are then used for one-vs-one binary classification. This way, after each step the number of classes considered is halved, until finally only one class remains. The remaining class is the classifier's decision. Each side of figure 9 illustrates this process.

Because different binary classifiers vary in reliability, this method can produce different results depending on the way classes are initially divided into pairs, as shown on figure 9, where two DAGSVMs make different decisions using the same binary classifiers. Obviously, one of those decisions is wrong, but it is not clear which one. That is why we construct this binary tree a few times (usually 5 times) and employ a simple voting strategy. Each time different separation into pairs is used. Different pairing distributions had little effect, as most of binary SVMs are quite reliable. Nevertheless it did improve classification rate a little, and had no trade offs, as it only consumes slightly more time, which is not of concern in the classification process.

5.2. Modelling the localization error

Aside from relative performances of ANN and SVM, figure 8 shows another interesting phenomena, namely that both classifiers have lower classification rates then we first anticipated. This was at first confusing, as both classifiers performed much better in initial tests that were used to verify validity of implementations, with classification rates around 95% (ANN) and 98% (SVM). We realised that the problem was caused by the localization inaccuracy of the detection responses. Many detections have a small offset, mostly only a pixel or two in each direction. Fig. 10 shows localization error of the detections, while Fig. 11 shows relative scale deviation[4], both with regard to the groundtruth annotations. The presented data was obtained by evaluating the previously described detector on images from dataset A and comparing the detections with annotated locations of traffic signs.

To solve this localization problem, we decided to expand training set with examples that resemble detector's errors, with traffic signs annotated slightly off. Specifically, for each annotated example in training set, we added another 10 examples which model detector's errors. As it can be seen from figures 10 and 11, both types of errors can be modeled with normal distribution. Localization error (expressed relative to the vertical or horizontal sizes of traffic signs) was modeled as normal distribution with parameters ($\mu = -0.014, \sigma = 0.0016$) for x-axis and ($\mu = -0.026, \sigma = 0.002$) for y-axis. Relative scale deviation was modeled as normal distribution, with parameters ($\mu = 1.065, \sigma = 0.074$). The distributions shown on figures 10 and 11 were obtained by comparing detection responses to the annotations from the training set. The SVM classifier trained on the modified training set got the correct classification rate of 95.42%, as opposed to 91.33% obtained with the unmodified training set. Figure 12 shows detailed comparison of results achieved with SVMs trained on different training sets.

The idea of increasing the training set by adding translational jitter has been proposed before, but with different purpose and motivation. For example, Laptev [10] employs this idea to enlarge the training dataset for learning an object detector, while our primary motivation is to improve the recognition performance in presence of localization inaccuracy of the detector responses.

6. System overview

After detection and classification is conducted on images, the next step is to identify the traffic sign through the consecutive images (i.e. video). An output of the Viola-Jones detection is considered a false positive, and thereby is discarded, if a detection is not present in at least three consecutive frames. Group-

[4]Relative scale deviation describes the ratio between detected size and the annotated size.

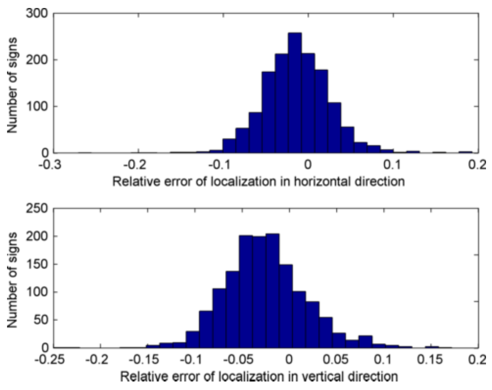


Figure 10. Relative translational deviation of the detection responses with regard to annotation.

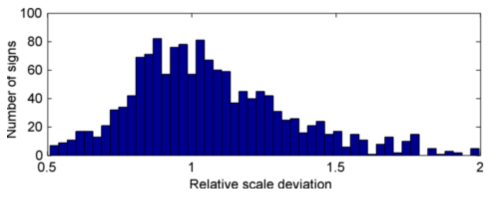


Figure 11. Relative scale deviation of the detection responses with regard to annotation.

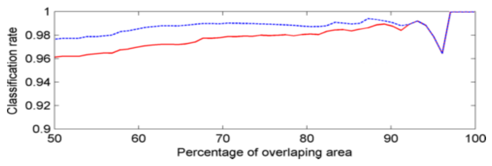


Figure 12. Comparison of classification results achieved with SVMs trained on unmodified (solid red line) and modified (dotted blue line) training sets. This graph represents the classification rate for all detections that have percentage of overlapping area with annotation larger than value plotted on the x-axis. The decrease of classification rate at about 97% overlap is a result of a single error in classification and therefore falls within limits of a statistical error.

ing of single frame detections to a joint detection through the video is based on thresholding the overlapping area between two consecutive detections. For each detection a classification phase is conducted and the final class is determined by voting with equal weights for all detections.

The final process consists of 5 phases:

1. Traffic sign detection with Viola-Jones detector
2. Filtering false detections with ANN classifier
3. Integration of multiple detections
4. Traffic sign classification (SVM)
5. Identifying traffic signs in video

Example of system behaviour is shown in figure 15.

Final system has a frame rate of 15 frames per second with an input image size of 480x360 pixels on Intel 1.8 GHz Dual Core computer. The detection process is implemented to take advantage of multithreading features of a processor.

7. Experimental results

The performed experiments are divided in two categories: results on standalone images and results on video. All experiments were conducted on the same test dataset B corresponding to about 1.5 hours of video material. Detailed information about the test dataset is as follows:

- duration: 1 hour, 28 minutes and 16 seconds
- resolution: 480x360 pixels
- frame rate: 25 fps
- number of frames: 132420
- number of physical traffic signs: 265

For each frame from the video sequence, position of all the traffic signs is given, along with the annotated classification.

7.1. Results on standalone images

We provide results achieved on standalone images first, as the employed core algorithms naturally take an image on input.

Parameters for detection algorithm are as follows:

- Viola-Jones scale factor: 1.2
- Viola-Jones sliding window step size: 5% of current window size
- minimal number of detections needed for confirming the detection: 3

Figure 13 shows achieved detection rates with regard to size of annotation. Total detection rate is 83.53%, which does not look all that impressive at first. Main reason for such low detection rate is the fact that our Viola-Jones implementation uses sliding window with minimal size of 24×24 pixels. If the images with smaller signs are excluded from the test dataset the detection rate increases to 89.18%, which is still too low for practical usage. However, almost all signs larger than 50×50 pixels were successfully detected (99.14%), which gives hope that detections on video would be good enough. The reason for this optimism lies in the fact that the size of a traffic sign increases as video progresses and the vehicle advances closer to the sign.

In experiments in this subsection, the classification was evaluated only on successful detections. Figure 14 shows comparison of SVMs trained on unmodified (dotted blue line) and modified (solid red line) training set (extracted from dataset A). Similarly to the detection results, the total classification rate of SVM trained on modified set is 93.59%, as opposed to 85.14% for SVM trained on unmodified set. It is important to note the importance of dataset modeling according to the localization error.

7.2. Results on video sequence

The results presented in the previous section can be extended to take advantage of multiple occurences

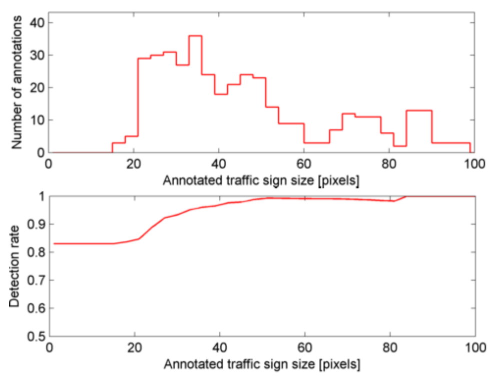


Figure 13. Detection rates with regard to the annotation size in the test dataset. The y-axis represents the detection rate for traffic signs that have an area larger than the value plotted on the x-axis.

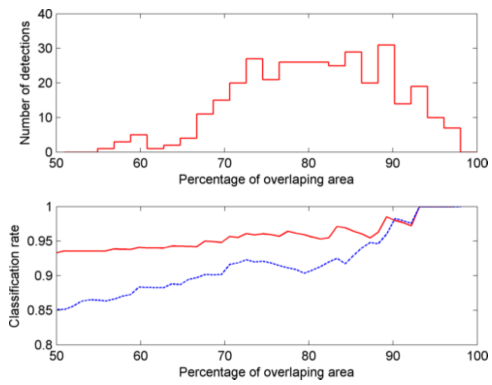


Figure 14. Classification rates with regard to percentage of overlap area between detection and annotation are shown. Solid red line represents classification rate for SVM trained on a modified set, and the dotted blue line represents classification rate for SVM trained on an unmodified set. Graphs represent the classification rate for all detections for which the percentage of overlapping area with the corresponding annotation is larger than value plotted on the x-axis.

Table 1. Final detection results on video sequence.

No. of traffic signs	265
No. of detected signs	260
No. of false detections	2
Detection rate	98.11%
False positive rate	0.75%
False positives per frame	0.0015%

of physical traffic signs in video. The detection results on the test video sequence are given in Table 1[5]. The achieved results are very good, since only smaller traffic signs of poor quality are not detected. This suggests that the detection rate could be increased if the video of higher resolution was used.

There are 243 physical traffic signs that are considered for classification as opposed to 265 total traffic signs present in the video sequence. The 22 missing signs belong to classes for which the classifier has not been trained because dataset A used for training purposes contains insufficient number of training examples from those classes (or none at all). Final results for classification on video sequence are given in table 2. Only two traffic signs were misclassified, both of which are similar to their respective target classes. By combining detection and classification processes, we get overall system performance of 97.3%. Typical results at the system level are illustrated in Fig. 15.

Table 2. Final classification results on video sequence.

No. of traffic signs	243
No. of correct classifications	241
Classification rate	99.17%

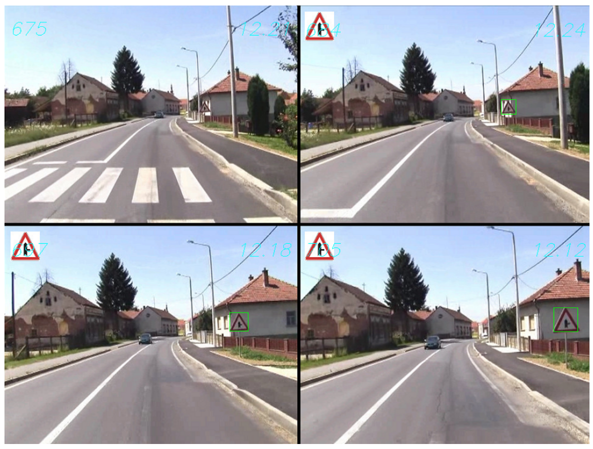


Figure 15. Typical behavior at the system level. In the beginning, the traffic sign is too small, but as time goes on, it becomes big enough and gets detected.

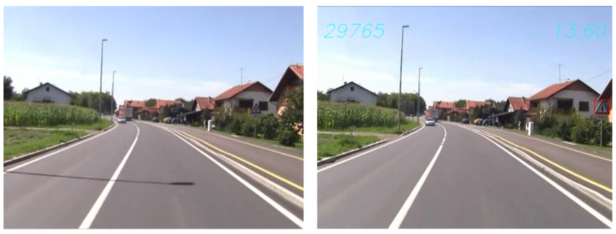


Figure 16. A misdetection of a traffic sign. Size of the traffic sign on the right is 25×25 pixels. Four out of five misdetections at the system level are very similar to this one.

Fig. 16 is an example of a misdetection. The traffic sign in question does not get bigger than 25×25 pixels so that it is detected only in a single frame and is consequently discarded.

8. Conclusion and future work

This paper presents two novel methods that can improve performance of detection and classification either in standalone images or in video. The first

[5] False positive rate on video sequence is defined as the number of false detected traffic signs divided by the total number of traffic signs.

method is used to decrease the false positive detection rate by extending the boosted Haar cascade with an additional stage containing a strong nonlinear binary classifier. The second method adapts the training set to the empirically estimated model of the localization error in the detection responses. Both methods were applied in the frame of a real-time system working on video sequences. The obtained results strongly suggest that automated road inspection is likely to become feasible in the near future.

Some categories of the triangular warning signs are represented with less than 10 samples in the employed training dataset. Thus we believe that it would be possible to obtain even better classification results by collecting a more complete training dataset.

The implemented method for combining information from consecutive frames is very simple, and could be improved in several ways. One of the directions we are currently pursuing is to obtain better detection for small signs by making the ANN detection filter less strict and resolve the remaining false positives with additional approaches. These additional methods would be based either on spatio-temporal properties of the recorded trajectories of the traffic signs, or on enforcing the temporal consistency of the detection responses corresponding to the common physical traffic sign.

We are also interested in expanding the scope of this research to other traffic sign types, such as round traffic signs (mandatory signs). Traffic signs of type C (informational signs) could prove to be especially challenging, as they come in many different shapes and sizes. Using a Viola and Jones' detector for each type of a traffic sign would slow the system considerably. Torralba et al. [19] proposed a method for multiclass object detection, which could be of use in dealing with this diversity without adding much overhead to detection time.

Finally, we are also interested in detecting the state of deterioration of the detected traffic signs. The issues we would like to deal with are fading colors, deformation of the sign pole or the sign itself, inappropriate orientation, or partial occlusion.

References

[1] C. Bahlmann, Y. Zhu, V. Ramesh, M. Pellkofer, and T. Koehler. A system for traffic sign detection, tracking, and recognition using color, shape, and motion information. *Proc. of IEEE IV*, 2005. 2

[2] X. Baro and J. Vitria. Fast traffic sign detection on greyscale images. *Recent Advances in Artificial Intelligence Research and Development*, pages 131–138, October 2005. 2

[3] S. M. Bascon, J. A. Rodriguez, S. L. Arroyo, A. F. Caballero, and F. Lopez-Ferreras. An optimization on pictogram identification for the road-sign recognition task using SVMs. *Computer Vision and Image Understanding*, 2010. 2

[4] K. Brkić, A. Pinz, and S. Šegvić. Traffic sign detection as a component of an automated traffic infrastructure inventory system. In *Proc. of OeAGM/AAPR*, pages 129–140, 2009. 2

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. of CVPR*, pages 886–893, 2005. 3, 4

[6] X. W. Gao, L. Podladchikova, D. Shaposhnikov, K. Hong, and N. Shevtsova. Recognition of traffic signs based on their colour and shape features extracted using human vision models. *J. Vis. Commun. Image R.*, pages 675–685, 2006. 2

[7] M. Garcia-Garrido, M. Sotelo, and E. Martin-Gorostiza. Fast traffic sign detection and recognition under changing lighting conditions. In *Proc. of IEEE ITSC*, pages 811–816, 2006. 2

[8] C.-W. Hsu and C.-J. Lin. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002. 5

[9] W.-J. Kuo and C.-C. Lin. Two-stage road sign detection and recognition. In *Proc. od IEEE ICME*, pages 1427–1430, 2007. 2

[10] I. Laptev. Improving object detection with boosted histograms. *Image Vision Comput.*, 27(5):535–544, 2009. 5

[11] G. Loy and A. Zelinsky. A fast radial symmetry transform for detecting points of interest. In *Proc. of ECCV*, page 358, 2002. 2

[12] S. Munder and D. Gavrila. An experimental study on pedestrian classification. *IEEE Transactions on PAMI*, 28(11):1863–1868, 2006. 2, 3

[13] Y. Nguwi and A. Kouzani. A study on automatic recognition of road signs. In *Proc. IEEE CIS*, pages 1–6, 2006. 2

[14] H. Ohara, I. Nishikawa, S. Miki, and N. Yabuki. Detection and recognition of road signs using simple layered neural networks. In *Proc. of ICONIP vol. 2*, pages 626–630, 2002. 2

[15] J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin dags for multiclass classification. In *Advances in Neural Information Processing Systems*, pages 547–553. MIT Press, 2000. 5

[16] A. Ruta, Y. Li, and X. Liu. Real-time traffic sign recognition from video by class-specific discriminative features. *Pattern Recognition*, 2010. 2

[17] S. Segvić, K. Brkić, Z. Kalafatić, V. Stanisavljević, M. Ševrović, D. Budimir, and I. Dadić. A computer vision assisted geoinformation inventory for traffic infrastructure. In *Proc. of ITSC*, Portugal, 2010. 2

[18] R. Timofte, K. Zimmermann, and L. Van Gool. Multi-view traffic sign detection, recognition, and 3d localisation. In *Proc. of WACV*, pages 69–76, Snowbird, Utah, 2009. 2

[19] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Transactions on PAMI*, 29(5):854–869, 2007. 8

[20] P. Viola and M. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004. 2

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

Estimating hidden parameters for text localization and recognition

Lukáš Neumann
neumalu1@cmp.felk.cvut.cz

Jiří Matas
matas@cmp.felk.cvut.cz

Center for Machine Perception
Department of Cybernetics
Czech Technical University
Prague, Czech Republic

Abstract. *A new method for text line formation for text localization and recognition is proposed. The method exhaustively enumerates short sequences of character regions in order to infer values of hidden text line parameters (such as text direction) and applies the parameters to efficiently limit the search space for longer sequences. The exhaustive enumeration of short sequences is achieved by finding all character region triplets that fulfill constraints of textual content, which keeps the proposed method efficient yet still capable to perform a robust estimation of the hidden parameters in order to correctly initialize the search. The method is applied to character regions which are detected as Maximally Stable Extremal Regions (MSERs).*

The performance of the method is evaluated on the standard ICDAR 2003 dataset, where the method outperforms (precision 0.60*, recall* 0.60*) a previously published method for text line formation of MSERs.*

1. Introduction

Text localization and recognition in images of real-world scenes is still an open problem, which has been receiving significant attention in the last decade [12, 1, 5, 4, 10, 3]. In contrast to text recognition in documents, which is satisfactorily addressed by state-of-the-art OCR systems [6], no efficient method for scene text localization and recognition has been yet published.

Methods for text localization are based on two approaches: sliding windows and connected component analysis. The methods based on sliding windows [2] are more robust to noise, but they have high computational complexity (scanning whole image with windows of multiple sizes is required) and they cannot detect slanted or perspectively distorted text. That is why methods based on individual region detection and subsequent connected component analysis are getting more attention in the text localization community [5, 4, 10]. On the most cited dataset (ICDAR 2003 [8]) the methods based on connected component analysis achieve state-of-the-art results in text localization [11].

In this paper, we present a text line formation method, which groups Maximally Stable Extremal Regions (MSERs) [9] representing characters into text lines. The main contribution of this work is an ability to exhaustively enumerate short sequences of character regions in order to infer values of hidden text line parameters (such as text direction) and subsequently applying the parameters to efficiently limit the search space for longer sequences. The exhaustive enumeration of short sequences is achieved by finding all character region triplets that fulfill constraints of textual content, which keeps the proposed method efficient yet still capable to perform a robust estimation of the hidden parameters in order to correctly initialize the search. The method was evaluated using the hypotheses-verification framework for text localization and recognition published by Neumann and Matas [10], where the heuristic text line formation stage was replaced by the proposed method.

The rest of the document is structured as follows: In Section 2, hidden text line parameters used by the proposed method are defined. Section 3 describes the proposed method for text line formation. Performance evaluation of the proposed method is presented in Section 4. The paper is concluded in Section 5.

2. Hidden text line parameters

It can be observed that text in real-world images follows a certain structure. The structure is not as strict as in the case of text in printed documents, but it is possible to make certain observations at least on the level of individual words; text parameters such as character height, character color, spacing between individual characters have only limited number of distinct values inside a single word. Moreover each word (and possibly more than one word) has an implied direction in which all characters are laid out.

In this paper, we refer to all such parameters as *hidden text line parameters* (or just *hidden parameters*). The initial values of the hidden parameters are obtained by exhaustively enumerating all region triplets and then the inferred values are used to limit the search space during next steps of the text formation. The hidden text line parameters used by the proposed method are height ratio (Section 2.1), centroid angle (Section 2.2) and text direction (Section 2.3).

2.1. Height ratio

The height of two following letters in a word is constrained to a limited interval. In order to express this relation, the height ratio hr between two characters c^1 and c^2 is introduced as

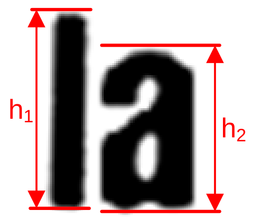
$$\mathsf{hr}(c^1, c^2) = \log \frac{h_1}{h_2} = \log \frac{c_b^1 - c_t^1}{c_b^2 - c_t^2} \qquad (1)$$

where c_t^i and c_b^i denote top and bottom co-ordinate of a bounding box of the character c (see Figure 1a). The measurement is scale invariant, but it is not rotation invariant, which implies that various rotations had to be included in the training set.

Figure 1b depicts the normalized histogram of height ratio values in the training set and their inferred approximation using a Gaussian Mixture Model.

2.2. Centroid angle

Given a sequence of three following letters in a word, the angle between lines connecting their centroids (see Figure 2a) is also constrained to a limited interval. The centroid angle ca of three characters c^1, c^2 and c^3 is defined as



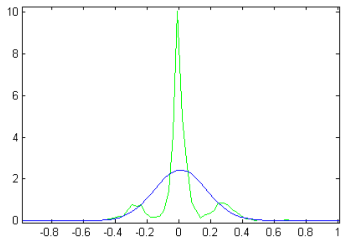
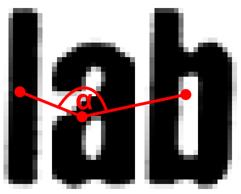

(a) (b)

Figure 1. Height ratio. (a) Measurement example. (b) Normalized histogram (green) and inferred Gaussian Mixture Model M_{hr} (blue)



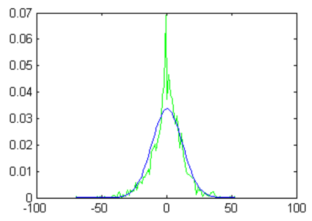

(a) (b)

Figure 2. Centroid angle. (a) Measurement example. (b) Normalized histogram (right, green) and inferred Gaussian Mixture Model M_{ca} (blue)

$$\mathsf{ca}(c^1, c^2, c^3) = \left| \arctan\left(\frac{c_{cy}^1 - c_{cy}^2}{c_{cx}^1 - c_{cx}^2}\right) - \arctan\left(\frac{c_{cy}^2 - c_{cy}^3}{c_{cx}^2 - c_{cx}^3}\right) \right| \qquad (2)$$

where c_{cx}^i (c_{cy}^i) denotes horizontal respectively vertical co-ordinate of a centroid of the character c^i. The measurement is both scale and rotation invariant.

Figure 2b depicts the normalized histogram of centroid angle values in the training set and their inferred approximation using a Gaussian Mixture Model.

2.3. Text direction

The structure of text in real-world images exhibits higher-order properties, which cannot be fully captured by measurements which are defined only using pairs or triplets of individual characters (such as the parameters in Sections 2.1 and 2.2).

In this paper we introduce a set of parameters called *text direction* to capture higher-order structure of text, which exploits an observation that the top and bottom boundaries of individual characters in a word can be fitted by a line. Depending on which letters form the word, each word has either 1 or 2 top lines (see Figure 3), depending whether only upper-case or both upper-case and lower-case letters are present in the word. Let $t_1(x)$ and $t_2(x)$ denote vertical position of first respectively second top line at point x. The same observation applies to the bottom lines where either 1 or 2 lines are present, depending whether underline characters such as "y" or "g" are present or not. Let $b_1(x)$ and $b_2(x)$ again denote vertical position of the bottom lines at point x. *Text direction* T is then defined as quaternion (t_1, t_2, b_1, b_2).

Given a text direction T, *text direction distance* of a character c is defined as

$$d(c,T) = \max\Big(\min(|t_1(c_l) - c_t|, |t_2(c_l) - c_t|), \min(|b_1(c_l) - c_b|, |b_2(c_l) - c_b|)\Big) \quad (3)$$

where c_t, c_l and c_b denote top, left and bottom co-ordinate of a bounding box of the character c.

Mutual position of the lines is not arbitrary either. An assumption was made that these lines are parallel, because height of individual characters in a single word is assumed to be constant and effects caused by perspective distortion in a single word are marginal. Let $D(a(x), b(x)) = |a(x) - b(x)|$ denote vertical distance between lines a and b at horizontal co-ordinate x. Since it was assumed that the lines are parallel, the distance D does not depend on the horizontal position and we can simply write $D(a, b)$ for distance between lines a and b.

In order to express the constraints for mutual vertical distance of the lines, a height of a top bend h_t, a middle bend h_m and a bottom bend h_b is defined (see Figure 3) as

$$h_t(T) = D(t_1, t_2) \quad (4)$$

$$h_m(T) = D(\max(t_1, t_2), \min(b_1, b_2)) \quad (5)$$

$$h_b(T) = D(b_1, b_2) \quad (6)$$

In order to make the text direction parameters scale invariant, they are normalized using a maximal height of a character in the word $h_{\max}$:

$$\bar{d}(c) = \frac{d(c)}{h_{\max}} \quad (7)$$

$$\bar{h}_t(T) = \frac{h_t(T)}{h_{\max}} \quad (8)$$

$$\bar{h}_m(T) = \frac{h_m(T)}{h_{\max}} \quad (9)$$

$$\bar{h}_b(T) = \frac{h_b(T)}{h_{\max}} \quad (10)$$

As shown in Figure 4 the variance of text direction distance $\bar{d}(c)$ measured on the training set is relatively small, which suggests that this parameter can be used as a feature to distinguish between textual and non-textual structures.

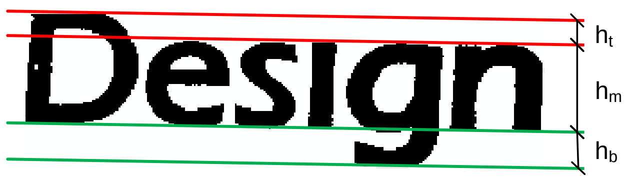


Figure 3. Text direction - top lines (red) and bottom lines (green)

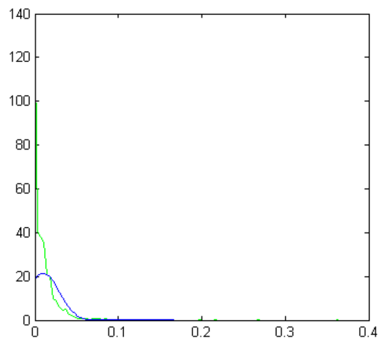


Figure 4. Text direction distance $\bar{d}(c)$ - histogram (green) and inferred Gaussian Mixture Model M_d (blue)

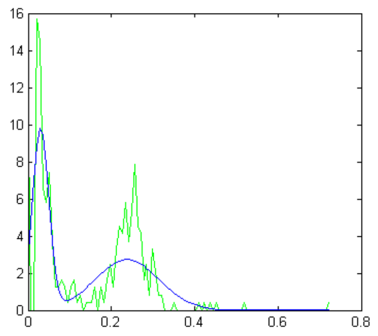


Figure 5. Top band height $\bar{h}_t$ - histogram (green) and inferred Gaussian Mixture Model M_{tb} (blue)

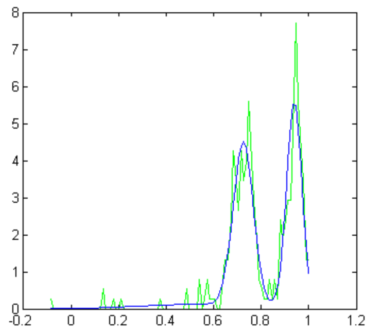


Figure 6. Middle band height $\bar{m}_t$ - histogram (green) and inferred Gaussian Mixture Model M_{mb} (blue)

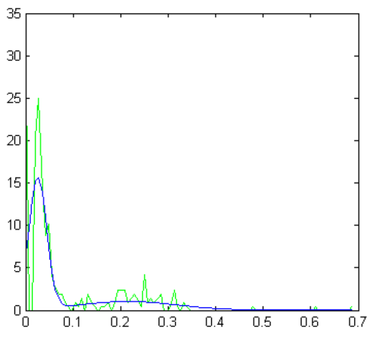


Figure 7. Bottom band height $\bar{b}_t$ - histogram (green) and inferred Gaussian Mixture Model M_{bb} (blue)

```
Procedure la(cc)
 tp := top points of all chars in cc
 bp := bottom points of all chars in cc
 ap := fit bp by a line using Least-Median Squares
 k := tangent of ap

 t1,t2 := fit(tp, k)
 b1,b2 := fit(bp, k)
 T := (t1, t2, b1, b2)
 return T

Procedure fit(points, k)
 bestError := Inf
 for each p,q in points
  line1 := line through p with tangent k
  line2 := line through q with tangent k

  error := 0
  for each r in points
   dist := (min(line1(r[x]),line2(r[x]))-r[y])^2
   error := error + dist

  if error < bestError
   bestError := error
   l1 := line1
   l2 := line2

 return (l1, l2)
```

Figure 8. Pseudo-code of the text direction approximation procedure la(cc)

In order to obtain the text direction T from a sequence of characters $cc = c^1, c^2 \dots c^n$ a procedure la(cc) is introduced (see Figure 8). The example output of the procedure is shown in Figure 9.

3. Text line formation

3.1. Region graph

Individual characters are obtained by detecting Maximally Stable Extremal Regions (MSERs) [9] and then including only the MSERs which are classified as characters using a trained classifier, as proposed by Neumann and Matas [10].

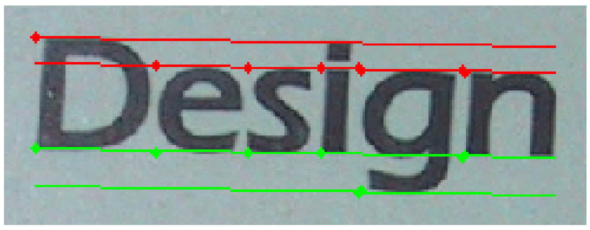


Figure 9. Sequence of characters cc with marked top (red) and bottom (green) points and text direction (top lines - red, bottom lines - green) obtained using the procedure la(cc)

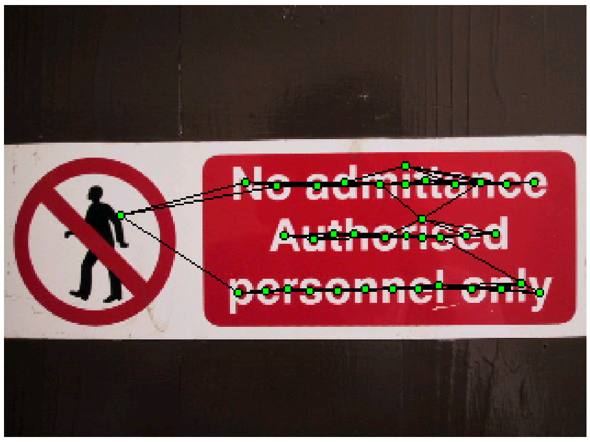


Figure 10. Region graph (initial configuration without any edge labeling)

Let $G = (V, E)$ denote the region graph. The set of vertices V corresponds to the set of character MSERs found in the image. The set of edges E is formed in the following matter: For each vertex, edges to 3 nearest neighboring vertices to the right are created (whilst excluding edges whose centroid angle α is above $40°$). The distance between two vertices is measured as the distance between their centroids. Figure 10 shows an example of such a graph.

3.2. Graph energy

Let $f : E \to \{0, 1\}$ denote a configuration of the region graph G. The text localization task is formulated as finding the best configuration f^* of given graph G such that graph energy $\mathcal{E}(G, f)$ is minimal:

$$f^* = \underset{f}{\operatorname{argmin}}\, \mathcal{E}(G, f) \tag{11}$$

The energy $\mathcal{E}$ is composed of the following weighted components

$$\begin{aligned}\mathcal{E}(G, f) = \alpha_1 \mathcal{E}_{hr}(G, f) + \alpha_2 \mathcal{E}_{ca}(G, f) \\ + \alpha_3 \mathcal{E}_d(G, f) + \alpha_4 \mathcal{E}_{la}(G, f)\end{aligned} \tag{12}$$

where $\mathcal{E}_{hr}$ denotes energy of character height ratios (see Section 2.1), $\mathcal{E}_{ca}$ denotes energy of character centroid angles (see Section 2.2) and $\mathcal{E}_d$ ($\mathcal{E}_{la}$) denotes energy of text direction distances and energy of line approximation respectively (see Section 2.3). Coefficients α_i then denote non-negative weights, which in our setup were all set to 1 in order to give each energy an identical weight. The individual energy components are defined using a Gaussian Mixture Model (GMM) approximation, which was created using the training dataset (as shown in Figures 1, 2, 4, 5, 6 and 7).

Given a Gaussian Mixture Model M obtained from training data

$$f(x) = \sum_{i=1}^{n} \alpha_i \mathcal{N}_{\mu_i,\sigma_i}(x) = \sum_{i=1}^{n} \alpha_i \mathcal{N}_M(x) \qquad (13)$$

the energy $\mathcal{L}_M(x)$ for corresponding model M at point x is defined as

$$\mathcal{L}_M(x) = \min\left\{ \left(\frac{\mu_i - x}{\sigma_i}\right)^2 : i = 1 \dots n \right\} - \theta \qquad (14)$$

where θ denotes a threshold parameter defining what square distance from mean value is considered acceptable. In our setup the value θ was set so that 95% values from training data is accepted.

Let E' denote a subset of edges $\{e \in E \,|\, f(e) = 1\}$ of the graph G and let $C(G, f)$ denote a set of strongly connected components of the graph G when taking into account only edges in E'.

The energy of character height ratios $\mathcal{E}_{hr}(G, f)$ is defined as

$$\mathcal{E}_{hr}(G, f) = \sum_{e \in E'} \mathcal{L}_{M_{hr}}\left(\mathsf{hr}\left(e_b, e_e\right)\right) \qquad (15)$$

where e_b (e_e) denotes a vertex where the edge e begins (ends).

The energy of character centroid angles $\mathcal{E}_{ca}(G, f)$ is defined as

$$\mathcal{E}_{ca}(G, f) = \sum_{\substack{e^1, e^2 \in E' \\ e^1_e = e^2_b}} \mathcal{L}_{M_{ca}}(\mathsf{ca}(e^1_b, e^1_e, e^2_e)) \qquad (16)$$

where again e^i_b (e^i_e) denotes a vertex where the edge e^i begins (ends).

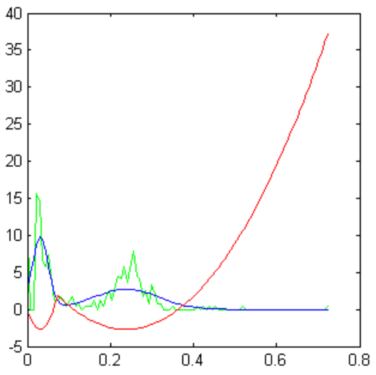


Figure 11. Normalized histogram of training data (green), inferred Gaussian Mixture Model M (blue) and corresponding energy function $\mathcal{L}_M$ (red)

The energy of text direction distances $\mathcal{E}_d(G, f)$ and energy of line approximation $\mathcal{E}_{la}$ are defined as

$$\mathcal{E}_d(G, f) = \sum_{cc \in C(G,f)} \sum_{c \in cc} \mathcal{L}_{M_d}\left(\frac{d(c, \tau)}{h_{\max}}\right) \qquad (17)$$

$$\mathcal{E}_{la}(G, f) = \sum_{cc \in C(G,f)} \max\left\{ \mathcal{L}_{M_{tb}}\left(\frac{h_t(\tau)}{h_{\max}}\right), \mathcal{L}_{M_{mb}}\left(\frac{h_m(\tau)}{h_{\max}}\right), \mathcal{L}_{M_{bb}}\left(\frac{h_b(\tau)}{h_{\max}}\right) \right\} \qquad (18)$$

$$\tau = \mathsf{la}(cc), \; h_{\max} = \max_{c' \in cc}(c'_b - c'_t)$$

3.3. Building region sequences

Region sequences are iteratively built by altering the graph configuration f in order to minimize the energy of the graph $\mathcal{E}(G, f)$. In each step the procedure `test` compares energy of newly created graph configuration f' to the best energy found so far and if a lower energy is found, the current configuration f is updated.

The method starts by enumerating all region triplets, taking only the acceptable triplets (the ones which decrease the graph energy $E(G, f)$) and thus initializing values of text line hidden parameters. Then, single regions are enumerated and the hidden text line parameters are used to efficiently prune the search space. As a last step the method tries to disconnect regions based on the inferred parameters of the whole line of text, because some regions might have been connected in the early stage as a result of inaccurate hidden parameters estimation on short sequences. The process is outlined in Figure 12, a result of the process is shown in Figure 13.

```
Procedure findBestConfiguration (G)
 f := (0,0, ... 0)
 E := 0
 { Connecting triplets of regions to obtain
   initial values of hidden parameters }
 for each subsequent pair of edges e,e' in G
  f' := f
  f'(e, e') = 1
  (E, f) := test(E, f, f')

 { Connecting single regions }
 for each edge e in G
  f' := f
  f'(e) := 1
  (E, f) := test(E, f, f')

 { Trying to disconnect pairs of nodes }
 for each edge e in G
  f' := f
  f'(e) := 0
  (E, f) := test(E, f, f')

 return f

Procedure test(E, f, f')
 E' = calculateEnergy(f')
 if E' < E
  E := E'
  f := f'

 return (E, f)
```

Figure 12. Pseudo-code of finding the best region graph configuration f in the region sequences building process

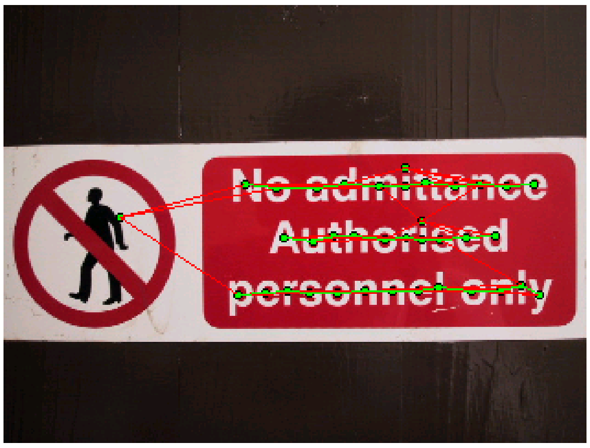


Figure 13. Region graph and its edge labeling corresponding to the best configuration (edges of the graph $f(e) = 1$ marked green, $f(e) = 0$ marked red)

4. Experiments

The method was evaluated using the hypothesis-verification framework proposed by Neumann and Matas [10] and replacing the heuristics text formation stage by the proposed method. The standard and most cited ICDAR 2003 Robust Reading Competition dataset[1] [8] was used for performance evaluation. The Train set was used to obtain the method parameters and an independent Test set was used to evaluate the performance. In total the ICDAR 2003 Test set contains 5370 letters and 1106 words in 249 pictures.

method	precision	recall	f
Pen et. al [11]	0.67	0.71	0.69
Zhang et. al [13]	0.73	0.62	0.67
Epshtein et. al [4]	0.73	0.60	0.66
Hinnerk Becker [7]	0.62	0.67	0.62
proposed method	**0.60**	**0.60**	**0.60**
Alex Chen [7]	0.60	0.60	0.58
Neumann and Matas [10]	0.59	0.55	0.57
Ashida [8]	0.55	0.46	0.50
HWDavid [8]	0.44	0.46	0.45
Wolf [8]	0.30	0.44	0.35
Qiang Zhu [7]	0.33	0.40	0.33
Jisoo Kim [7]	0.22	0.28	0.22
Nobuo Ezaki [7]	0.18	0.36	0.22
Todoran [8]	0.19	0.18	0.18

Table 1. Text localization results on the ICDAR 2003 dataset

Applying the evaluation protocol defined in [8], the proposed method achieved precision of 0.60 and recall of 0.60, which gives f-measure of 0.60. Figure 14 shows examples of text localization and recognition on the ICDAR 2003 dataset.

5. Conclusions

A novel method for text line formation was proposed. The method uses the hidden parameters of the text line (such as text direction) to group Maximally Stable Extremal Regions (MSERs) into lines of text. The exhaustive enumeration of short sequences is achieved by finding all character region triplets that fulfill constraints of textual content, which keeps the proposed method efficient yet still capable to perform a robust estimation of the hidden parameters in order to correctly initialize the search.

The proposed method was evaluated on the standard ICDAR 2003 dataset using the standard evaluation protocol [8], where it outperforms the method for forming text lines of Neumann and Matas [10] (f-measure is increased from 0.57 to 0.60). The method is still behind the state-of-the-art method for text localization (Pen et al. [11], f-measure 0.69), but the text localization results have to be interpreted carefully as there are known problems with the evaluation

[1]http://algoval.essex.ac.uk/icdar/Datasets.html

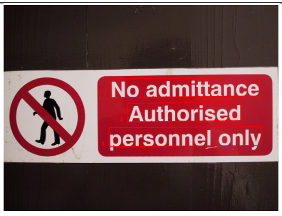
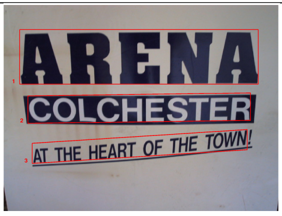
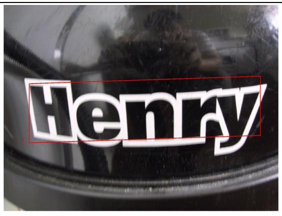
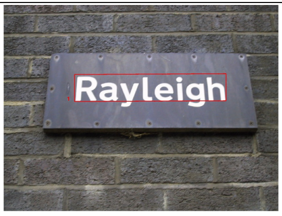
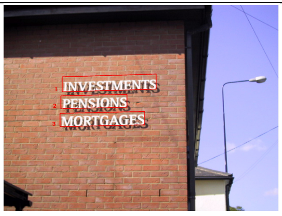
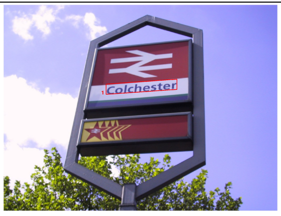
 No admittance Authori8ed personnel only	 ARENA COLCHESTER AT THE HEART OF THE TOWN
 Henry	 Rayleigh
 INVFSTMENTS PENSIONS MORTAGES	 Colchester

Figure 14. Text localization and recognition examples on the ICDAR 2003 dataset.

protocol and ground truth of the ICDAR 2003 dataset [7, 10]. The proposed method aims to solve the complete problem of text detection and recognition (see Figure 14), however all the methods superior in text localization performance [11, 13, 4, 7] aim only to solve one part of the problem and thus direct comparison cannot be made.

Most frequent problems of the proposed method is unsupported text line structure (Figure 15a), symbols or pictographs placed close to text lines (Figure 15b), letters not detected as individual regions (Figure 15c) and false positives caused by repetitive textures with a text-like spacial structure (Figure 15d).

Acknowledgement. The authors were supported by Czech Government reseach program MSM6840770038.

References

[1] X. Chen, J. Yang, J. Zhang, and A. Waibel. Automatic Detection and Recognition of Signs From Natural Scenes. *IEEE Trans. on Image Processing*, 13:87–99, Jan. 2004. 1

[2] X. Chen and A. L. Yuille. Detecting and reading text in natural scenes. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2:366–373, 2004. 1

[3] M. Donoser, H. Bischof, and S. Wagner. Using web search engines to improve text recognition. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1 –4, 2008. 1

[4] B. Epshtein, E. Ofek, and Y. Wexler. Detecting text in natural scenes with stroke width transform. In *CVPR '10: Proc. of the 2010 Conference on Computer Vision and Pattern Recognition*, 2010. 1, 6, 7

[5] N. Ezaki. Text detection from natural scene images: towards a system for visually impaired persons. In *In Int. Conf. on Pattern Recognition*, pages 683–686, 2004. 1

[6] X. Lin. Reliable OCR solution for digital content re-mastering. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, Dec. 2001. 1

[7] S. M. Lucas. Text locating competition results. *Document Analysis and Recognition, International Conference on*, 0:80–85, 2005. 6, 7

[8] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young. Icdar 2003 robust reading competitions. In *ICDAR '03: Proceedings of the Seventh International Conference on Document Analysis and Recognition*, page 682, Washington, DC, USA, 2003. IEEE Computer Society. 1, 6

[9] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable

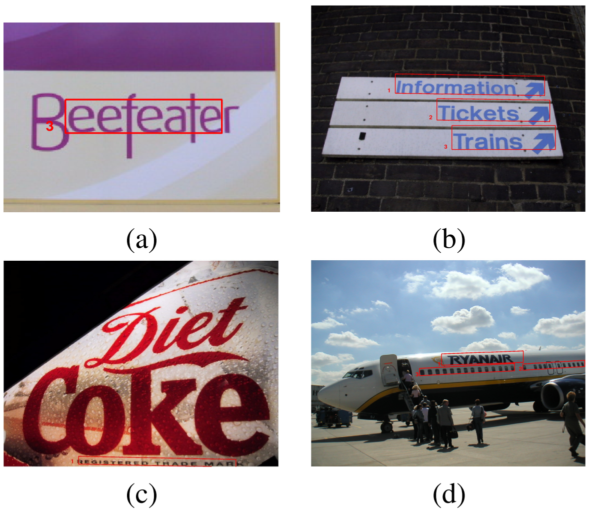


Figure 15. Problems of the proposed method. (a) Unsupported text line structure. (b) Pictographs placed close to text lines. (c) Letters not detected as individual regions. (d) False positives caused by repetitive textures with a text-like spacial structure

extremal regions. *Image and Vision Computing*, 22(10):761–767, September 2004. 1, 4

[10] L. Neumann and J. Matas. A method for text localization and recognition in real-world images. In R. Kimmel, R. Klette, and A. Sugimoto, editors, *ACCV 2010: Proceedings of the 10th Asian Conference on Computer Vision*, volume IV of *LNCS 6495*, pages 2067–2078, Heidelberg, Germany, November 2010. Springer. 1, 4, 6, 7

[11] Y.-F. Pan, X. Hou, and C.-L. Liu. Text localization in natural scene images based on conditional random field. In *ICDAR '09: Proceedings of the 2009 10th International Conference on Document Analysis and Recognition*, pages 6–10, Washington, DC, USA, 2009. IEEE Computer Society. 1, 6, 7

[12] V. Wu, R. Manmatha, and E. M. Riseman, Sr. Textfinder: An automatic system to detect and recognize text in images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(11):1224–1229, 1999. 1

[13] J. Zhang and R. Kasturi. Character energy and link energy-based text extraction in scene images. In R. Kimmel, R. Klette, and A. Sugimoto, editors, *ACCV 2010: Proceedings of the 10th Asian Conference on Computer Vision*, volume II of *LNCS 6495*, pages 832–844, Heidelberg, Germany, November 2010. Springer. 6, 7

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

Non-negative Matrix Factorization in Multimodality Data for Segmentation and Label Prediction

Zeynep Akata
Xerox Research Centre Europe
6, chemin de Maupertuis
38240 Meylan, France
zeynep.akata@xrce.xerox.com

Christian Thurau
Fraunhofer IAIS
Schloss Birlinghoven
53757 Sankt Augustin, Germany
christian.thurau@iais.fraunhofer.de

Christian Bauckhage
Fraunhofer IAIS
Schloss Birlinghoven
53757 Sankt Augustin, Germany
christian.bauckhage@iais.fraunhofer.de

Abstract. *With the increasing availability of annotated multimedia data on the Internet, techniques are in demand that allow for a principled joint processing of different types of data. Multiview learning and multiview clustering attempt to identify latent components in different features spaces in a simultaneous manner. The resulting basis vectors or centroids faithfully represent the different views on the data but are implicitly coupled and they were jointly estimated. This opens new avenues to problems such as label prediction, image retrieval, or semantic grouping. In this paper, we present a new model for multiview clustering that extends traditional non-negative matrix factorization to the joint factorization of different data matrices. Accordingly, the technique provides a new approach to the joint treatment of image parts and attributes. First experiments in image segmentation and multiview clustering of image features and image labels show promising results and indicate that the proposed method offers a common framework for image analysis on different levels of abstraction.*

1. Motivation and Background

The rise of the social web and the user generated content movement have turned the Internet into a virtually limitless repository of annotated and rated multimedia data. For example, as of this writing, there are more than 4.5 billion images available on `flickr` most of which are tagged, rated, categorized, and appraised by the community. This development offers tremendous possibilities for research on image understanding but also calls for methods that allow for an integrated processing of different types of data.

Our goal is a principled joint treatment of image features and image tags. We present a new technique for multiview clustering that simultaneously determines latent dimensions or centroid vectors in different feature spaces. In contrast to ad hoc methods such as, say, concatenating different types of features into a single descriptor, multiview clustering is faithful to the different characteristics of different descriptors. Since latent components or centroids are jointly estimated, multiview techniques allow for advanced inference. Since for every centroid in one feature space there is a corresponding centroid in another space, transitions between different views are straightforward. This offers auspicious new approaches to segmentation, automatic image tagging, or tag-based image retrieval.

Although they have a long and venerable tradition, there is a renewed interest in multiview learning and multiview clustering. The canonical example of a method that simultaneously uncovers latent components in different spaces is Hotelling's canonical correlation analysis (CCA) [12, 2] for which kernelized and probabilistic extension have been proposed as of late [7, 11, 3]. Other recent developments consider extensions of spectral clustering to multiple graphs that encode different types of similarities [27, 21].

Our new approach to multiview clustering extends non-negative matrix factorization (NMF) [17, 16] to the joint factorization of several data matrices. It is motivated by the following considerations:

i) Similar to principal component analysis (PCA) [13] or singular value decomposition (SVD) [9] CCA does not necessarily do justice to purely non-negative data such as color histograms or term frequency vectors. Non-negative matrix factorization, however, typically yields results that can be seen as part-based representations and accommodate human perception.

ii) Methods based on spectral clustering of similarity matrices scale quadratically with the number of data and are therefore prohibitive in modern, large-scale data and image analysis problems.

iii) For NMF, on the other hand, there exist efficient algorithms that factorize matrices of billions of entries [23] which may apply to the multiview setting.

In the next section, we clarify the relation between matrix factorization and clustering. Then, in section 3, we briefly review NMF according to [17, 16] and extend this approach toward the joint factorization of different data matrices. In section 4, we present experiments on using multiview NMF in image segmentation, label prediction, and image retrieval. A conclusion will end this contribution.

2. Matrix Rank Reduction and Clustering

In this section, we briefly review how matrix rank reduction applies to the problem of clustering or vector quantization.

Consider a data matrix $\boldsymbol{X} = [\boldsymbol{x}_1 \dots \boldsymbol{x}_n] \in \mathbb{R}^{m \times n}$ of rank $r \leq \min(m, n)$ whose column vectors $\boldsymbol{x}_i$ correspond to feature vectors obtained from some measurement process. Using the singular value decomposition (SVD) [9] any matrix $\boldsymbol{X} \in \mathbb{R}^{m \times n}$ can be written as

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T = \sum_{j=1}^{r} \sigma_j \boldsymbol{u}_j \boldsymbol{v}_j^T \tag{1}$$

where $\boldsymbol{U} = [\boldsymbol{u}_1 \dots \boldsymbol{u}_m] \in \mathbb{R}^{m \times m}$ and $\boldsymbol{V} = [\boldsymbol{v}_1 \dots \boldsymbol{v}_n] \in \mathbb{R}^{n \times n}$ are orthogonal matrices and $\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_1, \dots, \sigma_r)$. The SVD is a popular tool in data analysis because it is known that the optimal solution to the rank reduction problem

$$\min_{\operatorname{rank}(\tilde{\boldsymbol{X}})=k<r} \left\| \boldsymbol{X} - \tilde{\boldsymbol{X}} \right\|^2 \tag{2}$$

is given by

$$\tilde{\boldsymbol{X}} = \sum_{j=1}^{k} \sigma_j \boldsymbol{u}_j \boldsymbol{v}_j^T = \tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}}\tilde{\boldsymbol{V}}^T. \tag{3}$$

Substituting $\boldsymbol{W} = \tilde{\boldsymbol{U}} \in \mathbb{R}^{m \times k}$ and $\boldsymbol{H} = \tilde{\boldsymbol{\Sigma}}\tilde{\boldsymbol{V}}^T \in \mathbb{R}^{k \times n}$, we recognize that $\boldsymbol{X} \approx \boldsymbol{W}\boldsymbol{H}$ is approximated as a product of a matrix of basis vectors and a matrix of coefficients. This allows for dimensionality reduction, since

$$\boldsymbol{x}_i \approx \sum_{j=1}^{k} \boldsymbol{w}_j h_{ji} \tag{4}$$

so that to every data vector $\boldsymbol{x}_i \in \mathbb{R}^m$ there is a coefficient vector $\boldsymbol{h}_i \in \mathbb{R}^k$ where $k < m$.

Depending on which constraints are imposed on $\boldsymbol{W}$ and $\boldsymbol{H}$, one obtains different dimensionality reduction schemes when solving the general matrix factorization problem

$$\min_{\boldsymbol{W},\boldsymbol{H}} \left\| \boldsymbol{X} - \boldsymbol{W}\boldsymbol{H} \right\|^2. \tag{5}$$

For instance, principal component analysis (PCA) [13] is recovered from

$$\begin{aligned} &\min_{\boldsymbol{W},\boldsymbol{H}} \left\| \boldsymbol{X} - \boldsymbol{W}\boldsymbol{H} \right\|^2 \\ &\quad \text{s.t. } \boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}. \end{aligned} \tag{6}$$

Casting matrix factorization in a yet more general form reveals a connection to vector quantization and clustering. For example, running the k-means algorithm is tantamount to solving

$$\begin{aligned} \min_{\boldsymbol{G},\boldsymbol{H}} \; & \left\| \boldsymbol{X} - \boldsymbol{X}\boldsymbol{G}\boldsymbol{H} \right\|^2 \\ \text{s.t. } & \boldsymbol{g}_j^T \boldsymbol{1} = 1 \\ & \boldsymbol{g}_j \succeq \boldsymbol{0} \\ & \boldsymbol{h}_i = [0 \dots 010 \dots 0]^T. \end{aligned} \tag{7}$$

Due to the convexity constraints on the columns of $\boldsymbol{G}$, the resulting basis vectors in $\boldsymbol{W} = \boldsymbol{X}\boldsymbol{G}$ are convex combinations of certain data points in $\boldsymbol{X}$ and since the coefficient vectors in $\boldsymbol{H}$ are unitary vectors, every data point $\boldsymbol{x}_i$ in $\boldsymbol{X}$ will be represented by exactly one centroid $\boldsymbol{w}_j$ in $\boldsymbol{W}$.

3. NMF for Multiview Clustering

In this section, we first summarize non-negative matrix factorization (NMF) and then introduce our generalization of NMF toward multiview clustering.

3.1. Factorization of Data via NMF

Orthogonal basis vectors such as determined by PCA or SVD are not always the best choice for dimensionality reduction or clustering [17, 16, 25, 6, 15, 14]. In particular data that consist exclusively of non-negative measurements cannot be guaranteed to retain non-negativity after projection onto lower-dimensional subspaces that are spanned by its dominant eigenvectors. As an alternative that is true to the non-negative nature of certain data Lee and Seung popularized the idea of non-negative matrix factorization [17, 16]. In computer vision where image data typically consists of non-negative values, NMF was observed to yield superior results in segmentation, feature extraction, motion-, or pose estimation [26, 10, 4, 22].

Viewed as a constrained least squares optimization problem, NMF amounts to solving

$$\min_{\boldsymbol{W},\boldsymbol{H}} \left\|\boldsymbol{X} - \boldsymbol{W}\boldsymbol{H}\right\|^2 \quad \text{s.t. } \boldsymbol{W},\boldsymbol{H} \succeq \boldsymbol{0}. \tag{8}$$

Although (8) is convex in either $\boldsymbol{W}$ or $\boldsymbol{H}$, the simultaneous estimation of basis vectors and coefficients in (8) does not admit a closed form solution and is known to suffer from many local minima. A unique optimum provably exists [25], however, algorithms that are guaranteed to find it are not known to date (see the discussions in [25, 6, 15, 14]).

In the work presented here, we consider multiplicative fixed point iterations to find a solution to NMF because their extension to multiview clustering is immediate. In the following, $\boldsymbol{A} \odot \boldsymbol{B} \in \mathbb{R}^{m\times n}$ denotes the Hadamard product of two matrices $\boldsymbol{A},\boldsymbol{B} \in \mathbb{R}^{m\times n}$ where $\left(\boldsymbol{A} \odot \boldsymbol{B}\right)_{ij} = a_{ij} \cdot b_{ij}$. The Hadamard division $\oslash$ is defined accordingly but for better readability we write $\boldsymbol{A} \oslash \boldsymbol{B} = \boldsymbol{A}/\boldsymbol{B}$.

Concerned with the problem in (8), Lee and Seung [17, 16] randomly initialize the matrices $\boldsymbol{W}$ and $\boldsymbol{H}$. They derive the following update rules

$$\boldsymbol{W} \leftarrow \boldsymbol{W} \odot \frac{\boldsymbol{X}\boldsymbol{H}^T}{\boldsymbol{W}\boldsymbol{H}\boldsymbol{H}^T} \quad \text{and} \quad \boldsymbol{H} \leftarrow \boldsymbol{H} \odot \frac{\boldsymbol{W}^T\boldsymbol{X}}{\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{H}} \tag{9}$$

and prove their convergence using an expectation maximization argument. Next, we will extend this approach to multiview data.

3.2. Simultaneous Factorization of Multiview Data via NMF

Our main motivation behind the work presented in this paper is to cluster entities for which there are different types of data available. For instance, images retrieved from `flickr` can be characterized by means of different abstract image features but at the same time there are user generated tags or labels available that describe their content or formation. We hypothesize that simultaneous clustering of such different views on the data will yield more meaningful clusters and may provide a tool to fill in missing information. In particular, multiview clustering of image features and image tags may provide a way to predict a set of tags given an image or to retrieve relevant images from a database given a set of query tags.

Assuming a set of n different images, it can be characterized by an $m \times n$ image-feature matrix $\boldsymbol{X}$ as well as by an $l \times n$ term-by-image matrix $\boldsymbol{Y}$. Our basic idea is to uncover suitable bases $\boldsymbol{W}$ and $\boldsymbol{V}$ for the image- and text features, respectively, which are implicitly coupled via a common coefficient matrix $\boldsymbol{H}$. In other words, we aim at finding two low rank approximations

$$\boldsymbol{X} \approx \boldsymbol{W}\boldsymbol{H} \quad \text{and} \quad \boldsymbol{Y} \approx \boldsymbol{V}\boldsymbol{H} \tag{10}$$

where $\boldsymbol{W} \in \mathbb{R}^{m\times k}$, $\boldsymbol{V} \in \mathbb{R}^{l\times k}$, and $\boldsymbol{H} \in \mathbb{R}^{k\times n}$.

Our solution is to formalize this idea as a convex combination of two constrained least squares problems

$$\min_{\boldsymbol{W},\boldsymbol{V},\boldsymbol{H}} (1-\lambda)\left\|\boldsymbol{X} - \boldsymbol{W}\boldsymbol{H}\right\|^2 + \lambda\left\|\boldsymbol{Y} - \boldsymbol{V}\boldsymbol{H}\right\|^2 \quad \text{s.t. } \boldsymbol{W},\boldsymbol{V},\boldsymbol{H} \succeq \boldsymbol{0} \tag{11}$$

where $\lambda \in [0,1]$ is user specified constant that allows for expressing preferences for either of the two feature types. Just as with the original NMF problem in (8), the extended problem in (11) does not admit a closed form solution. We therefore adapt the Lee and Seung type fixed point iteration to our case. For the matrices of basis vectors $\boldsymbol{W}$ and $\boldsymbol{V}$, the update rules immediately carry through and read:

$$\boldsymbol{W} = \boldsymbol{W} \odot \frac{\boldsymbol{X}\boldsymbol{H}^T}{\boldsymbol{W}\boldsymbol{H}\boldsymbol{H}^T} \quad \text{and} \quad \boldsymbol{V} = \boldsymbol{V} \odot \frac{\boldsymbol{Y}\boldsymbol{H}^T}{\boldsymbol{V}\boldsymbol{H}\boldsymbol{H}^T}. \tag{12}$$

Since the coefficient matrix $\boldsymbol{H}$ now couples two bases, its update is slightly more involved. The simplified version of the fixed point iteration for the coefficients is:

$$\boldsymbol{H} = \boldsymbol{H} \odot \frac{(1-\lambda)\boldsymbol{W}^T\boldsymbol{X} + \lambda\boldsymbol{V}^T\boldsymbol{Y}}{\big((1-\lambda)\boldsymbol{W}^T\boldsymbol{W} + \lambda\boldsymbol{V}^T\boldsymbol{V}\big)\boldsymbol{H}}. \quad (13)$$

3.3. Discussion

Our choice of a convex combination of the individual optimization problems in (11) is not an arbitrary decision. There is a known close relation between non-negative matrix factorization and probabilistic latent semantic analysis [8, 5]. Assuming an appropriate normalization, NMF can be understood as learning the parameters of a joint probability distribution which is expressed as a product of marginal distributions. By choosing a convex combination of two NMF problems, this analogy may be lifted to the level of learning a distribution of distributions. This is akin to Latent Dirichlet Allocation [18, 1] but we will leave possible implications to future work.

We note that by setting $\lambda = 0$ or $\lambda = 1$ our model and its updates reduce to the original form of NMF. Moreover, the model is not confined to the case of two different types of views. Its extension to convex combinations of p different views is straightforward:

$$\begin{aligned} \min_{\boldsymbol{W}^i,\boldsymbol{H}} \; & \sum_{i=1}^{p} \lambda_i \big\|\boldsymbol{X}^i - \boldsymbol{W}^i\boldsymbol{H}\big\|^2 \\ \text{s.t. } & \boldsymbol{W}^i, \boldsymbol{H}, \boldsymbol{\lambda} \succeq \boldsymbol{0} \qquad (14) \\ & \boldsymbol{\lambda}^T\boldsymbol{1} = 1 \end{aligned}$$

Finally, as with with all alternating least squares schemes, convergence of the extended update algorithm for multiview NMF is guaranteed. We omit the formal proof but sketch the argument: Given $\boldsymbol{H}$, none of the updates in (12) will increase either term in (11); given $\boldsymbol{W}$ and $\boldsymbol{V}$, the update in (13) cannot increase the expression in (11).

4. Experiments

In the following subsections we present first experimental results obtained from using multiview NMF for image segmentation, label prediction, and image retrieval. Note that, so far, these are preliminary experiments intended to validate the approach. We are currently working on extended experimental evaluations to compare the proposed approach to other methods in the literature.

4.1. Image Segmentation via Joint Non-negative Matrix Factorization

In a first series of experiments, we apply simultaneous NMF to the problem of image segmentation. We consider color images of natural scenes downloaded from `flickr`. We convert the RGB pixel values into the LUV color-space because of its alleged perceptual uniformity which ensures that equally distant colors in the color space would be also equidistant perceptually.

In order to segment an image into homogeneous regions, we sample 1000 pixels from each image and build two feature matrices, one containing 1000 three dimensional column vectors of color information and one containing 1000 two dimensional column vectors containing pixel coordinates. This way, we separate color from location and run simultaneous NMF to obtain centroid vectors $\boldsymbol{W}$ and $\boldsymbol{V}$ in the respective spaces that are coupled via the common coefficients $\boldsymbol{H}$.

We conduct several experiments where we vary the number of centroids $k = \{4, 10, 20\}$ and the weighting parameter $\lambda = \{0.1, 0.5, 0.9\}$. When λ is larger, more weight is given to the color descriptor of the pixels and when it is smaller more weight is given to the location of the pixels. After random initialization to positive values sampled from a Gaussian distribution, we run the update rules for the matrices $\boldsymbol{W}$, $\boldsymbol{V}$ and $\boldsymbol{H}$ until convergence but at most 100 times.

Given the results of the *training* phase, the *test* phase in these experiments consist in assigning every pixel $\boldsymbol{x}$ of an image to one of the k resulting cluster centroids. Given $\boldsymbol{W}$ and $\boldsymbol{V}$, we solve $\min(1-\lambda)\|\boldsymbol{x}-\boldsymbol{W}\boldsymbol{h}\|^2 + \lambda\|\boldsymbol{x}-\boldsymbol{V}\boldsymbol{h}\|^2$ for the coefficients $\boldsymbol{h}$ and determine the cluster index c according to

$$c = \underset{j}{\operatorname{argmax}}\, h_j. \quad (15)$$

Figure 1 shows examples of images we considered in our segmentation experiments. The accuracy of the segmentation appears to improve with an increasing value of the weighting parameter λ. This corresponds to intuition because assigning more weight to color information should yield image segments grouped together based on color rather than on spatial proximity. However the result of segmentation seems best for $\lambda = 0.5$ where location and color values of the pixels contribute equally to the resulting matrix factors. This resembles the behavior of a bi-

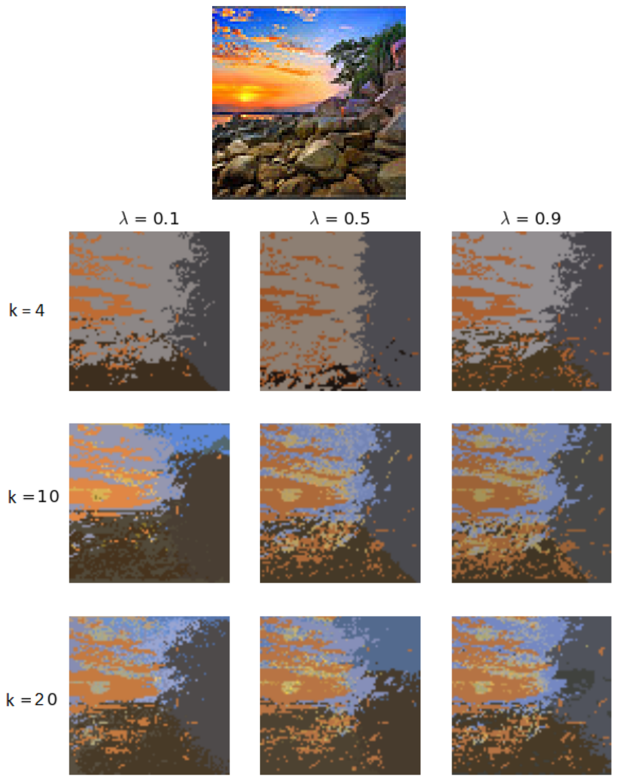


Figure 1. A sample image and its segmentation results obtained from computing cluster centroids using multiview NMF applied to pixel location- and color information. For a smaller λ, more weight will be assigned pixel location information, for a larger λ, more weight will be assigned to pixel color information. With larger weights on location information, small regions of rather homogenous color disappear in the segmentation process. For larger weights on color information, we observe a tendency towards over-segmentation and noisy segment boundaries. For the case where color and location information contribute equally, small regions are preserved and segment boundaries are smoother.

lateral filter [24] which also incorporates color- and location information and is known to yield smooth segment boundaries.

4.2. Label Prediction and Image Retrieval via Joint Factorization of Image- and Text-Features

This series of experiments aims at exploring whether or not multiview NMF is capable of filling in missing information. We considered a training set of natural images retrieved from the "most interesting" category at `flickr`. This set of training images contains 10 different classes (clouds, moonlight, beach, ship, bridge, mountain, forest, city, church, castle) of motives and we considered 300 images per class.

In these experiments, the feature vectors are calculated using local self similarity (SSIM) [20] feature extraction scheme. The feature vectors are then clustered into a visual vocabulary of $k = 750$ visual words. For each image in the dataset, a histogram of this vocabulary is created. The individual histograms of all the images in the dataset are then collected in an image-feature matrix $\boldsymbol{F} \in \mathbb{R}^{k \times n}$.

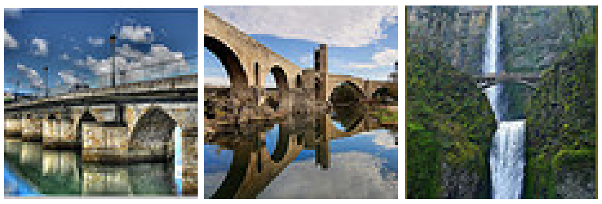
Textual descriptors for the tag list of the images are created by using the well known Bag of Features [19] approach. Firstly, the most frequent tags in the dataset are collected and the textual vocabulary or the dictionary is generated by filtering the irrelevant tags such as foreign names, `flickr` group names, and abbreviations. Secondly, all the tag lists corresponding to the respective images are compared with the dictionary and according to the presence (1) or absence (0) of the dictionary words in the tag list of an image, a binary text feature vector is formed. Finally, the feature vectors are stored in a matrix $\boldsymbol{X} \in \mathbb{R}^{m \times n}$ with n being the number of images in the dataset and $m = 1000$ being the size of the textual dictionary.

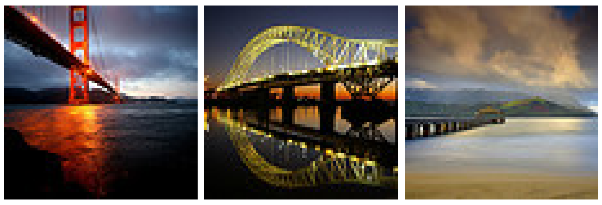
the matrices $\boldsymbol{W}$, $\boldsymbol{V}$ and $\boldsymbol{H}$ were initialized to random positive values sampled from a Gaussian and we ran the multiview NMF update algorithm until convergence but at most 100 times, to obtain coupled factorizations ($k = 10$, $\lambda = 0.5$) of the image- and text-feature matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively. In the test phase of these experiments, we considered two different settings.

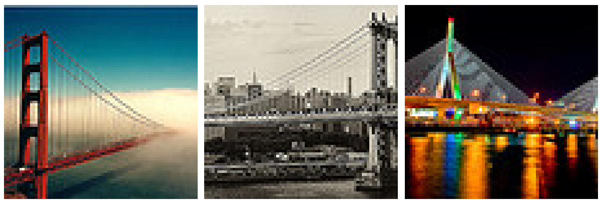
4.2.1 Label Prediction

Given an image that was not part of the training set, we compute its image-feature vector $\boldsymbol{x}$ and solve $\min \|\boldsymbol{x} - \boldsymbol{W}\boldsymbol{h}\|^2 \text{s.t.} \boldsymbol{h} \succeq \boldsymbol{0}$ for $\boldsymbol{h}$. Given $\boldsymbol{h}$, we plug it into $\boldsymbol{y} = \boldsymbol{V}\boldsymbol{h}$ to obtain a corresponding vector $\boldsymbol{y}$ in the text-feature space.

Given $\boldsymbol{y}$, we search for that column vector $\boldsymbol{y}_i$ of the training data matrix $\boldsymbol{Y}$ for which $\|\boldsymbol{y} - \boldsymbol{y}_i\|$ is minimal. We use $\boldsymbol{y}$ to predict a ranked list of tags. To this end, we determine and rank those words in the lexicon that correspond to the 20 basis vectors $\boldsymbol{t}_i$ in the original text-by-image space for which the projection $\boldsymbol{y}^T \boldsymbol{t}_i$ is maximal. The 10 highest ranked tags are selected to be the tag list of the test image. In Figure 1, the retrieved tags for some of the images are shown.


(a) bridge


(b) bridge + sea


(c) bridge + sea + sky

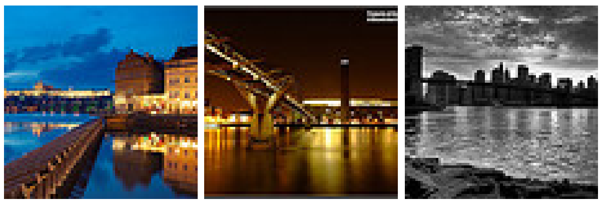

(d) bridge + sea + sky + building

Figure 2. The 3 most relevant images retrieved by querying with the word or the group of words below them. The retrieved images tend to be more specific with the increasing number of words used in the queries.

4.2.2 Image Retrieval

In this setting, we queried random words such as bridge, sea, sky individually or as a group to retrieve the best corresponding images. The text feature vector $\boldsymbol{y}$ of the random words are created the same way as training tag lists of the images. We then solve $\min \|\boldsymbol{y} - \boldsymbol{V}\boldsymbol{h}\|^2$s.t.$\boldsymbol{h} \succeq \boldsymbol{0}$ for $\boldsymbol{h}$. Given $\boldsymbol{h}$, we plug it into $\boldsymbol{x} = \boldsymbol{W}\boldsymbol{h}$ to obtain a corresponding vector $\boldsymbol{x}$ in the image-feature space.

Given $\boldsymbol{x}$, we search for that column vector $\boldsymbol{x}_i$ of the training data matrix $\boldsymbol{X}$ for which $\|\boldsymbol{x} - \boldsymbol{x}_i\|$ is minimal. The four most similiar images are shown in Table 2 that correspond to the words below.

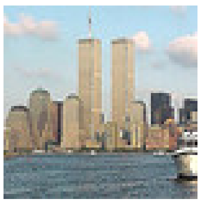
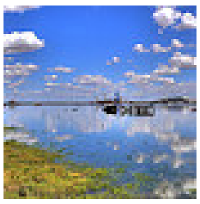
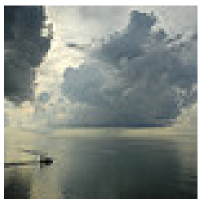
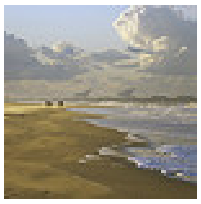
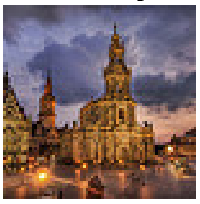
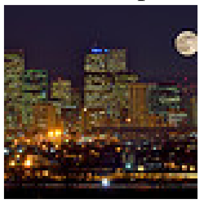
		
high	blue	water
travel	water	sky
cruise	trees	clouds
holiday	bridge	waves
morning	reflections	rocks
cityscape	grass	sea
daybreak	yellow	ocean
tower	woods	seascape
sea	railing	raining
land	waterscape	waterscape
		
water	sky	nightscape
beach	clouds	blue
sunrise	blue	stars
outdoors	holiday	afterdark
nature	red	sky
reflection	castle	night
landscape	bluesky	landscape
raining	raining	moonlight
walking	disneyland	yellow
yellow	middleages	city

Table 1. Results of automatic image annotation. The taglist corresponds to the first ranked 10 tags retrieved by querying an unknown image.

5. Conclusion and Future Work

The work presented in this paper aims at the analysis of images for which there is additional information available. We introduced a new model for multiview clustering that extends the idea of non-negative matrix factorization (NMF) towards the joint analysis of different types of features. We cast multiview NMF as a convex combination of individual optimization problems and adopt the well known multiplicative fixed point algorithm for NMF to this case. The approach avoids ad hoc combinations of different types of features and thus stays true to the nature of different descriptors. The individual optimization problems in our multiview NMF formulation are coupled via a common coefficient matrix. Due to this coupling, the resulting basis vectors or cluster centroids allow for inferring one type of descriptor (e.g. image labels) from another type of descriptor (e.g. image features).

In preliminary experiments we validated the applicability of the proposed approach in image segmentation, tag prediction, and tag-based image retrieval. Our first results suggest that multiview clustering can provide a framework for image analysis that applies to different levels of abstraction. Image parts could be identified by combining pixel-color and -location information in the principal manner that is provided by the multiview approach. Information as diverse as color histograms and text-by-image vectors were coupled using our framework and we found it to be capable to predict missing information from what data was available.

Currently, we are conducting more extensive experiments to provide a more quantitative analysis as well as to compare the proposed approach to other multiview methods such as (kernelized) canonical component analysis. In contrast to related methods from the literature, we expect that highly efficient implementations of multiview NMF will be possible. To this end, we are currently adopting techniques such as convex-hull NMF to our model. We will also further explore how multiview NMF relates to LDA and whether it offers an alternative approach to hierarchical latent topic models. Finally, we envision further applications of the proposed method, for instance in the area of hyperspectral imaging.

References

[1] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet Allocation. *J. of Machine Learning Research*, 3(Jan. 2003):993–1022, 2003. 4

[2] M. Borga, T. Landelius, and H. Knutsson. A Unified Approach to PCA, PLS, MLR and CCA. Technical Report LiTH-ISY-R-1992, ISY, Linköping University, 1997. 1

[3] K. Chaudhuri, S. Kakade, K. Liescu, and K. Shridharan. Multi-View Clustering via Canonical Correlation Analysis. In *Proc. ICML*, 2009. 1

[4] A. Cheriyadat and R. Radke. Non-Negative Matrix Factorization of Partial Track Data for Motion Segmentation. In *Proc. IEEE ICCV*, 2009. 3

[5] C. Ding, T. Li, and W. Peng. NMF and PLSI: Equivalence and a Hybrid Algorithm. In *Proc. ACM SIGIR*, 2006. 4

[6] D. Donoho and V. Stodden. When Does Non-negative Matrix Factorization Give a Correct Decomposition into Parts? In *Proc. NIPS*, 2004. 3

[7] K. Fukumizu, F. Bach, and A. Gretton. Statistical Consistency of Kernel Canonical Correlation Analysis. *J. of Machine Learning Research*, 8(2):361–383, 2007. 1

[8] E. Gaussier and C. Goutte. Relations between PLSA and NMF and Implications. In *Proc. ACM SIGIR*, 2005. 4

[9] G. Golub and J. van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996. 2

[10] D. Guillamet, J. Vitria, and B. Schiele. Introducing a Weighted Non-negative Matrix Factorization for Image Classification. *Pattern Recognition Letters*, 24(14):2447–2454, 2003. 3

[11] D. Hardoon and J. Shaw-Taylor. Convergence Analysis of Kernel Canonical Correlation Analysis: Theory and Practice. *Machine Learning*, 74(1):23–38, 2009. 1

[12] H. Hotelling. Relations Between Two Sets of Variates. *Biometrika*, 28(3–4):321–377, 1936. 1

[13] I. Jolliffe. *Principal Component Analysis*. Springer, 1986. 2

[14] B. Klingenberg, J. Curry, and A. Dougherty. Non-negative Matrix Factorization: Ill-posedness and a Geometric Algorithm. *Pattern Recognition*, 42(5):918–928, 2008. 3

[15] A. Langville, C. Meyer, and R. Albright. Initializations for the Nonnegative Matrix Factorization. In *Proc. ACM KDD*, 2006. 3

[16] D. Lee and H. Seung. Algorithms for Non-negative Matrix Factorization. In *Proc. NIPS*, 2000. 2, 3

[17] D. D. Lee and H. S. Seung. Learning the Parts of Objects by Non-negative Matrix Factorization. *Nature*, 401(6755):788–799, 1999. 2, 3

[18] D. MacKay and L. Peto. A Hierarchical Dirichlet Language Model. *Natural Language Engineering*, 1(3):1–19, 1995. 4

[19] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York u.a., 1983. 5

[20] E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *IEEE Conference on Computer Vision and Pattern Recognition 2007 (CVPR'07)*, June 2007. 5

[21] W. Tang, Z. Lu, and I. Dhillon. Clustering with Multiple Graphs. In *Proc. IEEE ICDM*, 2009. 2

[22] C. Thurau and V. Hlavac. Pose Primitive Based Human Action Recognition in Videos or Still Images. In *Proc. IEEE CVPR*, 2008. 3

[23] C. Thurau, K. Kersting, and C. Bauckhage. Convex Non-Negative Matrix Factorization in the Wild. In *Proc. IEEE ICDM*, 2009. 2

[24] C. Tomasi and R. Manduchi. Bilateral Filtering for Gray and Color Images. In *Proc. IEEE ICCV*, 1998. 5

[25] N. Vasiloglou, A. Gray, and D. Anderson. Non-Negative Matrix Factorization, Convexity and Isometry. In *Proc. SIAM DM*, 2009. 3

[26] R. Zass and A. Shashua. A Unifying Approach to Hard and Probabilistic Clustering. In *Proc. IEEE ICCV*, 2005. 3

[27] D. Zhou and C. Burges. Spectral Clustering and Transductive Learning with Multiple Views. In *Proc. ICML*, 2007. 2

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

Image retrieval by shape-focused sketching of objects

Hayko Riemenschneider, Michael Donoser, Horst Bischof
Institute for Computer Graphics and Vision,
Graz University of Technology, Austria
{hayko, donoser, bischof}@icg.tugraz.at

Abstract. *Content-based image retrieval deals with retrieval in large databases using the actual visual content. In this paper we propose to use hand-drawn object sketches highlighting the outline of an object of interest as query. Due to the lack of appearance, the focus lies on the shape of an object. Such a scenario requires a common representation for the sketch and the images. We propose novel shape-based descriptors that are calculated on local contour fragments. The contour descriptors are stored in a hierarchical data structure, which enables efficient retrieval in sub-linear time, potentially handles millions of images, and does not require retraining when inserting new images. We demonstrate superior performance in this query-by-shape-sketch retrieval for our novel features, and efficient retrieval in 50 milliseconds on a standard single core computer.*

1. Introduction

Image retrieval [5, 14, 15], which deals with the finding of similar images to a given query in large databases, has seen tremendous progress in the last years. Impressive advances were achieved in terms of number of images indexed in the database (up to millions) [5, 19, 23], types of features able to process (color, texture, shape) [11, 12] and most recently also the types of input. The last part deals with what kind of input is provided as query to run image retrieval, for example semantic language based queries, full feature images, or scene and object sketches [7, 13].

In general, mainly three different approaches of how to define the query in a retrieval system can be distinguished. The first group extends standard text retrieval systems relating them to images. The second group considers fully featured images as query, which contain rich scene information in appearance and shape. However, concerning a user guided image retrieval system, such data may not be available, because the user looks for a specific type of image and cannot provide an exemplar image, since this is the actual goal of the search. The third approach uses hand-drawn sketches, showing the desired scene colors or shape of objects, where the visual similarity is defined on a more abstract semantic level.

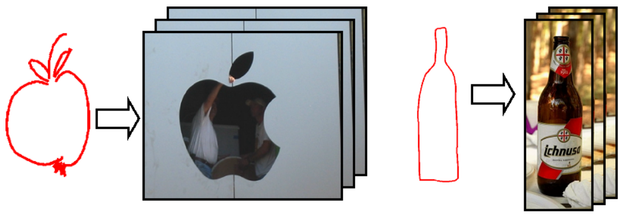


Figure 1. Query by shape sketch image retrieval: Sketching an object outline is the most intuitive user input to support visual image search. Contrary to scene sketching with focus on appearance, the main issue for this novel approach is efficient matching of the shape of an object as well its discrimination to background clutter.

The goal of this paper is to introduce a content-aware image retrieval system, which solely uses a sketch of the outline of an object as query as it is illustrated in Figure 1. This enables a novel intuitive system, where users simply sketch an object of interest on e. g. a tablet PC and immediately retrieve images containing the specified object.

Our content-based image retrieval system is based on a novel feature for describing the local shape of contour fragments and an efficient data structure to retrieve images from large databases in short response time. Inherent properties of our system are the focus on shape, efficient fragment matching considering connectedness of sketch stroke sequences and possible handling of occlusions. We demonstrate how our shape descriptor improves retrieval performance and allows for a content-based image retrieval focused on objects rather than scenes.

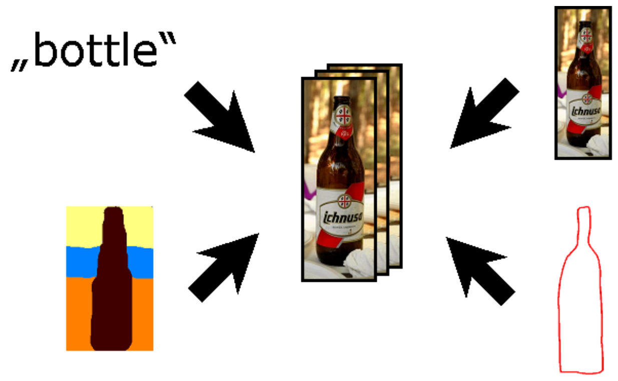


Figure 2. Overview of content-based image retrieval by query type: text, image, scene- and shape-sketch.

2. Related work

Methods for content-based image retrieval can be classified into the following four fields: (a) semantic language-based , (b) image-based, (c) scene sketch-based and (d) object sketch-based, see Figure 2 for an overview. In the following sections, related work in these four fields and properties are discussed.

2.1. Query by text (language-based)

The first field, denoted as *query-by-text*, deals with retrieving images by matching the user input to meta data provided with images, as for example is available on websites or image annotation databases. The features analyzed are not considering image content but rather text describing the content. Powerful scoring functions have been developed for such text retrieval systems to accurately measure the similarity between language data like the well-known term frequency / inverse document frequency (TF/IDF) scheme. Modern search engines such as Bing or Google deal only with semantic queries, whereas for example Cortina [11][1] is a combination of semantic knowledge and image features from the MPEG-7 specification [22] and the SIFT descriptor [17].

2.2. Query by image (full feature images)

In the second field, denoted as *query-by-image*, a single image is provided as query and the most similar images from the database should be obtained. This is for example required for re-localization in 3D reconstruction methods or to identify near duplicate images for copyright protection. The key features are extracted from the full extend of visual information in terms of texture, color and shape.

For example, TinEye[2] creates a unique fingerprint for a complete image (actual technique not revealed) to find the exact matches including crops, editing and resizing. Windsurf[3] retrieves images based on wavelet-indexing of images under region fragmentation. That is, multiple region segmentations are described and used in a one-to-many matching setup [1]. CIRES[4] uses perceptual grouping on low-level edges to obtain a structure of the image, which is a high-level semantic cue for retrieval together with features from Gabor filters and Lab color space [12]. FIDS[5] focuses on efficient retrieval by using color, edge histograms as well as wavelet decomposition [3].

Most of these approaches focus on complete image retrieval, which given the full feature image as input delivers visually similar and even near-duplicate retrieval results. Recent *query-by-image* retrieval systems [5, 14, 18] deal with better scoring strategies and more effective vocabulary construction.

2.3. Query by scene sketch (color drawings)

The third field, denoted as *query-by-scene-sketch*, uses a manual drawing reflecting an image scene by color as query. The user provides a drawing, where complex visual features may not be used because there is simply no data on which to compute them since the sketch is more a cartoon-like drawing.

For example, Retrievr[6] extracts a multi-resolution wavelet fingerprint of the complete image comparing color and shape [13]. The compression to just 20 coefficients allows efficient retrieval.

2.4. Query by shape sketch (line drawings)

The last field of image retrieval systems, which we denote as *query-by-shape-sketch*, uses simple shape sketches as query. The user simply draws a rough outline of an object focusing entirely on the shape as it is illustrated in Figure 1. In our opinion this level of user interaction provides the most natural extension of a language based word-level query, since it enables intuitive systems, where users can simply sketch an object e. g. on a Tablet PC with only a small amount of user interaction required. Previous work in this field uses only histograms or full image similarities to retrieve images containing similar content.

[1]http://vision.ece.ucsb.edu/multimedia/cortina.shtml

[2]http://www.tineye.com

[3]http://www-db.deis.unibo.it/Windsurf/

[4]http://cires.matthewriley.com/

[5]http://www.cs.washington.edu/research/imagedatabase/demo/fids/

[6]http://labs.systemone.at/retrievr/

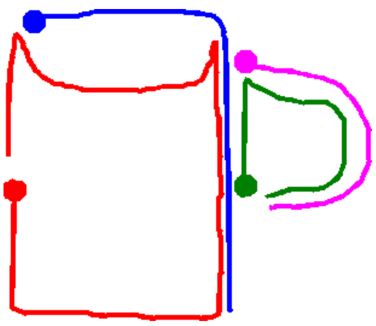


Figure 3. Properties of a user sketch: a) focus on shape, b) multiple strokes (shown in different colors), c) direction of drawing (shown by thicker starting point) and d) sequence information of the connected strokes.

We propose a novel scheme to combine sketches and shape cues using powerful local shape descriptors to retrieve images with similar objects.

3. Image retrieval by shape-sketch

The goal of this work is to enable efficient image retrieval in large databases based on modeling an object of interest using a sketch of the object shape. Hence, we define the term *sketch* as a thin line drawing by a user by means of an electronic pencil. As shown in Figure 3, such a sketch focuses on shape, has no appearance information, may contain one to many line strokes and has defined end points for each stroke (defining a valuable stroke point ordering).

In the following sections we will outline our novel retrieval method that uses such sketches as query to efficiently retrieve images containing the sketched object from potentially large databases. The core idea is to describe both the sketch and the images in terms of a bag of local fragment codewords, where codewords are fragment prototypes (found by comparing fragment shape) that are obtained from edges in the image database. For this we describe local fragments by a powerful shape descriptor that is explained in detail in Section 3.1. In Section 3.2 we describe how a hierarchical data structure denoted as *vocabulary tree* can be used to define our vocabulary of codewords. The vocabulary is built by analyzing the image database, nevertheless once the data structure is built, new images can be inserted without the need of re-training. Finally, in Section 3.3 we show how to use the obtained vocabulary tree in our *query-by-shape-sketch* object retrieval system.

3.1. Local contour fragment description

In a cluttered environment it is important to be able to discriminatively describe shape cues and distinguish them from mere background clutter. Additionally, for a content-based image retrieval system, efficient processing is a vital aspect. For this reason time-consuming learning tasks have to be moved to the offline preprocessing stage. Current state-of-the-art systems based on complex shape features still require a lot of online processing time and are dissimilar in terms of description of object sketches and images, i. e. they do not allow the same description. For this reason we made a thorough analysis of the related work in shape analysis focusing on speed and possible similar description of a binary shape and a full feature image. Possible descriptors include the Edge Histogram Descriptor (EHD) which is specified in the MPEG-7 standard [22], the Shape Context (SC) [2], the Turning Angle (TA), which is a subset of the Beam Angle Histograms (BAH) [20], and the PArtial Contour and Efficient Matching (PACEM) descriptor [21], which is a recent shape descriptor designed for partial matching and encoding of sequence information.

In this work we extend the shape description from [21] to enable efficient content-based image retrieval. As will be described in detail in the experimental section, our new shape descriptor makes *query-by-shape-sketch* feasible and successful because of an immense speedup and a powerful description of the shape of local fragments.

We define the term *contour* as a connected sequence of points, which might come from an edge obtained from an image or from a stroke from the input sketch. Further, a *contour fragment* is a connected subset of a contour. Essential to our description is that all contour fragments are an ordered list of points. Our descriptor is now calculated for such local contour fragments, all having a fixed number of points L and it considers the available ordering of the points. In comparison, the Shape Context (SC) [2] descriptor loses all the ordering information due to the histogram binning. It is further important to note, that the image edges and user strokes may be over-fragmented and broken into multiple contours. Contrary to [21], where partial matching is used to overcome this fragmentation, we simply analyze purely local contour fragments.

Our descriptor is inspired by the chord distribution. A chord is a line joining two points of a region boundary, and the distribution of their lengths and angles was used as shape descriptor before, as for example by Cootes et. al [6] or in the work on

Geometric Hashing [24]. Our descriptor analyzes these chords, but instead of building histograms of their distributions, we use the relative orientations between specifically chosen chords.

Our descriptor is based on angles α_{ij} which describe the relative spatial arrangement of the points $P_1 \dots P_L$ located on the analyzed contour fragment. An angle α_{ij} is calculated between a chord $\overline{P_i P_j}$ from a reference point P_i to another sampled point P_j and a chord $\overline{P_j P_\infty}$ from P_j to P_∞ by

$$\alpha_{ij} = \sphericalangle \left(\overline{P_i P_j}, \overline{P_j P_\infty} \right), \tag{1}$$

where $\sphericalangle(\dots)$ denotes the angle between the two chords and P_∞ is the point at vertical infinity. Thus the angle is calculated between the chord and a vertical line.

In the same manner L different angles $\alpha_{i1} \dots \alpha_{iL}$ can be calculated for one selected reference point P_i. Additionally, each of the sampled points can be chosen as reference point and therefore a $L \times L$ matrix A defined as

$$\mathrm{A} = \begin{pmatrix} \alpha_{11} & \cdots & \alpha_{1L} \\ \vdots & \ddots & \vdots \\ \alpha_{L1} & \cdots & \alpha_{LL} \end{pmatrix} \tag{2}$$

can be used to redundantly describe the entire shape of a fragment with length L. This descriptor matrix is not symmetric because it considers relative orientations. Please note, that such a shape descriptor includes local information (close to the main diagonal) and global information (further away from the diagonal) and it additionally encodes the global orientation of the fragment. In such a way the shape of every contour fragment of length L can be described by an $L \times L$ matrix.

3.2. Fragment vocabulary generation

In general, the goal of a content-based image retrieval system (CBIR) is to provide fast results on a large scale database. Most related work on *query-by-scene-sketch* and *query-by-shape-sketch* focuses on an approximated nearest neighbor search to achieve this. In this work we propose to use hierarchical data structures as introduced in *query-by-image* research to cluster and efficiently search our shape descriptors for defining a visual vocabulary. We apply a data structure known as *vocabulary tree* [19] for our purposes, which exhibits the benefits of data adaption and ability to handle high dimensional features as contrary to nearest neighbor search or kd-trees [10].

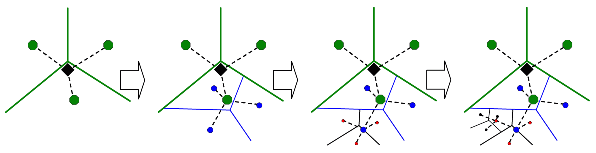


Figure 4. An example vocabulary tree for three cluster centers and a depth of four levels [19]. Each hierarchical level contains a part of the previous data and refines the clustering detail allowing for better data adaptation.

The vocabulary tree [19] is a highly effective way to define the vocabulary for bag-of-word representations and assign query descriptors to the codewords. The approach uses k-means clustering for each level of the tree. This yields a hierarchy of clusters which again is used to efficiently traverse the vocabulary tree and find matching cluster centers.

In its definition a vocabulary tree is a data structure of k cluster centers and a depth of l levels. Figure 4 shows an illustration from Nister and Stewenius of a vocabulary tree built for three cluster centers and a depth of four levels. For each new level the data clustered to the number of centers and divided. A new level of clustering provides more detailed quantization of the descriptors.

The cluster centers are referred to as nodes of the tree and the nodes at the last level are known as leaves. Each of these nodes contains an inverted file list. This list maintains an index to the images whose feature descriptors are included in the respective nodes. So instead of holding the actual descriptors themselves, only a correspondence between best matching node and image identifier is available.

Further each node contains a weight based on entropy. The more images are included in a node the less distinctive it becomes. Nister and Stewenius define various voting strategies for retrieval. First, the *flat strategy* defines a scoring where only the leaf nodes are used. If a descriptor of an image matches to a node in the lowest level, its weight is included in a sum later normalized by the number of descriptors in total. Second, the *hierarchical strategies* define scoring based on how many levels upwards from the leaf level are also considered during scoring. While the second one improves the recognition rate, the *flat scoring* allows much faster retrieval. We adopt this strategy and define the weight w_i of a node as

$$w_i = ln(\frac{N}{n_i}), \tag{3}$$

where the total number of images N in the vocab-

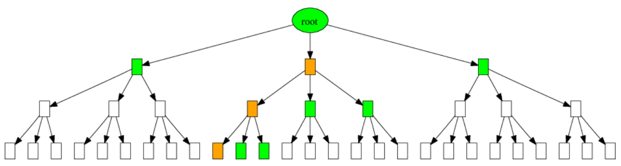


Figure 5. Illustration of the hierarchical traversal through a vocabulary tree: Only the nodes in green are considered at each level and the orange nodes indicate the best matches. These determine the k of k^{l+1} cluster nodes which are considered at the next level $l+1$.

ulary tree and the number of images n_i which are contained in a node i are used as entropy measure.

The final score s is determined by the sum over all nodes where the query descriptor matches this node. The frequency of matches for each descriptor is used and normalized by the total number of descriptors – for the query image and the already known images in the vocabulary tree. The final score is then defined as

$$s = \sum_i \frac{w_i \times q_i \ \times d_i}{Q \times D} \tag{4}$$

where w_i represents the weight of the current node, q_i and d_i the number of times a descriptor for a query or database image passed through the current node, and Q and D are the total number of query or database descriptors respectively.

We use the vocabulary tree to create a general shape vocabulary for an efficient retrieval system. For this, we extract Canny edges [4] from the images of our database using the standard hysteresis thresholding in Matlab and link the results to a set of coordinate lists. This linking is done by analyzing an 8 connected neighborhood [16]. Each edge is represented as an ordered sequence of sampled points and we extract heavily overlapping contour fragments from every edge (each of length L), by one point shifting, so that neighboring contour fragments overlap by $L-1$ points. The fragment length L is selected as a balance between discriminative power and edge fragmentation. For each fragment individually we sort the sequence clockwise and calculate the corresponding shape descriptor matrix of size $L \times L$ as it is described in Section 3.1. All obtained descriptors are then used to build the vocabulary tree. In such a way, at the final level of the clustering, the leaves define the desired codewords representing prototypes of contour fragments.

Once the vocabulary tree is generated, it may be used for fast codeword assignment and scoring, as well as insertion of new images for retrieval. For each of its description vectors, the top nodes and their cluster centers are matched. Only the children of the best matched cluster center are then matched again. This reduction of search space allows for a complete search of the vocabulary in $k \times l$ comparisons. Thus for a structure of ten cluster centers and six levels searching the one million leaf nodes for a best match only requires 60 comparisons, see Figure 5 for an illustration of this process.

During the insertion of a new image this advantage is used to find the best matching leaf node quickly. For each of the shape descriptors such a match is sought. Then, a new image identifier is included into the nodes' inverted file lists and their weights are updated. No further steps are required. The same hierarchical matching is used to determine the best matching leaf nodes for retrieval.

3.3. Retrieval system

For retrieving images from a database a user has to provide a shape sketch by drawing strokes. Therefore, we created a sketching interface, where the user draws on a tablet computer with an electronic pen or mouse. This directly allows to store the location and sequence of strokes. In such a way our object model has the following four attributes (refer to Figure 3): The sketch models the shape of the object of interest the user is searching for. It contains multiple strokes, which are the user-drawn lines to outline the object. Each stroke has a direction of drawing given by the user. Finally, adopted from the direction, each stroke has a sequence in which points along the stroke may be sampled.

To be able to retrieve images from the database, each stroke is modeled as a contour and highly overlapping contour fragments, as described in Section 3.2, are extracted for every stroke. All contour fragments are represented by our proposed powerful shape descriptor and the descriptors are passed to the vocabulary tree to obtain corresponding codewords. The retrieval result is a ranking of all images in the database using the final similarity scores as defined in Equation 4.

4. Experiments

The goal of this work is to propose a novel structure-capturing shape cue for content-based image retrieval (CBIR) systems. For this reason, we focus on evaluating shape cues for the quality in image retrieval. Due to the design of our retrieval system, it shows the following properties. The shape vocabulary generation is performed in an offline stage and stays the same over all experiments. The vocabulary tree allows a fast retrieval of images as well as insertion of new images in constant time. The computation only depends on the number of k clusters and l levels chosen for the vocabulary, which is k = 3 and l = 6 for all our experiments. The constant time for insertion or retrieval of new images is thus $O(k \times l)$, which is (almost) independent of the number of images in the database. Since we focus on efficient local shape features, the time for calculating the descriptors is a few milliseconds. The full retrieval is performed on average in 50 milliseconds seconds per object sketch.

4.1. Shape-based features

For evaluation of our contour descriptor, we analyzed four additional descriptor methods. The Shape Context (SC) [2] is a correlated histogram of edges and is intended to provide a description for a set of points to determine their correspondences. The description is a normalized binned histogram, however in a log-polar layout to capture the relative distribution of points. The Turning Angle (TA) is a subset of the Beam Angle Histogram (BAH) [20]. The BAH is a histogram over beam angle statistics, where the beam angles θ_{ij}, at points on the shape P_i, i = 1, 2, . . . , are the chord lines (P_{i-j}, P_i) and (P_i, P_{i+j}). The PArtial Contour and Efficient Matching (PACEM) [21] is a recent shape descriptor designed for partial matching and encoding of sequence information.

4.2. Experimental setup

The experiment is designed to evaluate the performance of a *query-by-shape-sketch*, where rich visual features are not available. As it is difficult to evaluate an interactive user scenario, we setup the experiment to use the ETHZ shape classes [9] of 255 images containing five classes and let several users draw sketches for each class. The benefits are that the class for each image is known and we can use it to evaluate the retrieval performance, which is otherwise not well-defined in large image retrieval systems, where the exact number of true positive matches is not known.

For evaluation we use all obtained sketches, which represent the range of variations of typical user sketches. See Figure 6 for an overview of some of the sketches, which are provided as query input to

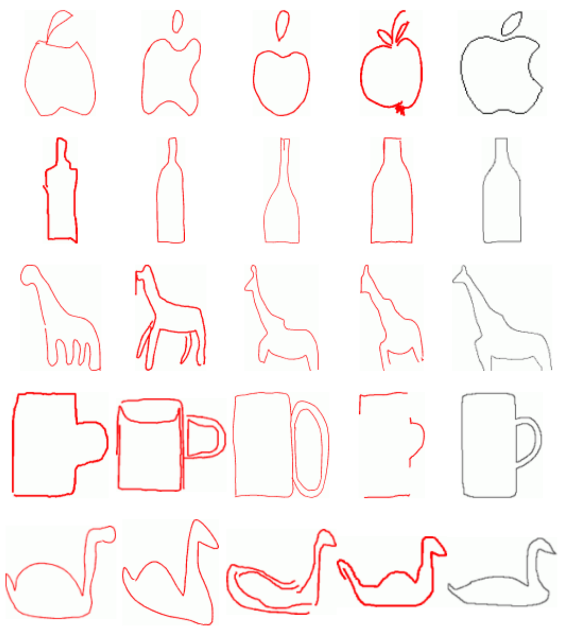


Figure 6. Subset of the 700 user sketches for the five ETHZ classes used in evaluation. The sketches cover the range of user input in a *query-by-shape-sketch* retrieval system. Second last column shows top performing sketches, and right column the sketches by Ferrari [9].

the image retrieval system. This new sketch dataset[7] contains 700 sketches drawn by 36 users. There are on average three user strokes with a length of 320 pixels. We use contour fragments of length 100 and sample every 5th point, leading to a length L = 20, which in experiments showed is a reasonable balance between discriminative power, dimensionality and limitations due to edge fragmentation.

For this dataset the performance measure is the top-T ranked results, where the top-T score is defined as the number of true positive images (ground truth class vs. sketch query class) over the top T = 20 result images. This performance score shows how many retrieved images actually contain the desired object.

4.3. Results and discussion

Table 1 shows a summary of the average results of the 700 queries for the top-20 ranked images. The results show that the performance scores of the novel *query-by-shape-sketch* image retrieval paradigm are still moderate, however clearly demonstrate the benefits of using a shape descriptor, which captures the sequence of user strokes. Our descriptor performs on average 25% better than other shape descriptions.

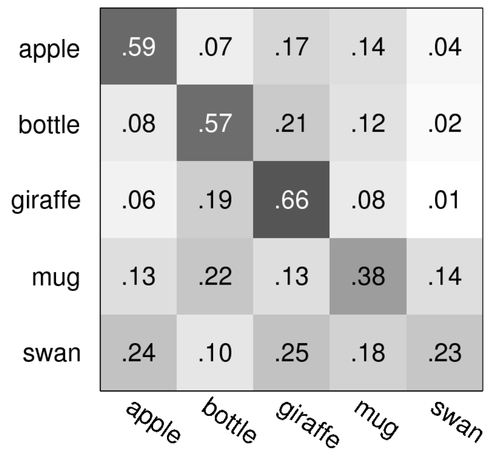
[7]www.icg.tugraz.at/Members/hayko/retrieval-by-sketch

Method	Sketch (avg./best)		Ferrari [9]
SC [2]	23.5%	41%	20%
TA	20.7%	44%	20%
BAH [20]	19.6%	55%	24%
PACEM [21]	19.2%	50%	20%
Proposed	**48.5%**	**87%**	**58%**

Table 1. Percentage of true positives within first 20 retrieved images using each of the 700 sketches of the novel dataset (average and best results) and the hand-drawn prototype models (right column).

Figure 8 shows a recall plot for the retrieval task, where the number of top ranked images was varied from T = 1 to T = 20. The retrieval score is consistent over all top ranked images.

For completeness, we can evaluate the individual class results of the ETHZ dataset. This is not relevant for the retrieval systems, however the confusion table in Figure 7 shows that some categories can be modeled better than others. The average percentage of true positives within the first 20 retrieved images over all user sketches (including very crude ones) is for Applelogo 59%, Bottle 57%, Giraffe 66%, Mug 38%, Swan 23%. This distribution of performance is also visible when using the hand-drawn prototype models provided by Ferrari et. al [9]: Applelogo 60%, Bottle 85%, Giraffe 90%, Mug 20%, Swan 35%. Furthermore the scores if only considering the top performing sketch per class yields: Applelogo 100%, Bottle 90%, Giraffe 100%, Mug 80%, Swan 70%. The classes for swans and mugs are the hardest, since they are most often confused with applelogos and bottles, respectively, due to similar local shapes (head, neck and straight vertical lines).



	apple	bottle	giraffe	mug	swan
apple	.59	.07	.17	.14	.04
bottle	.08	.57	.21	.12	.02
giraffe	.06	.19	.66	.08	.01
mug	.13	.22	.13	.38	.14
swan	.24	.10	.25	.18	.23

Figure 7. Confusion table for average scores for each ETHZ shape class [9] on the 700 user sketches.

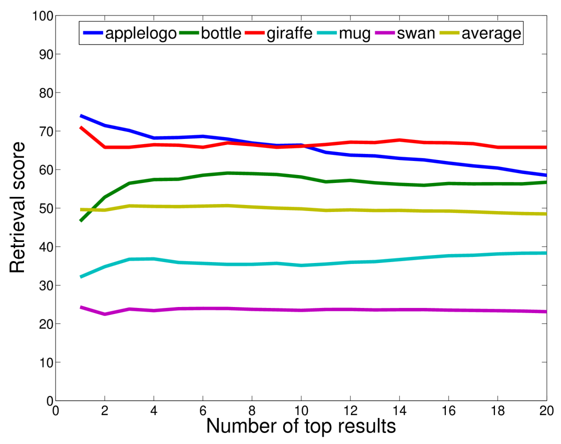


Figure 8. Recall for varying number of top ranks shows consistent retrieval results for *query-by-shape-sketch*.

However, for retrieval one is interested in the average performance over all classes, which is shown in Table 1. Here prototype models scored 58%, the average of all 700 user sketches scored 48.5% and the best single sketches scored 87%. Thus we can confirm that the sketches by Ferrari *et al.* resemble the shape prototypes quite well [8], however there are better prototypes, see second last column in Figure 6 for our best sketches. Thus on average, which reflects the typical user behavior, we can achieve a retrieval rate of 48.5%. This means using a simple hand-drawn sketch of the shape of an object, we can retrieve half of the desired images in an interactive content-based retrieval system in 50 milliseconds.

5. Conclusion

In this work we showed a novel content based image retrieval (CBIR) system, which queries a large database by means of a user-drawn sketch. This *query-by-shape-sketch* paradigm is the most intuitive extension of the current language-based semantic queries onto the visual domain. Our novel combination of shape-based features which exploit the properties of user sketches such as partial description, multiple line strokes, as well as direction and sequence of the stroke itself, and an efficient retrieval system based on hierarchical clustering and scoring allows the user to search for images by simply drawing the object of interest. This extends the current state-of-the-art by allowing an object-centered search rather than full scene retrieval.

Future work will focus on the integration of other input feature types such as color and texture, adopting a query expansion by linking *query-by-shape-sketch* results and a *query-by-image* strategy, localization and geometric verification of sketched objects within the retrieval results and finally, investigating the universality of the shape vocabulary.

Acknowledgements We would like to thank the following people for the sketching: andrea, andy, christian, drmike, eram, george, gerhard, gerlinde, gernot, horst, joachim, julia, katl, krisi, lisa, mark, markus, martina, martinL, mat, mughero, nadja, PapaG, peterk, pmroth, robert, sabine, silke, stefan, steffi, thomas, tom, waci, werner, wm and various anonymous Open Lab Night visitors; as well as the support by the Austrian Research Promotion Agency (FFG) project FIT-IT CityFit (815971/14472-GLE/ROD).

References

[1] I. Bartolini, P. Ciaccia, and M. Patella. Query Processing Issues in Region-Based Image Databases. In *Knowledge and Information Systems (KAIS)*, 2010.

[2] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 2002.

[3] A. Berman and L. Shapiro. A Flexible Image Database System for Content-Based Retrieval. *Computer Vision and Image Understanding (CVIU)*, 1999.

[4] J. Canny. A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 1986.

[5] O. Chum, M. Perdoch, and J. Matas. Geometric min-hashing: Finding a (thick) needle in a haystack. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[6] T. Cootes, D. Cooper, C. Taylor, and J. Graham. Trainable method of parametric shape description. *Journal of Image Vision Computing (JIVC)*, 1992.

[7] R. Datta, D. Joshi, J. Li, and J. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 2008.

[8] V. Ferrari, F. Jurie, and C. Schmid. From images to shape models for object detection. In *Intern. Journal of Computer Vision (IJCV)*, 2009.

[9] V. Ferrari, T. Tuytelaars, and L. V. Gool. Object detection by contour segment networks. In *Proc. European Conference on Computer Vision (ECCV)*, 2006.

[10] J. Friedman, J. Bentley, and R. Finkel. An algorithm for finding best matches in logarithmic expected time. In *ACM Transactions on Mathematical Software*, 1977.

[11] E. Gelasca, J. De Guzman, S. Gauglitz, P. Ghosh, J. Xu, E. Moxley, A. Rahimi, Z. Bi, and B. Manjunath. Cortina: Searching a 10 million + images database. In *Proc. of Conference on Very Large Data Bases (VLDB)*, 2007.

[12] Q. Iqbal and J. Aggarwal. CIRES: A System for Content-based Retrieval in Digital Image Libraries. In *Proc. of Intern. Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2002.

[13] C. Jacobs, A. Finkelstein, and D. Salesin. Fast Multiresolution Image Querying. In *Proc. of the Intern. Conference on Computer graphics and interactive techniques (SIGGRAPH)*, 1995.

[14] H. Jegou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *Intern. Journal of Computer Vision (IJCV)*, 2010.

[15] P. Kontschieder, M. Donoser, and H. Bischof. Beyond pairwise shape similarity analysis. In *Proc. Asian Conference on Computer Vision (ACCV)*, 2009.

[16] P. Kovesi. MATLAB and Octave Functions for Computer Vision and Image Processing. School of Computer Science & Software Engineering, The University of Western Australia.

[17] D. Lowe. Distinctive image features from scale-invariant keypoints. *Intern. Journal of Computer Vision (IJCV)*, 2004.

[18] A. Mikulik, M. Perdoch, O. Chum, and J. Matas. Learning a fine vocabulary. In *Proc. European Conference on Computer Vision (ECCV)*, 2010.

[19] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[20] N. Payet and S. Todorovic. From a set of shapes to object discovery. In *Proc. European Conference on Computer Vision (ECCV)*, 2010.

[21] H. Riemenschneider, M. Donoser, and H. Bischof. Using Partial Edge Contour Matches for Efficient Object Category Localization. In *Proc. European Conference on Computer Vision (ECCV)*, 2010.

[22] T. Sikora. The MPEG-7 Visual standard for conet description - An Overview. In *IEEE Trans. on Circuits and Systems for Video Technology*, 2001.

[23] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proc. IEEE Intern. Conference on Computer Vision (ICCV)*, 2003.

[24] H. Wolfson and I. Rigoutsos. Geometric hashing: An overview. *Computational Science and Engineering*, 1997.

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

Incremental Superpixels for Real-Time Video Analysis

Jochen Steiner Stefanie Zollmann Gerhard Reitmayr

`http://www.icg.tu-graz.ac.at/`
Graz University of Technology
Graz, Austria

Abstract. *The segmentation of images as input for image analysis is used in various applications. The resulting segments are often called superpixels and can be used for further analysis to compute certain information about the objects in the picture. Unfortunately, the majority of superpixel algorithms are computationally expensive. Especially for real-time video analysis it is hard to find a proper algorithm to compute superpixel representations without decreasing the quality of the results. Available algorithms for real-time use may not satisfy the requirements of every application case. In this paper we suggest an incremental approach for real-time segmentation of incremental video data. We show that our incremental approach does not influence the quality of the results noticeably. Finally the efficiency of the approach is demonstrated within a panoramic-tracking based application and shows the advantage over existing real-time superpixel algorithms.*

1. Introduction

There are several fields of application for segmentation of images or so-called superpixel algorithms. Computing a superpixel representation of an image is often a first step for further processing in image analysis. It is typically used to locate image regions or compute image statistics. Application fields are for instance medicine, geographical applications or augmented reality. Due to the wide range of application areas the requirements for superpixel algorithms differ. Geographical or medical applications need to process a large amount of data. On the other hand, using superpixels for image analysis on live camera streams like in augmented reality applications requires short computation times. For image interpretation perceptually natural shapes of the computed superpixel are useful. For instance Felzenszwalb and Huttenlocher introduced an approach that tries to preserve the natural shape of objects [2]. They showed the consistency of their superpixel elements and human perception of shapes. Due to these characteristics their *Efficient Graph-Based Image Segmentation* algorithm (*EGBIS*) is used for various applications. For example Hoim et al. used it for creating geometric interpretations of scenes from one image [3] and Zollmann et al. used the EGBIS superpixels as input for improving the depth-perception by applying occlusion management in augmented reality applications [13] depending on perceptual groups.

However, EGBIS is computationally too expensive for real-time processing, a requirement for augmented reality applications. On the other hand, existing real-time superpixel algorithms are subject to quality degradation compared to ground truth segmentation data [1].

Zollmann et al. managed EGBIS's lack of performance by capturing a panoramic image of the whole environment and computing the time-consuming superpixel calculation once in advance on this panoramic representation. During runtime, they remap the panoramic superpixel map into the current camera view by using tracking data. The disadvantage of this approach is that the complete environment has to be captured before starting their approach.

To avoid such an involved precomputational step, the idea of our approach is to compute superpixels incrementally at the time new image data is acquired. For example, in the panoramic mapping and tracking approach described by Wagner et al. [10], the complete panoramic image is not available from the beginning, but compiled over time. Applying the segmentation to the subset of newly recorded pixels is fast enough for real-time applications. However, traditional segmentation methods work in a global way and assumes the overall image is known. In this paper, we extended a traditional segmentation method to cope with an image that is recorded incrementally. We will show that the superpixels can be created in real-time. In an accuracy comparison with manually created ground thruth data we will prove that the results of our incremental method are almost the same like the results of EGBIS. Furthermore we will show the application of our incremental method by integrating it into a panoramic mapping and tracking approach and creating superpixels in real-time for a panoramic image at the same the panoramic image is build online.

2. Related work

There is a lot of previous work on segmentation based on different techniques. For instance graph-based image segmentation techniques represent an image as a graph $G = (V, E)$, where each node V represents one pixel in the image and each edge E connect neighboring pixels. An early graph-based method is the work of Zahn et al. [12]. It is based on the minimum spanning tree of the graph and uses fixed thresholds. The edge weights are based on the differences between pixels. Zahn's method breaks edges with large weights. But that leeds to high variability regions being split into multiple regions or it merges ramps and constant regions together.

In 1982 Urquhart et al. [9] tried to deal with that problem. They normalize the edge weights by using the smallest weight incident on the nodes connected by that edge.

One of the fastest superpixel methods is the graph-based EGBIS approach introduced by Felzenschwalb and Huttenlocher [2]. Superpixel created by EGBIS are perceptually natural in shape since they preserve details in low-variability image regions and ignore details in high-variability image regions.

Other segmentation methods are based on finding minimum cuts in a graph. The goal is to minimize the similarity between pixels that are being split. Wu and Leahy were the first to introduce a segmentation method using graph cuts [11]. But their algorithm was biased finding small components. Shi and Malik approached that bias with their normalized cut criterion often referred to as N-Cuts [7]. This was the basis for the graph cut superpixel algorithm of Ren and Malik, which segments an image into a large number of small and quasi-uniform superpixels [6]. But these algorithms are too slow for real-time use.

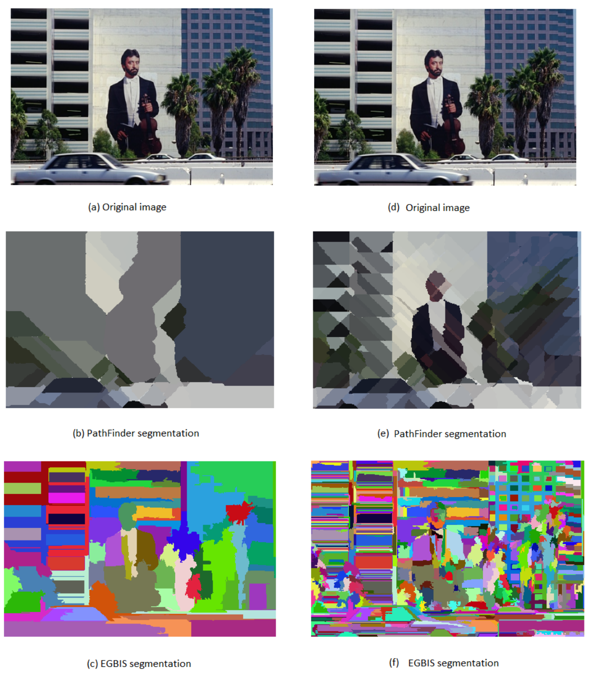


Figure 1. A Pathfinder segmentation compared to the result of an EGBIS segmentation for one of the ground truth images. Left: Both segmentations with a number of 70 superpixels. Right: Both segmentations with about 600 superpixels.

Levinshtein et al. [4] introduced the so-called *TurboPixels* that are based on geometric flow, limit under-segmentation and provide a better performance than N-Cuts but still slower than EGBIS.

A real-time superpixel algorithm that is based on least-cost paths was introduced by Drucker and MacCormick [1]. Indeed their *Pathfinder* can be 30x faster (depending on the image size) than EGBIS, but shows rasterized superpixels, which appeal synthetic in shape. Drucker and MacCormick showed the qualitative differences between Pathfinder and EGBIS by an quantitative comparison. For that purpose they computed the *mean accuracy* of the segmentation with reference to a manually drawn ground truth segmentation [8], as defined by Moore et al. [5].

Figure 1 shows the result of both methods for one of the ground truth images. EGBIS achieves more accurate and perceptually consistent superpixels than PathFinder at the same number of superpixels. The left side of Figure 1 presents the segmentation results for a very small number of superpixels while the right side shows the segmentation results for a commonly used number of superpixels. In both cases EGBIS

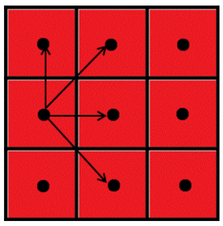


Figure 2. Edges in EGBIS are constructed for every pixel and four of its neighbors.

preserves much more details of distinct regions.

To provide the real-time computation of perceptually natural shaped superpixels with a high quality in reference to ground thruth segmentation, we decided to build an incremental version of the EGBIS segmentation method.

3. Traditional EGBIS Method

Since our incremental approach is based on the traditional EGBIS method, we will shortly describe the details of the method that are important for understanding the incremental approach.

The EGBIS algorithm represents images as a graph. While every node in the graph corresponds to a pixel of the image, the edge between two nodes represents the difference measure between the nodes.

The first step of the EGBIS method is the creation of the graph. Therefore Felzenszwalb and Huttenlocher compute difference measurements for every pixel and four of its neighbors (see Figure 2). These measurements form the weights of the edges. This computing step is repeated for every pixel and an edge for every pair of directly adjacent pixels is constructed. Finally the edges are sorted by non-decreasing edge weight.

The next step is the creation of superpixels themselves. When comparing two regions, the methods checks if the difference $D(C_1, C_2)$ between the components C_1 and C_2 is small compared to the *internal difference* within the two components. The difference between two components is defined as the minimum weight between them. The internal difference $I(C)$ of each component of the graph is defined as the largest weight in the minimum spanning tree.

$$I(C) = D(C_A, C_B) + \tau(C) \tag{1}$$

where $D(C_A, C_B)$ is the weight of the edge connecting the last two components (C_A and C_B) merged into C.

A threshold function $\tau(C)$, which is based on the size of the component and a constant k, is used to define how much the difference between the components must be smaller than the minimum internal difference,

$$\tau(C) = \frac{k}{|C|} \tag{2}$$

where $|C|$ is the size of the component and k is a parameter to control the preference for the component's size. If the difference between the components is smaller than the minimum internal difference of both, the components are merged. The algorithm iterates over all edges repeating that comparison for every edge.

Finally, in their implementation they used a post-processing step that is not described in their paper. The post-processing merges superpixels with a size below a defined minimum size min. Because edges are sorted in increasing order of their weights, regions with lower differences will be merged first.

4. The Incremental Superpixel Algorithm

The basic idea of our method is to divide the process of traditional EGBIS segmentation into smaller steps. Therefore we subdivide a full image into cells and instead of segmenting the complete image at once the image is segmented cell by cell. Thereby the current processed cell may correspond for instance to the current field of view of the camera or to parts of a video image that changed in content or interest.

Algorithm 1 Incremental Superpixel

```
Initialize data structures for complete image
for each newly completed cell do
    Initialize and smooth cell image
    Create and sort edges in cell
    Segment cell by using internal difference
    Merge sorted cell edges in complete edge set
    Copy disjoint-set forest containing superpixels
    Perform post-processing on complete image
end for
```

Our method consists of an initial part that has to be performed once in advanced to create all data structures and incremental parts that have to be performed for each processed cell. Like the EGBIS method our approach stores the segmentation by using a disjoint-set forest. Initially, this disjoint-set forest is constructed for the complete image with $w \times h$ default elements and without joined components (where w is width and h is height of the image). In each incremental step the edges for the current cell are calculated and sorted, the segmentation for the current

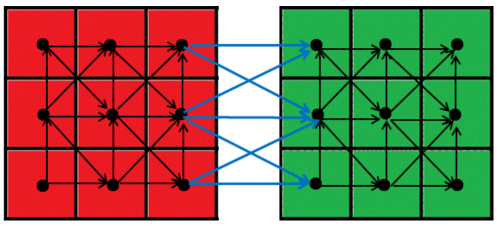


Figure 3. Edges in our method are constructed for every cell like in the original algorithm and additionally for border pixels with existing neighbors.

cell is performed and the edges of the current cell are merged into a set of edges of the complete image. Finally, the post-processing is performed in each step but on the complete already segmented data to ensure that also superpixels that are distributed over different cells are properly merged.

4.1. Edge Creation

The calculation of the weights of the edges is quite similar to the traditional EGBIS algorithm. For each processed cell we calculate the edges like for a separate image in the original algorithm. There is only one case which needs special attention: At the cell border neighboring cell could have been already segmented. That means the border between the current cell and the already processed cell has to be segmented as well. To find already segmented cells a boolean matrix stores for each pixel, wether it have been already processed. If a pixel at the cell border is stored as already segmented we compute an additional edge between the border pixels. The computation of the additional edges for two neighboring cells is illustrated in Figure 3.

4.2. Segmentation

To segment the current image cell all edges of the image cell are sorted in increasing order by their weights. After sorting we iterate through all edges of that cell and compute their corresponding components (preliminary superpixel) by using the disjoint-set forest. Each set of the disjoint-set forest is the data-representation of one component.

In the beginning of the iteration process each pixels corresponds to one preliminary component, that means that no pixel share the set with another pixel. To determine the components of each pixel we compare in each iteration step the weight of the current processed edge with the internal difference (as described in Equation 1) of the both components connected by this edge. If the weight is smaller than the internal difference we merge both sets representing the components into a single set.

At each part of this process special attention has to be paid to the border regions between two cells. For instance, regions that are considered to belong to one perceptual group but are distributed over several cells have to be handled with special care. Although these edges are sorted and processed in the cell's segmentation process as well, it may happen that due to the changed data order of the subdivide image originally connected superpixel are not detected.

The main problem of subdividing the image into cells is that very homogeneous components, which are distributed over two ore more cells, may be not determined as being part of the same component but as different components. We refer to this problem as tiling. Figure 4 shows that in the picture of the palm tiling occurs especially in the homogenous regions of the sky or the clouds. Instead of merging the homogenous components to one large component nearly for each cell one component is created.

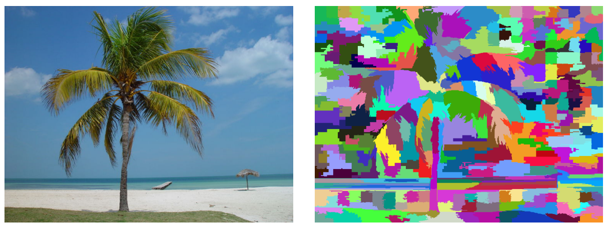


Figure 4. Left: Example image. Right: Example image segmented without using the border threshold.

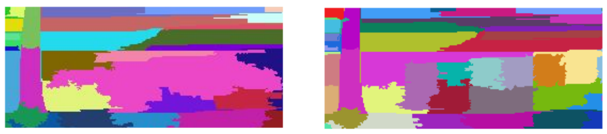


Figure 5. Tiling: Detailed view of Figure 4 (right side of the beach) showing the problem of tiling. Left: EGBIS without tiling. Right: Incremental methods shows tiling.

The reason for tiling is that instead of processing all edges of the complete image as in the traditional algorithm, in the incremental approach we only use the edges of the current image cell and the border edges for the segmentation.

The segmentation criterion of the original EGBIS is based on the amount of variability of adjacent segments. This creates problems when segmenting an image incrementally. To describe the origin of this problem we will briefly describe how superpixels grow in our method and how they grow in the original EGBIS depending on the internal difference.

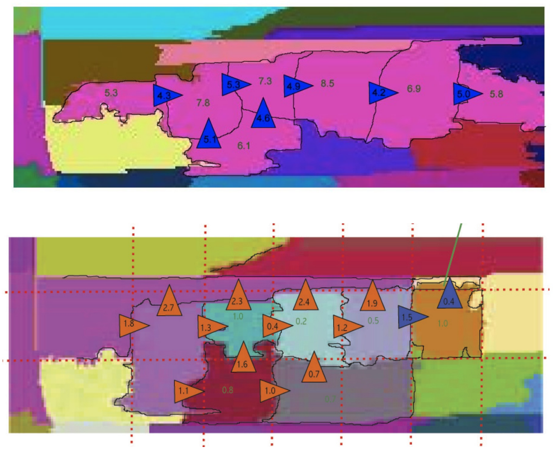


Figure 6. Illustration of tiling problem from Figure 5. Top: Example of merging superpixels by EGBIS. Bottom: Example of merging superpixels by our incremental method at a cell size of 30 pixels per side. The yellow and blue arrows show the edges and their weights.

As defined in equation 1 the internal difference of a superpixel C depends on the size of a superpixel $|C|$, on a constant for controlling the shape of the superpixels k and on the weight of the edge connecting the last two components merged into C. In the case that a cell of an image is a very homogeneous area like parts of the beach in Figure 4, the differences between all pixels in that cell are very small. Thus all weights of edges of the current processing step are very small. And not only that it is likely that nearly all pixels are mapped to one single component inside the current cell, but also the internal difference of this component is very small since $D(C_A, C_B)$ is staying small and $|C|$ is getting larger. Because of the homogeneity of the cell, there will be no edge that increases the internal difference. That means that it is getting unlikely that exactly the boarder edges between the cells will fulfill the requirement for merging, because they have a higher weight.

The original algorithm does not have this kind of problem, because the edges covering the whole image and are not limited to a homogenous area. That means that all kind of different weights will occur. Segments will start to grow with edges with low weights, but also edges with higher weights can be included in the growing segment, because if there are still few edges included $\frac{k}{|C|}$ is large and the internal difference may be not too small for merging.

Figure 6 illustrates this problem by showing the differences in segment merging of both algorithms. The top figure shows how a superpixel is created by merging a few smaller superpixels. In that example

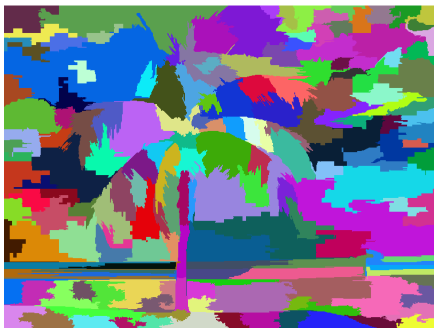


Figure 7. Example image from Figure 4 segmented by the incremental method using the border threshold.

all drawn edges are at a weight smaller than the internal weights of the superpixels they connect. So they are all merged together as shown in Figure 6 a) with the large pink segment. Figure 6 bottom shows the tiling in the incremental approach. By using only edges with a low differences in each incremental step (because a homogeneous cell has no edges with higher weights), the internal differences (visualized as background numbers) are getting very low and segments distributed over different cells often can not be merged, because nearly no border edge (visualized with the orange triangles) will meet the required minimal difference.

We decided to approach that problem by introducing a new threshold value. This threshold value allows us to investigate if components that are distributed over several cells should be merged. If the difference between two components of different cells that have a direct connection to each other is lower than this threshold value, they will be merged. In this case it is not likely that there exist a real perceptual border between these components. Figure 7 shows the result of using the threshold.

4.3. Post-processing

The postprocessing of the traditional EGBIS merges superpixels containing less pixels as the minimum size algorithm parameter. Changing the sorting of the edges due to subdividing the image into cells also influences the behaviour of the post-processing. If we would perform the post-processing on a per-cell base, it is likely that all superpixel that do not have the minimum superpixel size are merged to one large superpixels covering nearly the complete cell. Thereby the post-processing disregards that smaller superpixels may be distributed over several cells and may meet the size requirements in the next incremental step.

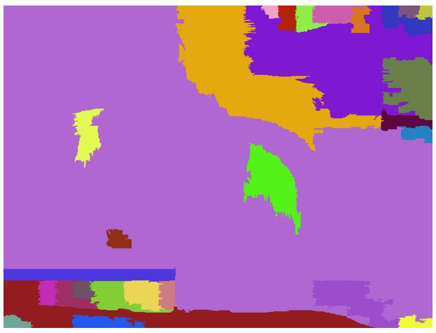


Figure 8. Post-processing error. If the post-processing is applied on a per cell-base, it is likely that the incremental approach creates very large superpixels covering nearly the complete image.

But in the next processing step the large superpixels of each cell may be merged together with other large superpixels of neighboring cells. That may result in even larger superpixels spanning nearly over the whole image as shown in Figure 8. To avoid this kind of post-processing error and since the post-processing itself is not highly computationally expensive, we decided to apply the post-processing not only on the currently processed cell, but also on all finished cells.

Furthermore we had to adapt the data management of the superpixels, because in the traditional algorithm the segmentation process as well as the post-processing are performed on the same disjoint-set forest. Since our approach is an incremental method, the output of the post-processing (the disjoint-set forest) is on the other hand the input for the next incremental step. To avoid the influence of the post-processing to the next incrementation step, we have to make a copy of the original disjoint-set forest and perform the post-processing on this copy.

Additionally the dataset that stores all edges has to be updated as well, since the post-processing uses the edges for iterating. Therefore we insert the edges of each new cell in the sorted list of the existing edges.

In each step four edges for each pixel are created. For instance for small cells of 30×30 pixels that would be an amount of 3600 edges. Sorting all these edges into complete set of edges which can have an final amount of $w \times h \times 4$ edges is also not computationally inexpensive. But in this step it is very easy to save processing time. Because the difference between two superpixels is defined as the minimum weight between them. That means it is not necessary to keep all edges between them. Only the minimum difference is needed. Deleting the redundant edges reduces the edges to be merged into the complete edge database for every cell to an average of 125 edges.

Size in pixel	30x30	50x50	70x70
Image Creation	9 ms	7 ms	9 ms
Edge Creation	1 ms	3 ms	5 ms
Merge	1 ms	1 ms	1 ms
Segment	5 ms	10 ms	24 ms
Postprocess	9 ms	8 ms	9 ms

Table 1. Computation times of the algorithm in relation to the cell size.

5. Results

To show quality of results and the improvements considering the processing time, we compared our incremental method with the traditional EGBIS algorithm.

We computed the average processing time for 50 different images with the same image size (750×563 pixel) but for different cell sizes. Figure 9 illustrates the processing time per cell in relation to the size of the cell averaged. The processing time per cell increases with the cell size, but is still capable for real-time applications until a cell size of 70×70 pixel. In table 1 we show the processing time of each step of the incremental method. The post-processing and creation of the image structure are the most computationally expensive parts, fortunately these steps are not increasing linearly with the cell size.

Figure 4 shows a selected example of the 50 images used for the average computation. On our test system (Intel Core i7 740QM, 8 GB Ram) the traditional EGBIS algorithm needs 577 ms to segment this image with the following parameter: preprocessing smoothing parameter $\sigma = 0.5$, $k = 50$, $min = 500$. Segmenting the image with the same

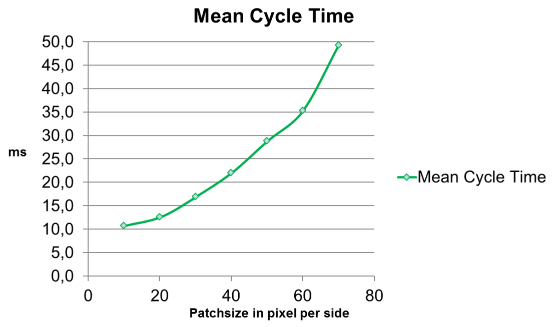


Figure 9. Average compuation time in relation to cell size (in pixel per side) of 50 different images.

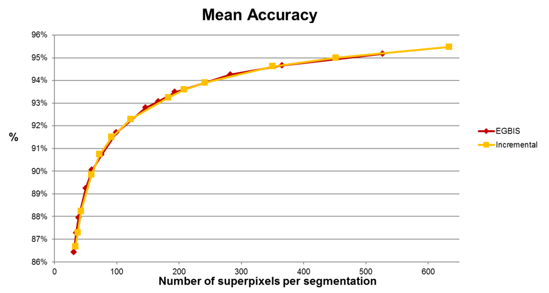


Figure 10. Comparison of the mean accuracy of EGBIS and the incremental superpixel method computed in respect of 50 ground truth segmentations.

parameters with the incremental method has an average processing time of 25 ms per 30 × 30 pixel cell. When assuming a cycle time of 25 ms per cell and one cell per application cycle a frame rate of 40 frames per second can be achieved, which is adequate for real-time applications.

For comparing the accuracy of our method to the accuracy of the traditional EGBIS algorithm we use ground truth segmentations as described by Drucker and MacCormick [1]. The comparison is calculated by finding for each superpixel of the algorithm output the ground-truth segment which shows the biggest overlap. The accuracy of a single superpixel is then defined as the amount of the overlap with the segment in relation to its size. We computed the mean accuracy as the average accuracy of all superpixels for 50 ground truth images. Figure 10 shows the mean accuracy of EGBIS and the Incremental Superpixel. We found that the accuracy of both algorithms is nearly the same with a mean of 91.3% of EGBIS to a mean of 91.7% of our method. That shows that our method does not decrease in accuracy compared to EGBIS.

Figure 11 shows an example of 50 ground truth segmentations. In Figure 12 we show the corresponding segmentations of that example. They were done by all three algorithms PathFinder, EGBIS and Incremental Superpixels with left approximately 70 and right approximately 600 superpixels.

These figures show that the incremental superpixel method can achieve a perceptual superpixel quality similar to EGBIS while PathFinder falls back behind both considerably.

6. Incremental superpixels for panoramic mapping and tracking

To test our method in an application we integrated the incremental superpixels into a simultaneous panoramic mapping and tracking approach similar to the one introduced by Wagner et al. [10].

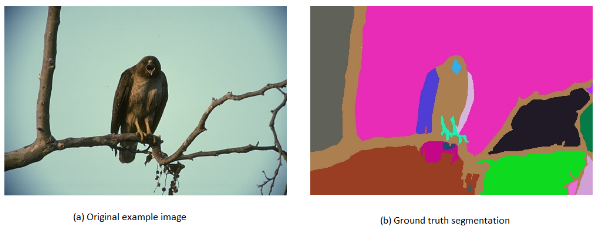
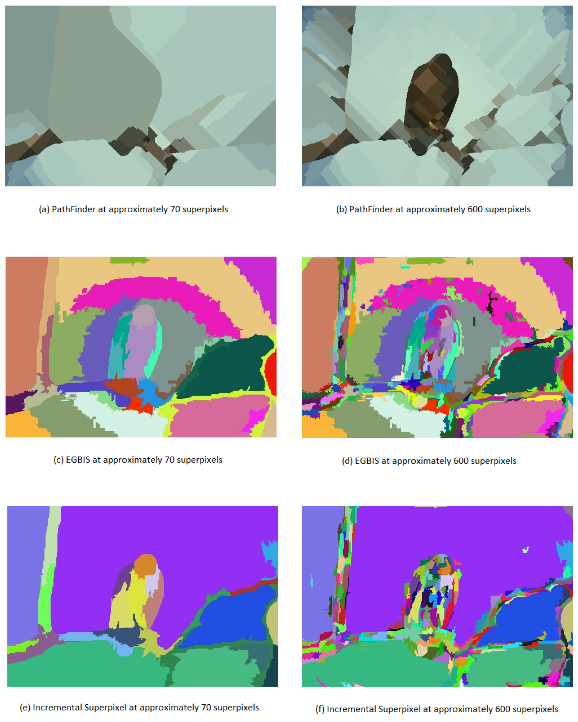


Figure 11. Example of a ground truth image segmentation.



(a) PathFinder at approximately 70 superpixels
(b) PathFinder at approximately 600 superpixels
(c) EGBIS at approximately 70 superpixels
(d) EGBIS at approximately 600 superpixels
(e) Incremental Superpixel at approximately 70 superpixels
(f) Incremental Superpixel at approximately 600 superpixels

Figure 12. Ground truth comparison of Pathfinder, EGBIS and the incremental superpixel method. Left: For around 70 superpixels. Right: For around 600 superpixels.

Simultaneous panoramic mapping and tracking allows accurate and robust rotation tracking in outdoor scenarios by creating a panoramic map from the live camera stream. The panoramic map is then used for tracking and is stored in three different resolution levels (2048 × 512, 1024 × 256 and 512 × 128 pixel).

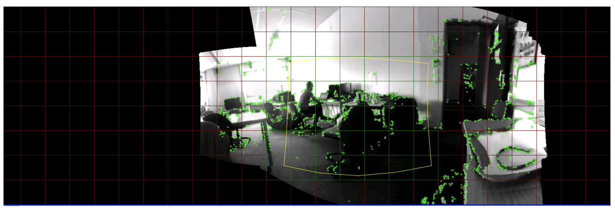


Figure 13. A partly finished panoramic map.

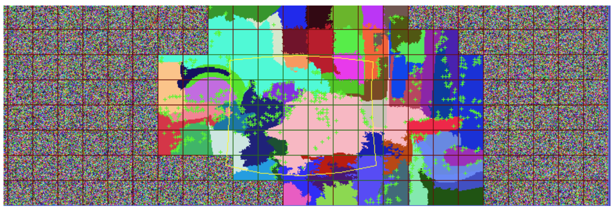


Figure 14. Incremental superpixels for panoramic mapping and tracking.

During the creation process of the panoramic map the map is splitted into 32x8 cells as shown in Figure 13. Every cell has a state that describes if the cell is either unfinished or finished. Finished cells are then down-sampled from the full resolution to the lower resolution levels. The lower resolution levels are used for keypoint extraction. We decided to use the medium resolution to calculate the superpixel representation by using our incremental approach. That means that for each finished cell superpixels are computed and merged to the existing superpixel representation as shown in Figure 14. The superpixel representation of the panoramic map can then be used to extract the superpixels for the current camera view by remapping the map into the current camera perspective as described by Zollmann et al. [13].

7. Conclusion

In this paper we introduced a method that reduces the cost of a superpixel segmentation by applying it incrementally as new image data is acquired. This approach can be used for all applications that rely on an incremental image data acquiring process, such as panoramic mapping and tracking or video analysis which is partly updated (e.g. fixed background). To implement the incremental method, we extended an traditional algorithm to enable the segmentation of newly arriving image cells. Merging the information of incremental segmentation steps has several challenges, such as the special attention that has to be paid to border edges of a cell, the reduced data set used in each incremental step and thus the changed order of sorted image edges. Furthermore the post-processing can not be applied incrementally.

Finally we showed the application of our method by integrating it into a panoramic mapping and tracking approach. Even if we do not reach the performance of the PathFinder method and will not get the exactly same results as the traditional EGBIS, our approach is a good trade-off between performance and quality of results.

Acknowledgements

This work was partially funded by the Austrian Research Promotion Agency (FFG) under contract no. FIT-IT 820922 (SMART Vidente).

References

[1] F. Drucker and J. MacCormick. Fast superpixels for video analysis. In *Motion and Video Computing, 2009 (WMVC2009)*, pages 1–8. IEEE Computer Society, Dec. 2009. 1, 2, 7

[2] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, Nov. 2004. 1, 2

[3] D. Hoiem, A. A. Efros, and M. Hebert. Automatic photo pop-up. In *ACM SIGGRAPH 2005 Papers*, pages 577–584. ACM, July 2005. 1

[4] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, and S. J. Dickinson. Turbopixels: Fast superpixels using geometric flows. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 31, pages 2290–2297. IEEE Computer Society, Oct. 2003. 2

[5] A. P. Moore, S. Prince, J. Warrell, U. Mohammed, and G. Jones. Superpixel lattices. In *CVPR2008*, pages 1–8. IEEE Computer Society, June 2008. 2

[6] X. Ren and J. Malik. Learning a classification model for segmentation. In *Ninth IEEE International Conference on Computer Vision, 2003*, pages 10–17. IEEE Computer Society, Oct. 2003. 2

[7] J. Shi and J. Malik. Learning a classification model for segmentation. In *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, pages 888–905. IEEE Computer Society, Aug. 2000. 2

[8] B. University. The Berkeley Segmentation Dataset and Benchmark, 2007. 2

[9] R. Urquhart. Graph theoretical clustering based on limited neighbourhood sets. *Pattern Recognition*, 15(3):173 – 187, 1982. 2

[10] D. Wagner, A. Mulloni, T. Langlotz, and D. Schmalstieg. Real-time panoramic mapping and tracking on mobile phones. In *VR*, pages 211–218, 2010. 1, 7

[11] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(11):1101–1113, 1993. 2

[12] C. T. Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. In *IEEE Transactions on Computer, C-20 Issue 1*, pages 68–86. IEEE Computer Society, Jan. 1971. 2

[13] S. Zollmann, D. Kalkofen, E. Mendez, and G. Reitmayr. Image-based ghostings for single layer occlusions in augmented reality. In *International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, Oct. 2010. 1, 8

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

A Structure Based Mosaicking Approach for Aerial Images from Low Altitude of Non-Planar Scenes

Daniel Wischounig-Strucl Markus Quartisch
Bernhard Rinner
Institute of Networked and Embedded Systems
Klagenfurt University, Austria
`<firstname.lastname>@uni-klu.ac.at`

Abstract. *In this approach we estimate the depth structure of sceneries in aerial images captured by small-scale UAVs to improve the mosaicking of an orthographic overview image. Initial image transformations derived from inaccurate position and orientation data of UAVs are enhanced by the camera pose obtained using Structure from Motion. Corresponding points are then selected on a common ground plane to find accurate image transformations. The resulting mosaick preserves distances and minimizes distortions. A rough placement is immediately presented and optimized incrementally if more images are considered.*

1. Introduction

For many applications, such as disaster response, monitoring accident scenes and building sites, up to date and spatially accurate overview images are required. In particular, after severe disasters such as earthquakes or floodings wide area overviews are of special interest and importance to guide first-time responders.

We are investigating an approach to generate a wide area overview image from single images, preserving spatial distances as seen in orthophotos. To cover wide areas we favor aerial images from unmanned aerial vehicles (UAVs), because images from static cameras are hardly available due to the lack of infrastructure in typical scenarios.

For taking the essential aerial images small-scale UAVs, flying autonomously at low altitudes, are preferred to human operated planes or helicopters because of their advantages in availability, safety, robustness, ease of use and cost efficiency. Applying standard image registration algorithms to images from low altitudes, often lead to perspective distortions.

We achieve an overview image that can be perceived as orthophoto, if we only consider the planar ground and neglect objects on the ground. To keep the uniform scale in an orthophoto we have to optimize the image transformations accordingly. We do not aim to generate true orthophotos, which would require dense 3D models. Hence, images are taken with a nadir view, i.e., orthogonal to the earth's surface, to reduce the perspective influences of the non-planar scene and to allow a simplified orthorectification.

The ideal solution would be, of course, a full 3D reconstruction of the scene. But this is not feasible on small scale UAVs due to limitations of payload, battery capacity and computational performance. Furthermore, the resulting overview image should be presented iteratively as quick as possible. Thus, the images are processed already during flight of our networked small-scale UAVs and interim results are transmitted over the wireless channel with limited bandwidth.

In our approach, rough image transformations based on the metadata are refined by structure data from overlapping images. The Structure from Motion technique is used to compute the scene structure within overlapping regions to specifically match areas on the ground plane. For selecting corresponding points only on the ground plane it is necessary to apply a plane fitting algorithm to the structure data. With the resulting points an image transformation is computed that preserves distances while mosaicking.

Furthermore, the position and orientation data from the UAV's sensors is merged with the data extracted from images by Structure from Motion to estimate the real camera orientation and position. This allows a more accurate spatial referencing of points on the ground plane and refined orthorectification of single images.

The remainder of this paper is organized as follows: Section 2 gives a short overview on related work. Section 3 elaborates challenges and research questions of mosaicking aerial images incrementally and leads to Section 4, that proclaims our approach for mosaicking by means of the scene structure. Section 5 presents mosaicking results and finally Section 6 concludes the paper and gives some outlook on future work.

2. Related Work

In many cases single transformations applied to one image are sufficient to achieve appealing mosaicks. Recent works from Xing et al. [12] show satisfactory results when applying perspective transformations estimated by RANSAC (Random Sample Consensus) [1] and optimized SIFT (Scale Invariant Feature Tracker) features, taking images from airplanes.

Wang et al. combines orthorectified images with panorama images in [11] for 3D reconstruction of buildings, where the user has to select lines on the ground plane in the panorama images. The camera pose is computed from these lines on the ground plane, which represent footprints of buildings. The camera is kept at the same position and rotated to build a panorama image. After processing and manual optimization the ground images are projected on the proposed 3D model. For a larger area many panorama images are taken and processed one by one and are finally combined using bundle adjustment.

When considering hundreds of images with little overlaps, the initial image transformation is estimated by the metadata as proposed in [13]. The authors assume an exact nadir view of the camera onto a planar scene and neglect perspective distortions. Images annotated with metadata, i.e., altitude, global position and camera pose, are aligned by their global position. These transformations are refined afterwards by processing image correspondences.

In [14] the authors describe an effective method for combining data from images, taken from an airplane, with data from inertial sensors to achieve a seamless and geo-referenced mosaic. For the mosaicking the data from the inertial sensors and position sensors are combined with image features without 3D reconstruction or complex global registration. Aerial images from airplanes are made with telephoto lenses and from high distances to objects do not show perceptible perspective distortions.

Manually selected reference points on the ground are the base for a mosaicking approach presented in [9] that first extracts and matches feature points by Multi-Scale Oriented Patches (MOPs), clusters images, and finally uses RANSAC-initialized bundle adjustment to optimize all constraints over the entire image set. A simultaneous optimization balances the requirements of precise mosaicking and absolute placement accuracy on an overview image.

In our work we go one step further and introduce a basic structure and scene reconstruction with Structure from Motion to improve the metadata and image based mosaicking to deliver high resolution and frequently updated overview images.

3. Problem Definition

The goal is to mosaick a high resolution overview image from single aerial images and at the same time keep the uniform scale in the scenery. In order to generate this orthographic overview image, high resolution images are taken from multiple UAVs. Each image is annotated with metadata that contains position and orientation information, among others, from the UAV's sensors.

Creating a mosaick by simple placing images based on their metadata will lead to bad results, because this data is associated with uncertainty due to inaccuracy from the low cost and light weight design of small-scale UAVs. To cover wide areas from low altitudes, typically up to $150\,m$ above ground, with a minimum number of images it is obvious to use wide angle lenses. The tolerance of the image boundaries, projected on the ground, is in the range of $10\,\%$ of the image size, explored in detail in the work [13].

Hence, the challenge is to compute image transformations in the orthographic mosaick, while the non-planar scenery induces significant perspective distortions at individual images compared to aerial images taken from high altitudes. Moreover, a detailed 3D model of the scenery is not available.

We have to cope with several constraints, most prominent are the resource limitations. We cannot compute the whole overview image on the UAV nor transmit all high resolution images to the ground or other UAVs. For an online mosaicking a distributed processing is of interest, considering that high res-

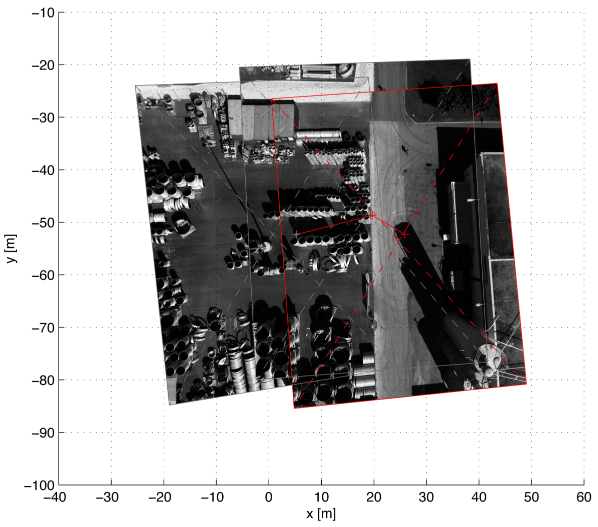


Figure 1. Initial image placement of images I_i by raw metadata where $i \in \{1, 2, 3\}$. Mosaicking errors can be explored on the ground plane. The trajectory of the UAV is shown in red.

olution images are not available immediately at the ground station.

To achieve a correct image placement that preserves distances within the overview image we need to estimate the camera position more accurately.

4. Structure Based Matching for Image Mosaicking

Our approach of mosaicking nadir aerial images annotated with metadata can be split into two main components:

I For the required online mosaicking the image transformations can be done with raw metadata without considering image contents.

II In parallel, these transformations can be refined as soon as more accurate camera extrinsics (position and orientation data), are estimated.

To improve the accuracy of the camera extrinsics from the metadata the Structure from Motion is used.

Hence, we model an optimization problem extending the two-step approach presented in [13] to find appropriate image transformations for each image in the set of aerial images. To avoid the accumulation of local perspective errors the metadata from cameras and the structure of the scene is taken into account.

4.1. Refined Estimation of Camera Extrinsics

In parallel to the rough placement, only by exploiting metadata, a refinement of the image transformation is executed as outlined in the following. Due to resource limitations the processing pipeline considers distributed execution; some processing steps can be executed directly on the UAV.

1. Determine a pair of images with sufficient overlap.
2. Match extracted feature points within the overlapping areas.
3. Use Structure from Motion to compute camera position and 3D structure for the matched feature points.
4. Merge the resulting camera extrinsics with the raw extrinsics and orthorectify both images.
5. Use plane fitting in the 3D structure to select feature points on the common ground plane and estimate the final image transformation.

Find a pair of images with sufficient overlap.

First the overlapping image areas O are determined by projecting the raw camera extrinsics from the metadata P_{IMU}, cf. Equation 17, onto the estimated ground plane. In Figure 1 the projection by the metadata and initial state for three images is presented before computing the refined transformations.

From all available pairs that overlap, a pair of images $\{I_i, I_j\}$ is selected to have the maximum overlapping area. Furthermore, for each image the features are extracted and the feature descriptor vectors δ_i and feature coordinates f_i are stored. For the following processing steps only the features, a few kilobyte in size, are necessary, instead of the whole image of up to 4 megabytes (compressed). This allows the reduction of the communication bandwidth significantly. In this approach we currently use the SIFT (Scale Invariant Feature Tracker) features [2], because it is has been proven to be very powerful [6].

$$\{\delta_i, f_i\} = \mathit{SIFT}_{\mathit{extract}}(I_i), \quad \{\delta_i, f_i\} \in \boldsymbol{F}_i \qquad (1)$$

Match extracted feature points within the overlapping areas.

Only features within the overlapping area $O_{i,j} = I_i \cap I_j$ are considered for the matching. This reduced feature set $\boldsymbol{F}'_i \subseteq \boldsymbol{F}_i$ for image I_i and $\boldsymbol{F}'_j \subseteq \boldsymbol{F}_j$ for image I_j in the overlapping image area $O_{i,j}$ are matched simply by a nearest neighbor search. The

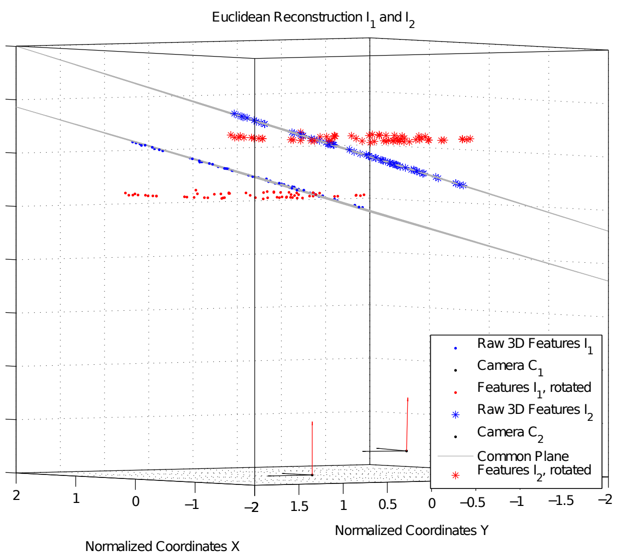


Figure 2. Matched features in the Euclidean scene reconstruction. Note, only the inliers on the same plane are plotted for a better visualization.

minimum Euclidean distance for the invariant feature descriptor vector δ_i' of feature $f_i' \in \boldsymbol{F}_i'$ is compared to a descriptor vector δ_j' of $f_j' \in \boldsymbol{F}_j'$ to find correspondences as suggested by Lowe [5].

$$\{\delta_i', f_i'\} \in \boldsymbol{F}_i', \quad \{\delta_j', f_j'\} \in \boldsymbol{F}_j' \tag{2}$$

$$\boldsymbol{M} = \{\boldsymbol{F}_i', \boldsymbol{F}_j' | f_i', f_j' \in O_{i,j}\} \tag{3}$$

$$\{\hat{f}_i, \hat{f}_j\} = \mathit{match}\left(\boldsymbol{F}_i', \boldsymbol{F}_j'\right) \tag{4}$$

Use Structure from Motion to compute camera position and 3D structure for the matched feature points.

From the matched features $\hat{f}_i, \hat{f}_j$ in the overlapping image area $O_{i,j}$ we compute the scene structure of these points by triangulation. Thus, the 3D structure, cf. Figure 2, i.e., elevation levels and the camera pose, is reconstructed by an estimation of the epipolar geometry [4]. The epipolar geometry, defined by the fundamental matrix F, essential matrix E, and the epipoles e_1 and e_2 is computed by Structure from Motion [3, 7]. Since we are using calibrated cameras, the camera calibration matrix K is known, the camera extrinsics P_{SfM_i}, cf. Equation 8, are determined by a singular value decomposition (SVD) from the essential matrix and epipoles [8].

$$E = [\hat{t}]_\times \hat{R} = U \Sigma V, \quad F = K^{-T} E K^{-1} \tag{5}$$

$$\hat{\boldsymbol{x}}_i^T E \hat{\boldsymbol{x}}_i = \boldsymbol{x}_i^T K^{-T} E K^{-1} \hat{f}_i = \boldsymbol{x}_i^T F \hat{f}_i \tag{6}$$

The essential matrix is estimated by using RANSAC within the matched features $\hat{f}_i$ and $\hat{f}_j$ to reduce outliers that do not match the approximated resulting essential matrix, cf. Equation 6.

In Figure 3 the structure inliers for each image are presented in the image plane. The point coordinates of selected feature points $\hat{f}$ in image I_i and I_j are mapped to 3D point coordinates $\boldsymbol{x} = [x, y, z]^T \in \mathbb{R}^3$. With the estimated camera extrinsics, cf. Equation 5, the Euclidian coordinates of the scene points $\boldsymbol{x}_i \in \boldsymbol{X}_i$ and $\boldsymbol{x}_j \in \boldsymbol{X}_j$ are reconstructed.

Merge the resulting camera extrinsics with the raw extriniscs.

The camera pose P_{SfM} from the image data is merged with the camera orientation and position P_{IMU} from the metadata. With the relative coordinates from Structure from Motion and the scaling from the metadata, the resulting camera extrinsics P_C are computed, cf. Equation 9. P_C describes the projective view of the camera that is used to transform images to their nadir view before the mosaicking. This process is known as orthorectification.

$$P_{\mathrm{IMU}_i} = [R_{\mathrm{IMU}_i}, T_{\mathrm{GPS}_i}]_{4\times 3} \tag{7}$$

$$P_{\mathrm{SfM}_i} = [\hat{R}_i, \hat{t}_i]_{4\times 3} \tag{8}$$

To project and maintain the spatial coordinates and distances on the ground plane the rotation component R_{C_i} of camera pose P_{C_i} is used. The optimized camera pose P_{C_i} replaces the first estimation from the raw metadata for image I_i.

$$P_{C_i} = [R_{C_i}, T_{C_i}]_{4\times 3} \tag{9}$$

Fitting a ground plane into the 3D structure

A subset of points from the 3D points $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$ is adjudged as optimum for the final image transformation computation by the following constraint: All points on the same elevation level, respectively plane, preserve spatial relations with the image transformation $T_{\mathrm{match},i}$, cf. Equation 15. Hence, it is important to find those points that avoid perspective distortions and inaccurate distances in the final mosaicking stage.

Inliers on the common plane $\boldsymbol{X}_\Pi$ are determined from the structure points in $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$ by fitting a plane to all available points with RANSAC. The fitting function for RANSAC is the plane function for plane Π in Equation 10, that is further optimized to be the most perpendicular plane to the camera's principal axis. Therefore, the angle between the plane normal vector $\vec{\boldsymbol{n}}$ and the principle axis vector $\vec{\boldsymbol{p}}$, derived from P_{C_i}, is minimized, assuming a horizontal ground plane.

$$\begin{aligned} \Pi &= \vec{\boldsymbol{n}} \cdot \boldsymbol{q} \qquad \arccos(|\vec{\boldsymbol{n}}| \cdot |\vec{\boldsymbol{p}}|) \leq \varepsilon & (10) \\ \vec{\boldsymbol{n}} &= (\boldsymbol{x}'_2 - \boldsymbol{x}'_1) \times (\boldsymbol{x}'_3 - \boldsymbol{x}'_1) & (11) \\ \boldsymbol{X}_\Pi &= \{\boldsymbol{x}'_1, \boldsymbol{x}'_2, \boldsymbol{x}'_3\} \in \boldsymbol{X} & (12) \end{aligned}$$

At least the three points defining the plane are sufficient to compute the matching transformation $T_{\text{match},i}$ in the order of a similarity transformation. For an improved matching function, e.g., by estimation and fitting again with an approximation approach, additional points $\boldsymbol{x}'_i$ can be selected by their closest distance d to the plane within a certain threshold γ.

$$\begin{aligned} d = |\vec{\boldsymbol{n}} \cdot \vec{\boldsymbol{v}}| \qquad \vec{\boldsymbol{v}} = \boldsymbol{x}' - \boldsymbol{q} && (13) \\ \boldsymbol{x}'_i \in \boldsymbol{X}_\Pi \quad | \quad d \leq \gamma && (14) \end{aligned}$$

The matching transformation T_{match} applied to the whole image is computed by the normalised direct linear transformation algorithm given by Hartley and Zisserman [3].

$$\boldsymbol{x}' = T_{\text{match}}\, \boldsymbol{x} = [sR, t]_{3\times 3}\, \boldsymbol{x} \qquad (15)$$

4.2. Incremental Mosaicking

After refining the image transformations and camera poses with the structure base matching the inaccurate mosaicking from raw data can be improved as expressed in the following.

Raw mosaicking with camera extrinsics

Single images I_i are merged with function $\biguplus$ to the overview image I, cf. Equation 16. Hence, the merging function $\biguplus$ is an arbitrary image fusion function. For demonstration we use a simple overlay function with alpha-blending. Initially images are placed by transformations derived from P_{IMU}, cf. Equation 17, based on their annotated GPS and IMU data. The images are orthorectified by the projective transformation $\tilde{R}_i$ and placed on the overview image by the transformation $T_{\text{pos},i}$ (cf. Figure 1).

$$I = \biguplus_{i=1}^{n(t)} T_i I_i \qquad (16)$$

$$P_{\text{IMU}_i} = [R_{\text{IMU}_i}, T_{\text{GPS}_i}]_{4\times 3} \Rightarrow \{\tilde{R}_i, T_{\text{pos},i}\} \qquad (17)$$

Refine the mosaicking with the output from the structure based matchting

Next, the refinement of the global mosaicking is achieved by the structure based matching, as described in Section 4.1. The optimized camera extrinsics matrix P_{C_i}, now improves the orthorectification of each image, opposed to $\tilde{R}_i$. Furthermore, the initial placement by $T_{\text{pos},i}$ is enhanced to the image alignment based on the scene structure.

Finally, the images are mosaicked with neighboring images by the transformation $T_{\text{match},i}$ that is approximated to optimize the output quality within the reduced search space in the overlapping image areas.

Hence, omitting perspective distortions that may propagate over images is one benefit of using projective transformations only for single images. When aligning individual images I_i to an overview image I by $T_{\text{match},i}$ only lower order transformations like the similarity transformation are allowed.

The resulting optimized image transformation T_i applied in the final mosaicking stage, cf. Equation 18, is composed from the raw metadata position and structure based transformation. The perspective projection R_{C_i} derived from the camera's intended pose P_{C_i} orthorectifies the image into nadir view, while the global alignment is applied with the refined global position T_{C_i}.

$$T_i = R_{C_i} \cdot T_{C_i} \cdot T_{\text{match},i} \qquad (18)$$

5. Preliminary results

In the current state of evaluations the method of SIFT feature extraction is used for finding correspondences. However, the used feature extraction and matching methods are exchangeable, but SIFT shows sufficiently good results for our approach. The features are extracted from a copy of each image I_i, that is downscaled to 816×612 pixels.

In Figure 2 the result of the Structure from Motion point reconstruction in the overlapping area is presented. Note, only points on the common plane $\{\boldsymbol{x}'_i, \boldsymbol{x}'_j\} \in \boldsymbol{X}_\Pi$ and the two cameras P_{C_i}, P_{C_j} are plotted for better visualization. Figure 4 shows the finally transformed image I_i on the previous overview image. Image I_i and image I_j of the current test set $\boldsymbol{I}$ where $i = 1, j = 2$ are orthorectified by R_{C_i}, R_{C_j} derived from P_{C_i}, P_{C_j} beforehand. The selected features on the common plane are marked with red and blue crosses.

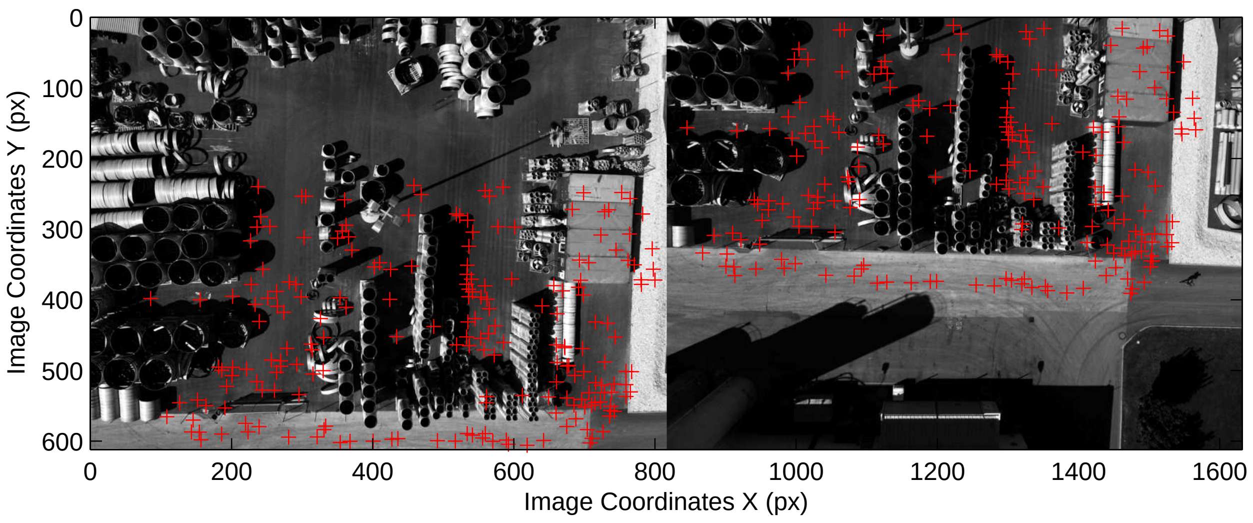


Figure 3. Image I_1 (left) and Image I_2 (right) with red markers on the remaining inliers from the Structure from Motion in the overlapping image region. These points show the input $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ for the plane fitting.

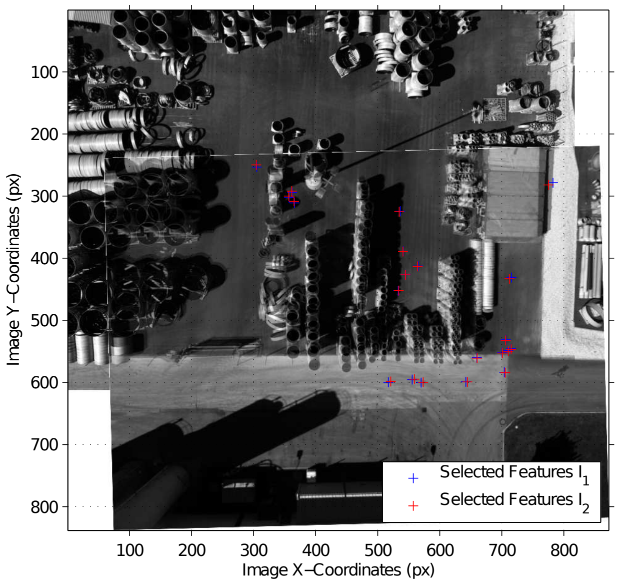


Figure 4. Matched features on the same plane in image I_1 and I_2

In the next iteration with the increased set of images the image I_3 has the maximum overlap with image I_2. The overview image presented in Figure 5 shows the previously mosaicked images I_1 and I_2 and the newly transformed image I_3 mosaicked on top. The red and blue markers show the common plane points from $\{\boldsymbol{x}'_2, \boldsymbol{x}'_3\} \in \boldsymbol{X}_\Pi$ again.

Moreover, in Table 1 the evolution of the features used for the final transformation optimization is presented where the significant reduction of the plane inliers to 21 in $I_1 \cap I_2$ and 13 in $I_2 \cap I_3$ can be explored.

For each image and every pair of images the quality function Q is evaluated and its result is presented in Figure 6(a) for I_1, I_2 and Figure 6(b) for I_2, I_3. Figure 6 shows the pixel deviation of the inliers $\boldsymbol{x}'_i$ on the ground plane, which can be directly transformed to spatial deviations when projecting with the camera position and pose P_{C_i} and P_{C_j}. The correlation error for those ground plane inliers shows excellent results in a radius $r = 5$ pixels.

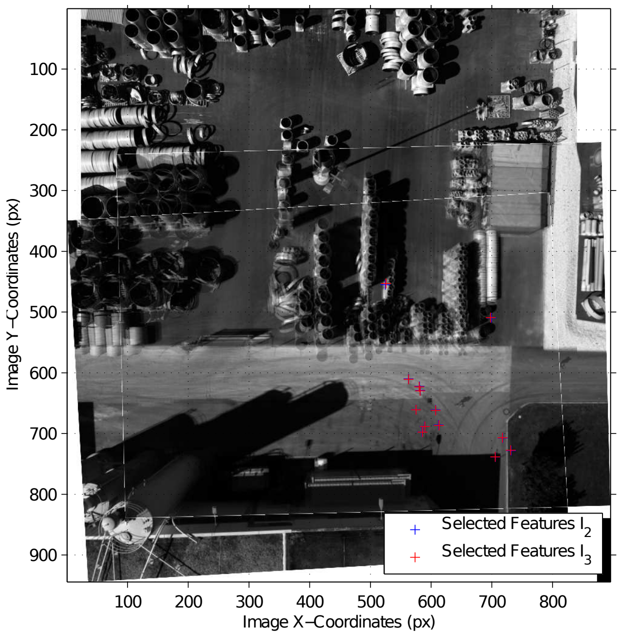
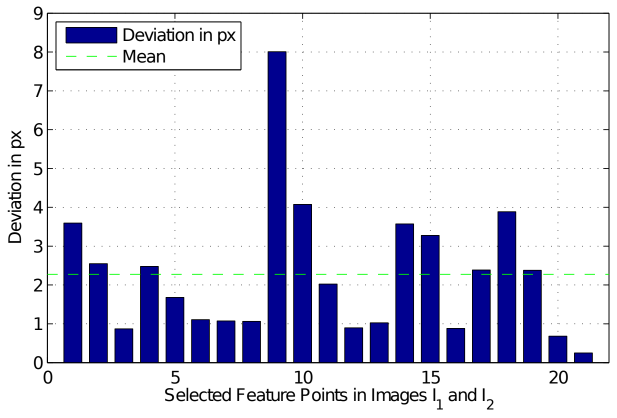


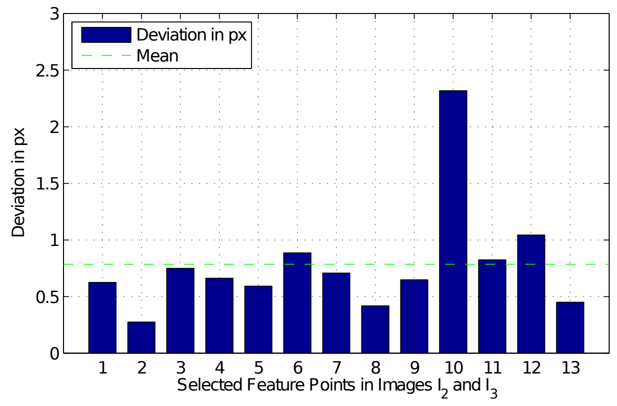
Figure 5. Matched features on the same plane in image I_2 and I_3 on top of image I_1. The correlation on the ground plane is excellent compared to the distortion effects of objects in the scene.

Processing Stage	I_1	I_2	I_3
Feature Extraction	1660	1492	1518
Reduced Search Range I_1, I_2	342	311	
Correlation Matching I_1, I_2	247	247	
SfM inliers I_1, I_2	201	201	
Plane Fitting I_1, I_2	21	21	
Reduced Search Range I_2, I_3		568	602
Correlation Matching I_2, I_3		548	548
SfM inliers I_2, I_3		483	483
Plane Fitting I_2, I_3		13	13

Table 1. The number of feature points can be significantly reduced from considering only overlapping regions to inliers on the same plane.



(a) Spatial distance error in pixels after transformation of image I_2 on image I_1



(b) Spatial distance error in pixels after transformation of image I_3 on image I_2

Figure 6. Distance deviations of points on the ground plane in the final mosaick.

Transformation Quality

The quality function Q weights the spatial accuracy function $G_i(I_i, I)$ and the pixel correlation function $C_i(I_i, I)$ by $\alpha, (0 \le \alpha \le 1)$ defined in Equation 19. The distance function of a projected feature point $\boldsymbol{x}_i$ on the ground plane of image I_i to the corresponding feature point on the overview image I is denoted by d and c measures the pixel correlation in a small neighborhood r of the feature point coordinate to the corresponding area on the overview image I.

$$Q = \sum_{i=1}^{n} (\alpha\, G_i(I_i, I) + (1-\alpha) C_i(I_i, I)) \tag{19}$$

$$\begin{aligned} G_i(I_i, I) &= \frac{1}{m} \sum_{k=1}^{m} d(\boldsymbol{x}_k \in I_i, I) && (20) \\ C_i(I_i, I) &= \frac{1}{m} \sum_{k=1}^{m} c(\boldsymbol{x}_k \in I_i, I, r) && (21) \\ & \quad |r = \beta\, size(I_i) \end{aligned}$$

6. Conclusion and Future Work

In this approach we have shown that distorted aerial images from low altitudes and taken with wide angle lenses can still be used to build an orthographic overview image that preserves a uniform scale on the ground plane. We compute the structure of the scene with Structure from Motion and optimize a rough mosaicking from annotated metadata of the images, i.e., GPS and IMU data of the UAV, to an accurate mosaick with matched correspondences on a common ground plane.

In this work, the results from Structure from Motion are only used to find a common plane and to enhance the estimation of the camera pose. This improves the spatial projection on the ground plane and delivers more accurate image transformations. We have experienced that the computational effort is significantly reduced when limiting the search range to structure inliers on the same plane and determining corresponding images from a large set by their proposed positions.

We will further analyze enhanced Structure from Motion estimation algorithms and optimization strategies for fitting common planes in adjacent images in the 3D domain. The reconstruction of the 3D structure of the scene can be further optimized by bundle adjustment [10]. We will investigate whether this method will get along with the available resources.

In future steps this additional knowledge about the scene could be used to generate a detailed depth model or mark objects in the scene.

Acknowledgment

This work was performed in the project *Collaborative Microdrones (cDrones)* of the research cluster Lakeside Labs and was partly funded by the European Regional Development Fund, the Carinthian Economic Promotion Fund (KWF), and the state of Austria under grant KWF-20214/17095/24772.

References

[1] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24:381–395, June 1981. 2

[2] P. Hansen, P. Corke, W. Boles, and K. Daniilidis. Scale invariant feature matching with wide angle images. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2007.*, pages 1689–1694, Nov. 2007. 3

[3] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004. 4, 5

[4] C. Loop and Z. Zhang. Computing rectifying homographies for stereo vision. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, page 131, 1999. 4

[5] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 4

[6] O. G. Luo Juan. A Comparison of SIFT, PCA-SIFT and SURF. *International Journal of Image Processing IJIP*, 3(4):143–152, 2009. 3

[7] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry. *An invitation to 3D vision, from images to models*. Springer Verlag, 2003. 4

[8] D. Nister. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):756–770, June 2004. 4

[9] P. Pesti, J. Elson, J. Howell, D. Steedly, and M. Uyttendaele. Low-cost orthographic imagery. In *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, GIS '08, pages 24:1–24:8, New York, NY, USA, 2008. ACM. 2

[10] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, ICCV '99, pages 298–372, London, UK, 2000. Springer-Verlag. 7

[11] L. Wang, S. You, and U. Neumann. Semi-automatic registration between ground-level panoramas and an orthorectified aerial image for building modeling. In *IEEE 11th International Conference on Computer Vision, ICCV 2007.*, pages 1–8, Oct. 2007. 2

[12] C. Xing and J. Huang. An improved mosaic method based on SIFT algorithm for UAV sequence images. In *Proceedings of International Conference on Computer Design and Applications (ICCDA)*, volume 1, pages V1–414–V1–417, June 2010. 2

[13] S. Yahyanejad, D. Wischounig-Strucl, M. Quaritsch, and B. Rinner. Incremental Mosaicking of Images from Autonomous, Small-Scale UAVs. *7th IEEE International Conference on Advanced Video and Signal-Based Surveillance*, Sept. 2010. 2, 3

[14] Z. Zhu, E. Riseman, A. Hanson, and H. Schultz. An efficient method for geo-referenced video mosaicing for environmental monitoring. *Machine Vision and Applications*, 16:203–216, 2005. 2

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

An Algebraic Approach to Camera Calibration Based on Distance Constraints

Tanja Schilling and Tomáš Pajdla
Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University in Prague
schillin@cmp.felk.cvut.cz, pajdla@cmp.felk.cvut.cz

Abstract. *In this paper, we propose an approach to an algebraic computation of a projective transformation matrix that upgrades a 3D reconstruction. Constraints are derived from images of segments of equal lengths and yield a set of polynomial equations which we try to solve by the means of Gröbner bases. Since a straightforward computation is infeasible for this problem, a strategy is introduced in which a Gröbner basis is constructed for a special template data first and the actual data is processed afterwards according to the resulting procedure template. We present experiments that encourage the assumption that this method is applicable in general.*

1. Introduction

The presented approach is motivated by problems arising in the field of medical robotics and image-guided surgery. Within that scope, accurate and robust 3D reconstruction and tracking of deformable tissue is a fundamental but difficult task, because such surfaces often lack distinctive features and detailed texture or they are partially covered by liquids. Various approaches to 3D tissue deformation recovery have been published [15, 18, 14]. Some of them apply optical markers and thus require special surgical equipment to cope with the mentioned issues. But as space is strongly limited in this environment, methods without the need of extra instrumentation are preferable.

The instruments used for the particular surgical procedure are anyway present in the scene and almost always visible. They are rigid and mostly well detectable. Consequently, it seems utile to benefit from these properties in order to improve the tissue reconstruction. Though, relying on exact dimensions of the tools is not advisable, because detailed specifications of commercial items are hardly to get and the instruments are changed quite frequently. Often, various scissor-like instruments are applied whose two legs have equal lengths. From this fact a constraint could be derived in order to upgrade a preliminary projective reconstruction and the corresponding camera projection matrix.

The approach in [13] is based on that idea. Therein, the authors introduce a method to reconstruct dynamic articulated structures from multiple uncalibrated views assuming affine cameras. In particular, they aim to track human motion in sports broadcasts. An affine 3D reconstruction is computed first. Constraints are derived from the fact that the lengths between rotational joints of the body remain constant over time. As parallel projection is assumed, these equations are linear and can be solved via singular value decomposition. Although the affine camera model is a reasonable assumption in recordings of sport events where long focal lengths are used, it is not applicable to the case of endoscopy where the observed surface is located very close to the camera.

Assuming perspective projection, the constraints derived from equal lengths of observed objects constitute a system of non-linear algebraic equations. Polynomial systems occur in various computer vision problems and many of them have been solved by means of Gröbner bases [7, 8, 17]. But there is no easy, straightforward method to solve general polynomial systems efficiently and robustly. Instead, each particular problem usually requires the manual design of a suitable Gröbner basis solver. However, an automatic generator of minimal problem solvers was presented in [12]. But it turned out to be inapplicable to solve the present problem in preliminary tests.

In this paper, we present ongoing work to find algebraically a transformation to upgrade a reconstruction based on constraints derived from equally distant pairs of points. The following section gives a description of the problem. Section 3 briefly introduces the notion of Gröbner bases. In section 4, we explain a possible strategy to solve the problem by constructing a Gröbner basis for template data first and processing the actual data according to the resulting procedure template afterwards. We give an overview of the experiments we have performed so far in section 5 and summarize our conclusions in section 6.

2. Problem formulation

The image projection of a scene point X_i, represented by its homogeneous coordinates [10], is denoted as

$$x_i \propto \mathbf{P}X_i, \tag{1}$$

i.e. an $\alpha_i \in \mathbb{R}$ exists, such that $x_i = \alpha_i \mathbf{P} X_i$.

Let us assume that we have measured n image coordinates $\hat{x}_i \not\propto x_i$, $i \in \mathbb{N}$, and computed $\hat{\mathbf{P}}$, $\hat{\mathbf{P}} \not\propto \mathbf{P}$, as well as $\hat{X}_i \not\propto X_i$, from those points, such that

$$\hat{x}_i \propto \hat{\mathbf{P}}\hat{X}_i \tag{2}$$

Now, we want to find a non-singular matrix $\mathbf{H} \in \mathbb{R}^{4\times4}$ which upgrades $\hat{X}_i$ and $\hat{\mathbf{P}}$ by a projective transformation, such that

$$\mathbf{P}X_i \propto \hat{\mathbf{P}}\mathbf{H}^{-1}\mathbf{H}\hat{X}_i \tag{3}$$

and

$$X_i \propto \mathbf{H}\hat{X}_i. \tag{4}$$

In order to reduce the number of unknowns in $\mathbf{H}$, we have to fix the reference frame. For this purpose, we choose three of the preliminarily reconstructed points $\hat{X}_i$ to determine the origin, the x-axis and the the xy-plane. From now on, let us assume that all points $\hat{X}_i$ have already been mapped by a similarity transform such that there exist three points $(0,0,0,1)^\top$, $(\hat{x}_i,0,0,1)^\top$ and $(\hat{x}_j,\hat{y}_j,0,1)^\top$, $\hat{x}_i,\hat{x}_j,\hat{y}_j \in \mathbb{R}\setminus\{0\}$, which determine the coordinate frame.

In order to map the point of origin to itself, $(0,0,0,1)^\top \propto \mathbf{H}(0,0,0,1)^\top$, points on the x-axis to the x-axis, $(x_i,0,0,1)^\top \propto \mathbf{H}(\hat{x}_i,0,0,1)^\top$, and points in the xy-plane to the xy-plane again, $(x_j,y_j,0,1)^\top \propto \mathbf{H}(\hat{x}_j,\hat{y}_j,0,1)^\top$, the transformation matrix has to be of the form

$$\mathbf{H} = \begin{pmatrix} h_1 & h_2 & h_3 & 0 \\ 0 & h_4 & h_5 & 0 \\ 0 & 0 & h_6 & 0 \\ h_1 - h_9 & h_7 & h_8 & h_9 \end{pmatrix}. \tag{5}$$

Two constraints on $\mathbf{H}$ easily arise from that. First, the projection matrix has to be invertible and therefore $\mathbf{H}$ must fulfill

$$0 \neq \det(\mathbf{H}) = h_1h_4h_6h_9. \tag{6}$$

Secondly, we want to avoid solutions that yield points at infinity. Hence, it has to be

$$0 \neq X_{4i} = \mathbf{H}_4\hat{X}_i, \quad \forall i \in \mathbb{N}, i \leq n \tag{7}$$

with $\mathbf{H}_4$ denoting the 4-th row of $\mathbf{H}$ and X_{4i} describing the 4-th element of X_i.

Let us now assume that there are two pairs of points, $(X_i, X_{i'})$ and $(X_j, X_{j'})$, with equal Euclidean distances between the points of each pair $\|X_i - X_{i'}\| = \|X_j - X_{j'}\|$, and the indices i, i', j, j', where $i \neq i', j \neq j'$ and $i' \neq j'$, are known to us. Replacing X_i by $\mathbf{H}\hat{X}_i$, yields the following constraint on $\mathbf{H}$:

$$\begin{aligned} 0 = &(\mathbf{H}_4\hat{X}_j)^2(\mathbf{H}_4\hat{X}_{j'})^2 \sum_{l=1}^{3}(\mathbf{H}_l\hat{X}_i\mathbf{H}_4\hat{X}_{i'} - \mathbf{H}_4\hat{X}_i\mathbf{H}_l\hat{X}_{i'})^2 \\ &- (\mathbf{H}_4\hat{X}_i)^2(\mathbf{H}_4\hat{X}_{i'})^2 \sum_{l=1}^{3}(\mathbf{H}_l\hat{X}_j\mathbf{H}_4\hat{X}_{j'} - \mathbf{H}_4\hat{X}_j\mathbf{H}_l\hat{X}_{j'})^2 \end{aligned} \tag{8}$$

where $\mathbf{H}_l$ denotes the l-th row of $\mathbf{H}$.

Equation (8) constitutes a homogeneous polynomial of degree 8 in 9 variables, with 808 terms in the general case. Therefore, at least 9 quadruples $(X_i, X_{i'}, X_j, X_{j'})$ are required, to obtain the 9 equations which determine $\mathbf{H}$. Introducing two more variables h_{10} and h_{11}, we can rewrite the inequalities (6) and (7) as equalities [5]

$$0 = 1 - h_1h_4h_6h_9h_{10} \tag{9}$$

$$0 = 1 - h_{11}\prod_{i=1}^{n}\mathbf{H}_4\hat{X}_i \tag{10}$$

Now, the problem is to solve the system of $m = m' + 2$ algebraic equations, $m' \geq 9$, in 11 variables $h_1, \ldots, h_m$,

$$0 = f_1(\mathbf{h}) = \cdots = f_m(\mathbf{h}). \tag{11}$$

3. Gröbner bases

Systems of polynomial equations can be solved efficiently by means of Gröbner bases [9]. F denotes the set of m polynomials $F = \{f_1(\mathbf{h}), \ldots, f_m(\mathbf{h}) | f_i(\mathbf{h}) \in K[h_1, \ldots, h_n]\}$ in n variables $\mathbf{h} = (h_1, \ldots, h_n)$ over a field K. The ideal $I = \langle F \rangle$ generated by F is the set of all polynomial linear combinations

$$I = \left\{ \sum_{i=1}^{m} f_i(\mathbf{h})q_i(\mathbf{h}) | q_i(\mathbf{h}) \in K[h_1, \ldots, h_n] \right\}. \tag{12}$$

A Gröbner basis is a special set of generators with desirable algorithmic properties. In particular, a Gröbner basis of an ideal I has the same set of solutions as I. But similar to a system of linear equations after Gaussian elimination, the solutions of I can be easily identified in the corresponding Gröbner basis w.r.t. a lexicographical monomial ordering [9].

Theoretically, the Gröbner basis can be computed from any generating set of I by a method called Buchberger's algorithm [9]. The basic mechanism is to take each pair $(f_i(\mathbf{h}), f_j(\mathbf{h}))$ from F, $f_i(\mathbf{h}) \neq f_j(\mathbf{h})$, compute its S-polynomial (see appendix), reduce it by F and add the remainder to F if it is not zero. This is done until the S-polynomials of all pairs in F reduce to zero.

Unfortunately, this problem is known to be EXPSPACE-complete [11]. Nevertheless, much better bounds can be found for many cases that actually occur in practice and several well-known methods exist to improve the basic algorithm. But they do not guarantee that a given practical problem can be solved within the limits of available memory. Our first attempts to solve the system of polynomial equations with standard methods for Gröbner bases computations in Maple 12, Macaulay2 [3] and SINGULAR [4] were not successful due to a lack of memory after some time of computation.

4. Strategy to solve via template data

As a straightforward computation of a Gröbner basis from the set of equations introduced in section 2 is not feasible, an alternative strategy to solve this problem has to be found. We propose to construct a Gröbner basis for a simpler template set of polynomials first. These polynomials are generated in the same way as explained above and differ from the original set only in its coefficients. The underlying data determining the coefficients must be chosen in such a way that the construction of the Gröbner basis from that template set can be done within a reasonable amount of time.

We assume that the sequence of operations to construct the Gröbner basis is basically identical for different sets of polynomials, given that the equations in those sets contain the same monomials and differ only in their coefficients [19]. With this assumption, we rely on the fact that Buchberger's algorithm and its improved variants do not consider the values of non-zero coefficients for the choice of critical pairs, the detection of unnecessary pairs or the selection of reductors.

During the computation of the template set we log which pairs effectively contribute to the final basis, i.e. those pairs that form S-polynomials which later do not reduce to zero. In the second step, the Gröbner basis is constructed from the original set. But S-polynomials are only computed for those pairs identified before. Thus, a lot of computation time usually wasted for reducing S-polynomials to no avail can be saved. Succeeding in that, we will be able to solve systems of polynomial equations that cannot be computed by direct construction of a Gröbner basis up to now.

The crucial point in this scheme is to find an appropriate template system that is simple enough to be feasible yet general enough to be used for the original problem. The difficulty is in the fact that having identical monomials in the template polynomials and in the original is required to achieve the same sequence of computation but does not necessarily lead to the desired result.

To generate the template set, we simplified the general problem by using small integers in a finite field $\mathbb{Z}_p$ instead of real numbers as point coordinates. Moreover, we found that a suitable template system can be derived from integer points with equal distances $\|X_i - X_{i'}\|$ and $\left\|X_j - X_{j'}\right\|$ being also integer numbers. That means each pair of points $(X_i, X_{i'})$ and $(X_j, X_{j'})$ has to fulfill

$$d^2 = \|X_i - X_{i'}\|^2 = \|X_j - X_{j'}\|^2 = a^2+b^2+c^2, \quad (13)$$

where $a, b, c, d \in \mathbb{Z}$, $d \neq 0$, thus forming a Pythagorean quadruple if $a, b, c \in \mathbb{Z} \setminus \{0\}$ or a Pythagorean triple respectively if $a = 0$, $b \neq 0$ and $c \neq 0$. The possibility to use such a special polynomial system as a template for the general problem is justified by the fact that every triple in $\mathbb{R}^3$ has a sufficiently precise scaled representation as a Pythagorean triple in $\mathbb{Z}^3$ [16]. Hence, we are looking for a generic Pythagorean case which is feasible to compute and at the same time implementable for a wide range of practically occurring systems originating from real coefficients. Experiments to find such cases are presented in the next section.

5. Experiments

The construction of a Gröbner basis for such a Pythagorean case to generate a template as explained above is still time consuming. Previously, we conducted some experiments where we used the *slimgb* algorithm incorporated in SINGULAR, an open source computer algebra system for polynomial computations, to construct Gröbner bases from several template systems. Those computations took about 15 minutes for each template system. However, SINGULAR turned out to be unsuitable to adopt the proposed strategy.

Consequently, we implemented our own version of the *slimgb* algorithm, described in [6], in C++ using CoCoALib-0.99 [1], a library for computations in commutative algebra. As the current implementation is not as optimized as the one in SINGULAR, computation time amounts to approximately 75 minutes for the same template systems. Thus, optimizing the implementation suggests itself. Before we spend a lot of time doing this, presumably trading in clarity of the code and possibilities to adapt it easily to new ideas for faster computation, we want to be sure that these efforts are likely to lead to a practicable method.

For this reason, simpler constraints are used for a start to provide an evidence that the strategy outlined in the previous section is effectively applicable. Instead of assuming equal lengths of two segments, we act on the assumption of known distances between the two points of each pair (X_i, Y_i), $\|X_i - Y_i\|^2 = d_i^2$ for $i = 1, \ldots, N$.

That simplifies equation (8) to

$$\begin{aligned} f_i(\mathbf{h}) &= \sum_{l=1}^{3} (\mathbf{H}_l \hat{X}_i \mathbf{H}_4 \hat{Y}_i - \mathbf{H}_4 \hat{X}_i \mathbf{H}_l \hat{Y}_i)^2 - (\mathbf{H}_4 \hat{X}_i \mathbf{H}_4 \hat{Y}_i)^2 d_i^2 \\ &= 0 \end{aligned} \quad (14)$$

yielding a homogeneous polynomial of degree 4 in 9 variables with 97 terms in the general case.

The additional two constraints, which ensure that $\mathbf{H}$ is non-singular and points at infinity are avoided, are applied as in equations (9) and (10)

$$f_{N+1}(\mathbf{h}) = 0 = 1 - h_1 h_4 h_6 h_9 h_{10} \quad (15)$$

$$f_{N+2}(\mathbf{h}) = 0 = 1 - h_{11} \prod_{i=1}^{n} \mathbf{H}_4 \hat{X}_i \mathbf{H}_4 \hat{Y}_i. \quad (16)$$

All 6 experiments presented here follow the same workflow. In the first step of each experiment, ten template sets F'_k and test sets F_k are generated, $k = 1, \ldots, 10$. Each template set F'_k contains 17 polynomials $f'_i(\mathbf{h})$ that are constructed from template data comprising 15 pairs of points (X'_i, Y'_i) and a matrix $\mathbf{H}'$ according to equation (5). In general, this data is generated from random numbers but some restrictions apply.

In order to determine the coordinate frame, the first two pairs of points must have the form

$$X'_1 = (0,0,0,1)^\top, \qquad Y'_1 = (y'_{1,1}, 0, 0, 1)^\top \quad (17)$$

$$X'_2 = (0,0,0,1)^\top, \qquad Y'_2 = (y'_{1,2}, y'_{2,2}, 0, 1)^\top, \quad (18)$$

whereas the remaining 13 pairs are

$$X'_i = (x'_{1,i}, x'_{2,i}, x'_{3,i}, 1)^\top, \;\; Y_i = (y'_{1,i}, y'_{2,i}, y'_{3,i}, 1)^\top \quad (19)$$

with $x'_{l,i} \neq 0$ and $y'_{l,i} \neq 0$ for all $l = 1, 2, 3$ and $i = 1, \ldots, 15$.

Additionally, each pair (X'_i, Y'_i) must form a Pythagorean quadruple $(x'_{1,i} - y'_{1,i}, x'_{2,i} - y'_{2,i}, x'_{3,i} - y'_{3,i}, d)$, such that

$$d_i'^2 = \|X'_i - Y'_i\|^2 = \sum_{l=1}^{3} (x'_{l,i} - y'_{l,i})^2 \quad (20)$$

where $d'_i \in \mathbb{Z} \setminus \{0\}$ for all $i = 1, \ldots, 15$.

Furthermore, the pairs of points as well as the matrix $\mathbf{H}'$ have to be selected in such way that the 15 polynomials $f'_i(\mathbf{h}) \in F'_k$ resulting from equation (14) using the pairs of points $(\hat{X}'_i, \hat{Y}'_i)$, where $\hat{X}'_i = \mathbf{H}'^{-1} X'_i$ and $\hat{Y}'_i = \mathbf{H}'^{-1} Y'_i$ for $i = 1, \ldots, 15$, have the maximum number of terms which is 3 if $i = 1$, 9 if $i = 2$ and 97 otherwise. Together with the two polynomials that we get from equations (15) and (16), this constitutes a set of 17 template polynomials $F'_k = \{f'_1(\mathbf{h}), \ldots, f'_{17}(\mathbf{h})\}$.

Basically, the test sets F_k are generated in the same way from test data as explained above for the case of template sets. But there are two major differences. One distinction is that the pairs of points (X_i, Y_i) in the test data are not required to form Pythagorean quadruples according to equation (20). The other difference concerns the underlying field from which random numbers are generated. For the template data, all random numbers are integers, such that $x'_{l,i}, y'_{l,i}, h'_j \in \mathbb{Z}$ for all $l = 1, 2, 3$, $i = 1, \ldots, 15$ and $j = 1, \ldots, 11$. This does not necessarily hold for generating test data as random numbers are rational in some experiments. However, the interval of admissible numbers is equally limited in both cases, i.e. $x'_{l,i}, y'_{l,i}, x_{l,i}, y_{l,i} \in [-20, 20]$ for all $l = 1, 2, 3$ and $i = 1, \ldots, 15$, and $h'_j, h_j \in [-10, 10]$ with $h'_j \neq 0$ and $h_j \neq 0$, for all $j = 1, \ldots, 11$.

In the second step of each experiment, a reduced Gröbner basis G'_k is constructed for the ideal generated by each template set F'_k. The algorithm corresponds mostly to the one in [6] and is implemented in C++ using CoCoALib-0.99 [1]. The computation of coefficients is done in the finite field $\mathbb{Z}_p$ with

$p = 3322513141$ as this proved to be a sufficiently large prime number in earlier experiments.

During Gröbner basis construction, the procedure template is recorded. In that, the pairs of polynomials whose S-polynomial does not reduce to zero are logged, as well as the new position of the reduced S-polynomial in the ordered basis. In that way, time is saved later during the computation of the test data according to this template because the reduction of S-polynomials that do not contribute to the result is avoided. Furthermore, the reduced polynomials do not have to be compared to the ones already in the current basis (which contains up to 834 polynomials in these experiments) to find the appropriate position.

Subsequently, all test sets F_k, $k = 1, \dots, 10$, are processed according to the template created from F'_{k^*} for several $k^* \in \{1, \dots, 10\}$. In this step, coefficients are in principle computed in $\mathbb{R}$. Though in practice, exact computation in $\mathbb{R}$ is impossible and additional difficulties arise from accumulating rounding errors of floating point arithmetic. As a consequence, some coefficients are very small but not exactly zero causing terms not to cancel out each other correctly. In principle, this can be avoided by memorizing when which coefficients get zero during the initial Gröbner basis construction from template data. Then, the corresponding coefficients are set to zero during Gröbner basis construction of the test data according to the recorded scheme, following a simplified version of the proposition in [20].

However, this would take a considerable amount of memory and computation time. Hence, we favour the more practicable way of simply processing the template data once again, simultaneously with the test data. Whenever a coefficient of the template data (computed in $\mathbb{Z}_p$) becomes zero, the corresponding coefficient in the test data is set to zero, too. Since the CoCoALib provides no means to realize this technique of computing "shadow" coefficients, we had to implement the algorithm for Gröbner basis construction from scratch without that library. To be able to handle large integers and high precision floating point coefficients, we employ the GNU Multiple Precision Arithmetic Library [2] in our implementation.

In the final step of each experiment, the reduced Gröbner basis G_{k,k^*}, that was constructed from the test set F_k using the template generated with template set F'_{k^*}, is evaluated. As we created the test data artificially, we know the desired values for the actual unknowns $h_1, \dots, h_{11}$. If these given $h_1, \dots, h_{11}$ are in the solution set of G_{k,k^*}, i.e. if all $g_j(\mathbf{h}) = 0 \ \forall g_j(\mathbf{h}) \in G_{k,k^*}$ when those values are inserted into the polynomials $g_j(\mathbf{h})$, then F'_{k^*} constitutes a suitable template set for the test set F_k. If this holds for all test sets F_k, especially for all $F_k \neq F'_{k^*}$ in the considered experiment, we call the template generated by F'_{k^*} an adequate template.

As our ultimate goal is to solve many general real-world cases, we are looking for templates that can be applied to correctly construct Gröbner bases from as many test sets as possible. The purpose of the following experiments is to verify that we can create adequate templates from simplified Pythagorean cases computed in $\mathbb{Z}_p$ that give correct results when used to process more general non-Pythagorean cases in $\mathbb{R}$. For this reason, we start the experiments with very restricted, thus simple, template and test sets, and increase the generality of test data successively.

All experiments presented here led to the same template procedure and the same form of reduced Gröbner basis G'_k, varying only in the coefficients of $g'_j \in G'_k$,

$$g'_1 = h'^6_9 h'_{10} + c'_1 h'_5 h'_6 \qquad g'_8 = h'^2_6 + c'_8 h'^2_9 \tag{21}$$
$$g'_2 = h'^7_{10} + c'_2 h'_5 h'_6 h'_{11} \qquad g'_9 = h'_1 + c'_9 h'_9 \tag{22}$$
$$g'_3 = h'^6_9 h'_{11} + c'_3 h'^6_{10} \qquad g'_{10} = h'_2 + c'_{10} h'_9 \tag{23}$$
$$g'_4 = h'_5 h'^4_9 h'_{10} + c'_4 h'_6 \qquad g'_{11} = h'_3 + c'_{11} h'_9 \tag{24}$$
$$g'_5 = h'_6 h'^4_9 h'_{10} + c'_5 h'_5 \qquad g'_{12} = h'_4 + c'_{12} h'_5 \tag{25}$$
$$g'_6 = h'_5 h'_6 h'^2_9 h'_{10} + c'_6 \qquad g'_{13} = h'_7 + c'_{13} h'_9 \tag{26}$$
$$g'_7 = h'^2_5 + c'_7 h'_1 h'_9 + c'_{15} h'^2_9 \qquad g'_{14} = h'_8 + c'_{14} h'_9. \tag{27}$$

Consequently, the reduced Gröbner bases G_k that result from processing the templates differ from G'_k also only in the coefficients of the contained polynomials.

Generating the template took about 1:30min for each F'_k and processing time of F_k was $3min$ on average. Certainly, this can still be accelerated. Anyway, the fact that processing the test cases takes actually longer than generating the templates does not contradict our previously claimed objective of saving computation time by applying the proposed strategy. Ideally, the template generation would be required only once in order to process all cases of a particular class of such problems which cannot be solved otherwise. In addition, the ratio of computation times is likely to be reciprocal for the more general problem of equal but unknown distances.

Results of the experiments are displayed in the following tables. The symbol ✓ in the third row (F'_3) and second column (F_2) indicates, *e.g.*, that during the respective experiment, $g_j = 0$ for all $g_j \in G_{2,3}$ resulting from processing the test set F_2 with the template produced by F'_3. The symbol ✗ is used when at least one $g_j \in G_{k,k^*}$ exists, such that $g_j \neq 0$. Thus, the template generated by F'_{k^*} is an adequate template if all entries in the respective row contain a check mark.

Experiment 1

The first experiment is performed to see if the presented method can be applied at least to different sets of polynomials that all originate from data that forms Pythagorean quadruples, such that equation (20) holds. The test data in this case is the same as the template data, i.e. $F_k = F'_k$ for all $k = 1, \dots, 10$. Furthermore, we restrict the the diagonal elements of $\mathbf{H}'_k$ to be 1, $h'_1 = h'_4 = h'_6 = h'_9 = h'_{10} = h'_{11} = 1$, in order to ensure that all elements of $\mathbf{H}'^{-1}_k$, and thus all $\hat{X}'_i, \hat{Y}'_i$ as well, are integers.

The diagonal entries of table 1 show expectedly that the coefficients computed in $\mathbb{R}$ within the reduced Gröbner basis for each test set F_k using the template of the respective template set F'_k are correct (because here $F_k = F'_k$ for all k). Moreover, we see that the sets F'_6, F'_7 and F'_{10} provide an adequate template, in the above explained sense, for all test sets $F_1, \dots, F_{10}$. It would be interesting to investigate the reasons why only some sets F'_k constitute adequate template sets.

	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
F'_1	✓	✗	✓	✓	✓	✗	✗	✓	✗	✗
F'_2	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗
F'_3	✓	✗	✓	✓	✓	✗	✗	✓	✗	✗
F'_4	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
F'_5	✓	✗	✓	✓	✓	✗	✗	✓	✗	✗
F'_6	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
F'_7	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
F'_8	✓	✗	✓	✓	✓	✗	✗	✓	✗	✗
F'_9	✗	✓	✗	✗	✗	✗	✗	✗	✓	✗
F'_{10}	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 1. Results of experiment 1

Experiment 2

Now, we want to find out whether the template sets from the previous experiment can be used to process data that does not originate from Pythagorean triples and quadruples. Therefore, we reuse the template sets F'_k from the previous experiment, but generate new test sets F_k, such that $F'_k \neq F'_k$ for all $k = 1, \dots, 10$. This test data is as well generated from random integers, hence $x_{l,i}, y_{l,i}, h_j \in \mathbb{Z}$ for all $l = 1, 2, 3$, $i = 1, \dots, 15$, and $j = 1, \dots, 11$. But test points do not form Pythagorean quadruples, therefore equation (20) does not hold. As before, diagonal elements of $\mathbf{H}'$ and $\mathbf{H}$ are set to 1 to guarantee that the elements of the corresponding inverse matrices are also only integers.

This experiment shows that the templates which proved be adequate in the first experiment are also applicable to correctly process non-Pythagorean integer test cases. As expected, templates that failed in the experiment before, are not successfully applicable here either and not all of them were tested.

	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
F'_1	✓	✗	✗	✓	✗	✓	✓	✓	✓	✓
F'_2	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
F'_3	✓	✗	✗	✓	✗	✓	✓	✓	✓	✓
F'_4	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
F'_6	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
F'_7	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
F'_{10}	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 2. Results of experiment 2

Experiment 3

Next, we release the restrictions on the matrices $\mathbf{H}_k$ for creating test sets. New test sets F_k are generated similarly as in the previous experiment using random $x_{l,i}, y_{l,i}, h_j \in \mathbb{Z}$ that do not form Pythagorean quadruples for all $l = 1, 2, 3$, $i = 1, \dots, 15$ and $j = 1, \dots, 11$. But now, instead of setting the diagonal elements of $\mathbf{H}_k$ to one, they have to be a power of 2, $h_1, h_4, h_6, h_9 \in \{\pm n^2 | n \in \mathbb{N}\}$, in order to avoid repeating decimals in $\mathbf{H}_k^{-1}$ which possibly introduce additional errors when truncated at the beginning of the procedure. The same template sets F'_k as in the two experiments before are used here.

In this experiment, none of the templates from the sets F'_k used so far represented an adequate template to process the present test sets F_k. There were at least 2 $g_j \neq 0$ in each G_{k,k^*}, for all $k = 1, \dots, 10$ and $k^* = 1, \dots, 10$. Consequently, templates that were created with the restriction that the diagonal elements of $\mathbf{H}'$ are one cannot be successfully applied to test sets F_k for which this constraint does not hold.

Experiment 4

As a consequence of the results in experiment 3, we create new template sets with less rigorous restrictions on $\mathbf{H}'_k$. In particular, the diagonal elements of $\mathbf{H}'_k$ must be a power of 2, $h'_1, h'_4, h'_6, h'_9 \in \{\pm n^2 | n \in \mathbb{N}\}$. Apart from that, the template sets F'_k are generated as before with $x'_{l,i}, y'_{l,i}, h'_j \in \mathbb{Z}$ for all $l = 1, 2, 3$, $i = 1, \ldots, 15$ and $j = 1, \ldots, 11$, forming Pythagorean quadruples, such that equation (20) holds.

Analogously to the first experiment, we use the template sets as test sets here, such that $F_k = F'_k$ for all $k = 1, \ldots, 10$, to verify that the method can be applied to different Pythagorean cases. As expected, the diagonal entries of table 1 again show that the coefficients computed in $\mathbb{R}$ within the reduced Gröbner basis for each test set F_k using the template of the respective template set F'_k are correct. Similar to experiment 1, the results in table 3 demonstrate that some F'_k constitute adequate templates while others do not.

	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
F'_1	✓	✗	✗	✓	✗	✗	✗	✗	✗	✗
F'_2	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗
F'_3	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗
F'_4	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
F'_5	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗
F'_6	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
F'_7	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗
F'_8	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗
F'_9	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
F'_{10}	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 3. Results of experiment 4

Experiment 5

Similarly to experiment 2, we want to investigate now if the template sets from the previous experiment are applicable to process non-Pythagorean cases. Hence, the template sets F'_k from experiment 4 are reused. The test sets F_k are generated in the same way as in experiment 3, i.e. test data is generated from integers that do not form Pythagorean quadruples and the diagonal elements of $\mathbf{H}$ are powers of 2.

The results shown in table 4 confirm that the template sets F'_k which were successfully applied to process test data in the previous experiment, represent adequate templates in this experiment, too. The other templates are expected to be not adequate as they were in the experiment before. Hence, they were not tested, except for the one from F'_2 which supports this assumption.

	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
F'_2	✗	✗	✗	✗	✗	✗	✓	✓	✗	✗
F'_6	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
F'_9	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
F'_{10}	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 4. Results of experiment 5

Experiment 6

So far, the values of $x_{l,i}$, $y_{l,i}$, h_j to generate the test sets were integers only. Finally, we check if the proposed strategy also works for rational test data. Again, we reuse the template sets F'_k from experiments 3 and 4. As mentioned, the test sets F_k are generated from random rational numbers, i.e. $x_{l,i}, y_{l,i}, h_j \in \mathbb{Q}$ for all $l = 1, 2, 3$, $i = 1, \ldots, 15, j = 1, \ldots, 11$. No further restrictions apply on $x_{l,i}, y_{l,i}$ or h_j. In particular, diagonal elements of $\mathbf{H}$ may be any rational number in $[-10, 10]$ except zero.

Table 4 displays the results for this experiment. According to that, the template sets F'_6, F'_9 and F'_{10} are adequate to process the considered rational non-Pythagorean cases. Solving the system of equations (21) to (27) in the resulting reduced Gröbner basis G_k reveals that there are in fact 4 real solutions, including the $h_1, \ldots, h_{11}$ that were use to generate the test data, determined only up to a non-negative h_9. The 4 solutions correspond to 4 possible orientations of the resulting sets of points $\{X_i, Y_i | i = 1, \ldots, 15\}$ for which Y_1 is on the x-axis and Y_2 in the xy-plane.

$$h_1 = -c_9 h_9 \qquad h_2 = -c_{10} h_9 \tag{28}$$

$$h_3 = -c_{11} h_9 \qquad h_4 = -c_{12} h_5 \tag{29}$$

$$h_5 = \pm\sqrt{c_7 c_9 - c_{15}} h_9 \qquad h_6 = \pm\sqrt{-c_8} h_9 \tag{30}$$

$$h_7 = -c_{13} h_9 \qquad h_8 = -c_{14} h_9 \tag{31}$$

$$h_{10} = -c_6 h_5^{-1} h_6^{-1} h_9^{-2} \qquad h_{11} = -c_3 c_6^6 h_5^{-6} h^{-6} h_9^{-18} \tag{32}$$

	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
F'_2	✗	✗	✗	✗	✗	✗	✓	✓	✗	✗
F'_6	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
F'_9	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
F'_{10}	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 5. Results of experiment 6

6. Conclusion

In this paper, we proposed an algebraic way to compute a transformation to upgrade a preliminary projective reconstruction to an Euclidean one by means of constraints derived from segments of equal lengths. We think, this could be useful in environments were distinct features are rare and hence 3D reconstruction by standard methods is error-prone.

We have shown that though the general problem is impossible to solve with standard methods, it is possible to solve it for some special template cases and use the obtained template to solve more general cases. We assume that this can be generalized to real world cases. Our experiments with simplified constraints indicate that this is possible. However, the method has to be tested more thoroughly using the original constraints to provide evidence for this assumption and to investigate under what circumstances it may fail.

Acknowledgements

This work was supported by the project MSM6840770012 - Transdisciplinary Biomedical Engineering Research II.

Appendix: Notation

We use the notations *term* and *monomial* as they are explained in [9], i.e. given a polynomial ring $K[x_1, x_2, \dots, x_n]$, a monomial is a product of the form $x_1^{\alpha_1} \cdot x_2^{\alpha_2} \cdots x_n^{\alpha_n} = x^{\alpha}$, with non-negative integer exponents $\alpha_1, \alpha_2, \dots \alpha_n$. A term then denotes the product $a_\alpha x^\alpha$ of a monomial and a non-zero coefficient $a_\alpha \in K$.

An *S-polynomial* of a pair of polynomials (f, g) is computed as $S(f, g) = x^\gamma(\mathrm{LT}(f))^{-1}f - x^\gamma(\mathrm{LT}(g)^{-1}g$ where $x^\gamma = \mathrm{LCM}(\mathrm{LM}(f), \mathrm{LM}(g))$ is the least common multiple of $\mathrm{LM}(f)$ and $\mathrm{LM}(g)$. $\mathrm{LM}(f)$ denotes the leading monomial and $\mathrm{LT}(f)$ the leading term of f w.r.t. a monomial ordering.

References

[1] CoCoA: a system for doing computations in commutative algebra. `http://cocoa.dima.unige.it`. 4

[2] GMP: The GNU multiple precision arithmetic library. `http://gmplib.org`. 5

[3] Macaulay2: a software system for research in algebraic geometry. `http://www.math.uiuc.edu/Macaulay2/`. 3

[4] Singular: a computer algebra system for polynomial computations. `http://www.singular.uni-kl.de`. 3

[5] T. Becker and V. Weispfenning. *Gröbner bases: a computational approach to commutative algebra*. Graduate Texts in Mathematics. Springer, 1993. 2

[6] M. Brickenstein. Slimgb: Gröbner bases with slim polynomials. Reports on Computer Algebra 35, Centre for Computer Algebra, University of Kaiserslautern, 2005. 4

[7] M. Bujnak, Z. Kukelova, and T. Pajdla. A general solution to the P4P problem for camera with unknown focal length. In *CVPR 2008*, Anchorage, Alaska, USA, June 2008. 1

[8] M. Byröd, K. Josephson, and K. Åström. Fast optimal three view triangulation. In *ACCV 2007*, Tokyo, Japan, November 2007. 1

[9] D. Cox, J. Little, and D. O'Shea. *Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra*. Undergraduate Texts in Mathematics. Springer, 2nd edition, 1997. 2, 3, 8

[10] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2003. 2

[11] K. Kühnle and E. W. Mayr. Exponential space computation of Gröbner bases. In *ISSAC 1996*, Zurich, Switzerland, 1996. 3

[12] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic generator of minimal problem solvers. In *ECCV 2008*, Marseille, France, October 2008. 1

[13] D. Liebowitz and S. Carlsson. Uncalibrated motion capture exploiting articulated structure constraints. *International Journal of Computer Vision*, 51(3):171–187, 2003. 1

[14] B. Lo, A. J. Chung, D. Stoyanov, G. Mylonas, and G.-Z. Yang. Real-time intra-operative 3D tissue deformation recovery. In *ISBI*, May 2008. 1

[15] T. Ortmaier, M. Gröger, D. H. Boehm, V. Falk, and G. Hirzinger. Motion estimation in beating heart surgery. *IEEE Transactions on Biomedical Engineering*, 52(10):1729–1740, 2005. 1

[16] P. Shiu. The shapes and sizes of Pythagorean triangles. *Mathematical Gazette*, 67:33–38, 1983. 3

[17] H. Stewenius, D. Nister, F. Kahl, and F. Schaffalitzky. A minimal solution for relative pose with unknown focal length. In *CVPR 2005*, San Diego, CA, USA, June 2005. 1

[18] D. Stoyanov, A. Darzi, and G. Z. Yang. A practical approach towards accurate dense 3D depth recovery for robotic laparoscopic surgery. *Computer Aided Surgery*, 10(4):199–208, 2005. 1

[19] C. Traverso. Gröbner trace algorithms. In *ISSAC*, 1988. 3

[20] C. Traverso and A. Zanoni. Numerical stability and stabilization of groebner basis computation. In *ISSAC 2002*, Lille, France, 2002. 5

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

Automated identification of tree species from images of the bark, leaves and needles

Stefan Fiel and Robert Sablatnig
Institute of Computer Aided Automation
Vienna University of Technology, Austria
fiel@caa.tuwien.ac.at

Abstract. *In this paper a method for the automated identification of tree species from images of leaves, bark and needles is presented. The automated identification of leaves uses local features to avoid segmentation. For the automated identification of images of the bark this method is compared to a combination of GLCM and wavelet features. For classification a Support Vector machine is used. The needle images are analyzed for features which can be used for classification.*

The proposed method is evaluated on a dataset provided by the "Österreichische Bundesforste AG" ("Austrian federal forests"). The dataset contains 1183 images of the most common Austrian trees. The classification rate of the bark dataset was 69.7%.

1. Introduction

Identification of tree species from images of bark, leaves, and needles is a task which requires expertise. This expert knowledge can be expected from foresters and botanists. For people without this knowledge the identification of tree species is a difficult assignment since the difference between some tree species is small or information for the identification, like the shape of the leaf, or the color and haptics of the bark have been forgotten.

Within a project with the "Österreichische Bundesforste AG" ("Austrian federal forests") people should be able to identify tree species using their mobile devices by photographing leaves, bark, or needles of a tree and the identification is done automatically by the mobile device and additional information for this tree species is then displayed on the screen.

An approach for an automated classification of the tree species from images of the leaves has been presented in Fiel and Sablatnig [4] which avoids the binarization of the leaves. In contrast, this work proposes a method for the automated classification from bark images using local descriptors based on the method for the identification of leaf images. The advantage of the proposed approach is that the same methodology can be used for leaf and bark images. Thus, a preprocessing step can be introduced to distinguish between leaf and bark images without calculating new features. Furthermore the proposed method is compared to a combination of Gray Level Co-occurence Matrices (GLCM) and wavelet features.

The needle images are analyzed for features which can be used for classification and a method is described to distinguish between fir and spruce needles.

This paper is organized as follows: Section 2 reviews the state of the art for automatic identification of plant species from images of the bark. In Section 3 the methodology for the identification of bark images is presented and features which can be used for the identification of needle images are searched. The results are presented in Section 4. Finally a conclusion is given in Section 5.

2. Related work

This section describes the current methods for the automated identification of the tree species. First an overview of the identification using books is given, then the automated classification of images bark is given. To the best knowledge of the author no work has been published about the identification of tree species using images of needles.

The traditional identification of tree species is done manually by using a book like Godet [5]. For the classification of leaves and needles, these books contain a diagnostic key where the user has to make various decisions which describes the leaf or the needle better in each step. The users have to follow a tree step by step to identify the leaf. Since the bark can not be described as easily as leaves or needles, the user has to scroll through the book and has to look for the corresponding bark. The process of the identification can take several minutes since it includes scrolling through the book because most of the decisions lead to another page. Also the users have to be familiar with the vocabulary or have to compare the leaf with the illustrations in the book.

The automated identification of plant species from photos of the bark is done with a texture analyzing methods. Wan et al. [12] made a comparative study based on statistical features. The gray level run length method, GLCM, histogram method, and the auto-correlation methods are compared. For each GLCM the entropy, angular second moment, contrast, inverse different moment, cluster tendency, cluster shade, correlation, maximum probability, and two correlation information measures are calculated. The best results are achieved with the GLCM with an average recognition rate of 77%, followed by the auto-correlation method with 72% and the run-length method with 69%. The histogram method has the lowest results with 65%. To improve the classification rate each of the three color channels are handled separately. This improves the recognition rate for the GLCM method to 89%, for the run-length method to 84% and for the histogram method to 80%. The dataset used contained 160 preselected images of 9 classes.

Song et al. [11] proposed to use a combination of gray scale and binary texture features. As gray scale texture features the GLCM and as binary texture features the long connection length emphasis are used. The classification rate with a nearest neighbor classifier is 87.5% on a dataset containing 180 images of 8 classes.

Huang et al. [8] uses fractal dimension features additional to the GLCM features. The fractal dimension describes the complexity and self-similarity of texture at different scales. For the classification a three layer artificial neural network is used and a recognition rate of 91.67% is achieved. The dataset consisted of 360 preselected images of 24 classes.

Huang [7] combined color and textural information for bark image recognition. Both information were extracted using the multiresolution wavelets. For the textural features the energy of the wavelet transformed images have been used. The color features were gained by transforming the color from RGB values to the YCbCr color space and calculating the energy at depth 3 of the wavelet pyramid for each channel. A radial basis probabilistic neural network is used for classification and an average recognition rate of 84.68% is achieved. The dataset consisted of 300 preselected bark images.

Chi et al. [2] proposed to use Gabor filter banks for the recognition due to its efficiency and accuracy. They introduced multiple narrowband signals model to overcome problems with textures with a lot of maximas. The recognition performance for this approach is 96%. The dataset containted 8 classes of plants and each class containing 25 samples.

3. Methodology

In this section the methodology for the automated identification of tree species from images of the bark is presented, followed by an evaluation of needles images for classification. The automated identification of tree species from images of the leaves is described in Fiel and Sablatnig [4].

3.1. Identification of bark

Classification of the bark is done by using texture analysis methods. In Chen et al. [1] texture is defined as repetitive patterns that occur in a region. The bark of trees does not have exact periodical and identical patterns due to natural growth. Natural cover of the bark, like moss and lichens, distort these patterns or the repetitive occurrence. Due to different lighting conditions the gray values of the patterns are changing and influence the recognition of the patterns. The color of the bark can not be taken into account since with changing lighting conditions and cameras the variance is high.

One of the defining qualities of texture is the spatial distribution of gray values. This distribution can be described using statistical texture analysis methods like the GLCM which was introduced by Haralick et al. [6]. It describes the spatial distribution of the gray values in an image in given orientation and distance. The features used for the classification are contrast, correlation, homogeneity and energy.

Other techniques rely on signal processing like wavelets which was introduced to multi-resolution signal decomposition by Mallat [10]. It allows the decomposition of a signal using a series of elemental functions which are created by scalings and translations of a base function. Thus, wavelets provide spa-

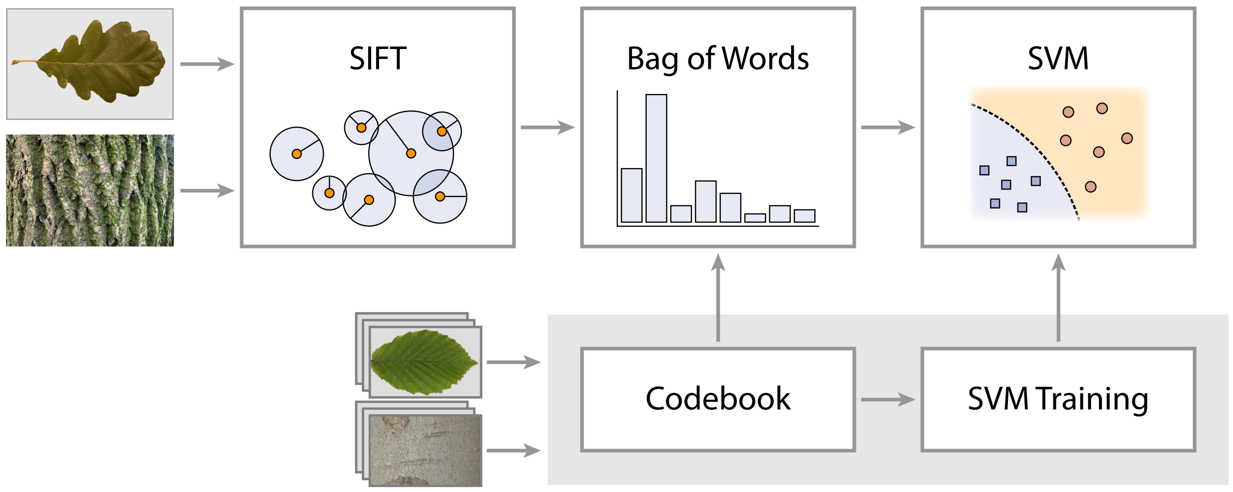


Figure 1: Workflow of the proposed methodology: The input image is normalized to a gray scale image on which the SIFT features are calculated. A histogram of occurrences is generated by searching the nearest cluster center in the codebook which is used for the classification. The gray area represents the machine learning part where codebooks are generated from the trainingsset. For each image in the trainingsset a histogram of occurrences is calculated which are then used to train the SVMs.

tial and frequency information at different scales. As feature for the classification the average energy of the wavelets coefficients are used.

Zhang et al. [13] showed that SIFT features, introduced by Lowe in [9], can keep up with common texture classification methods. The SIFT features are used to describe the texture of the region. Since this method is used for the automated identification of the leaf images it has also been tested on bark images. The advantage of this method is that it does not rely on periodical patterns but on patterns which occur frequently in the image. With the bag of words approach, which was introduced by Csurka et al. [3], these patterns do not have to be identical since the nearest cluster center is searched which represents similar regions. So the method from Fiel and Sablatnig [4] which is used for the identification of leave images is also applied for the automated identification of tree species from images of the bark.

Figure 1 illustrates the workflow which is used for the classification. The method consists of three steps. First the images are transformed into a normalized gray scale image. There the SIFT features are calculated by searching keypoints in the images using DoG at different scales. The neighborhood of these keypoints are then described using orientation histograms which are then used for a bag of worlds model. Features of the trainings set were clustered to form a codebook. New images can then be described by generating a histogram of occurrences of the nearest cluster center. This histogram can then be classified using a one-vs-all Support Vector Machine (SVM).

A SVM is used since it rather minimizes the overall risk than the overall error of a training set, which results in a good generalization performance even for high-dimensional features. To handle multiple classes the one-vs-all approach is used. It generates a SVM for each class which classifies the data points of the class against all other data points. The classification is done by a winner-takes-all strategy, meaning that the classifier with the highest output function assigns the class. For each class the value of the output function can be used as percentage of belonging to this class. Thus, a threshold can be introduced to eliminate images which have only a small percentage of belonging to a class.

3.2. Identification of needles

The dataset of needle images contains the 6 most common Austrian conifer trees which can be divided into two classes. The first class are trees on which one needle grows separately on the branch and the second class are the trees on which the needles grow on clusters at the branch.

Fir and Spruce are the two trees on which the needles grow separately on the branch. The easiest way to distinguish their needles is that the spruce needle has two white stripes on the backside. Since it can not be assumed that every image shows the backside of the needle this characteristic can not be used. The next differences between the needles is that spruce needles are blunt and they grow in one plane on the branch and fir needles are pointed and they can grow in every direction. Due to overlapping needles of the spruce the grow direction can not be determined. The endings of the needles are found by segmenting the image (see Figure 2 a)) followed by calculating the skeleton of the needles (see Figure 2 b)). The endpoint of the skeleton are the endpoints of the needles (see Figure 2 c)). The endings of the needles are now analyzed by calculating features like the eccentricity, solidity, curvature features, and the moment invariants.

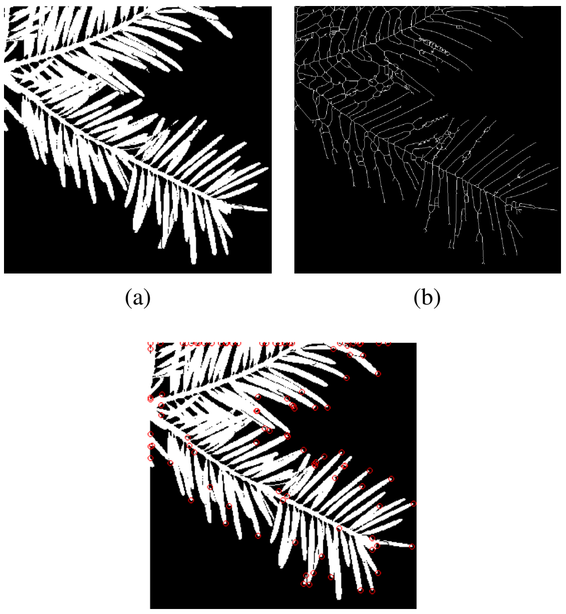

(a) (b) (c)

Figure 2: Finding the endings of the needles of an image of a spruce. The segmentation of the branch and the needles from the background is shown in (a). In the middle is the skeleton of the first image and with this skeleton the endpoints of the needles can be found (c).

The trees on which the needles grow in cluster are distinguished by the number of needles in the cluster. The endings of the needles can be found with the method described above. Since the needles in the cluster are overlapping and the clusters are lying close to each other the number of needles in the cluster can not be determined.

4. Experiments and results

In this section the experiments and results are presented. The experiments were done on datasets which were provided by the “Österreichische Bundesforste AG”. In Section 4.1 the experiments on the leaf dataset are presented. Afterwards, in Section 4.2, the experiments on the bark dataset are evaluated.

Since no method has been found to identify the tree species from images of the needles no experiments have been carried out for the trees on which the needles grow in clusters. Fir and spruce can be identified by analyzing the endings of the needles. 5 of 5 images of the fir and 7 of 9 images of the spruce were identified correctly. The spruce needles are misclassified since the needles are rotated on the branch and so the blunt ending of the spruce needles become pointed in the image.

4.1. Experiments on the leaf dataset

The leaf dataset consists of 134 images of the five most common Austrian broad leaf trees. This images are scaled to either 800 pixel height or 600 pixel width. Each class has between 25 and 34 images. Experiments have shown that 30 cluster centers for each class lead to the best results on our dataset.

The description of the results has already been shown in Fiel and Sablatnig [4]. For reasons of completeness the results are shown again in Table 1.

	Ash	Beech	Hornbeam	Mountain oak	Sycamore maple
Ash	**14**	1		1	1
Beech		**20**		2	
Hornbeam	1		**25**		
Mountain oak				**14**	
Sycamore maple					**15**

Table 1: Confusion matrix of the first experiment on the leaf dataset. The tree names on the top are the estimated classes, the names on the left side the true classes.

4.2. Experiments on the bark dataset

The bark dataset consists of 1183 images of the eleven most common Austrian trees. The images, which are showing a section of the bark of the size of approximately an A4 paper, are scaled to either 800 pixel height or 600 pixel width. Each class contains between 16 and 213 images.

The first experiment is done on the whole bark dataset. The amount of the centers has been set to 30 per class and the size of the trainings set is set to 30, which was evaluated empirically. Classes which have less then 30 images are also trained but no images are left for testing. These classes are skipped in the rows of the table. The results are shown in Table 2. The classification rate is 69.7%. The highest recognition rate has the Spruce with 82% (101 out of 123 images), followed by the fir with 76% (51 out of 67). 55 images (which are 72%) of the black pine images are assigned to the correct class. The larch and the swiss stone pine have a classification rate of 70 respectively 67% (77 out of 110 respectively 12 out of 18 images). 62% of the mountain oak image are assigned correctly which are 29 out of 47. The scots pine has a recognition rate of 53% (53 out of 100). The ash has the poorest result with 33% but since there are only three images remaining in the test set this result is not representative.

The same dataset was tested with a combination of GLCM features (contrast, correlation, energy, and homogeneity) and the average energy of the wavelets coefficients. The GLCM features are calculated for 0, 45, 90, and 135 degrees with a distance of 1 and 5 pixels and the depth of the wavelet packet was 5. The results of this experiment are presented in Table 3. The classification rate is 61.2%. All three remaining images of the ash dataset are identified correctly. The fir and the spruce are the classes with the second best recognition rate of 67% respectively 65%. 53% of the black pine are assigned to the correct class, whereas the method has the worst performance on the mountain oak and the swiss stone pine with 43 respectively 39%.

The next experiment was done on a subset of the bark images containing 9 images of each class. The trainings sets are maximal 30 images, for those classes which have less then 39 images the rest of the dataset was used as trainings set. The number of centers per class for the bag of word method remained at 30.

This subset was presented with an online survey to two employees of the "Österreichische Bundesforste AG". The first is a biologist, who studied at the University of Natural Resources and Life Sciences in Vienna and is now working in the natural resource management department and the second is a forest ranger with practical experience of more than 15 years. The classification rate of the first experts was 56.6% and the classification reate of the second expert was 77.8%. Both experts said at the end of the experiment that they had the biggest problem by distinguishing the three pine species and the larch. Sample images which are showing that the difference between the classes is often lower then the intraclass variance can be seen in Figure 3. Both experts noted that they use other chracteristics for the identification, like the location where the tree grows, the habit of the bark, or the buds on the branches.

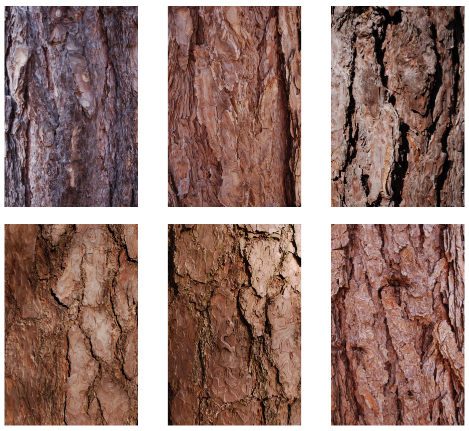


Figure 3: Sample images to show the intraclass difference. The first row shows 3 images of black pines, the second row shows two images of a scots pine and one image of a larch.

The proposed method is also applied on the same subset of images. The results of this experiment can be seen in Table 4. The classification rate is 65.5%, which is approximately between the rate of the two experts. The ash, beech, black pine, fir, and spruce have a recognition rate of 88.8%. The hornbeam and the swiss stone pine have a recognition rate of 77.7%. 6 of the 9 images of the scots pine are classified correctly and 5 of the mountain oak images are assigned correctly. None of the larch or sycamore maple are identified. The reason why none of the sycamore maple is classified correctly is that in these images shadows occur and the trees are covered with moss and lichens. 6 of the larch images are assigned to the black pine class, also three images of the scots pine,

	Ash	Beech	Black pine	Fir	Hornbeam	Larch	Mountain oak	Scots pine	Spruce	Swiss stone pine	Sycamore maple
Ash	**1**							1	1		
Black pine	1		**55**			10		10			
Fir			1	**51**		2	3	1	9		
Larch			13	1		**77**	1	11	2	3	2
Mountain oak	2		2		1	3	**29**			6	4
Scots pine	1		10	4	2	19		**53**	4	6	1
Spruce	2	1		4	2		7	1	**101**	1	4
Swiss stone pine					1	2	1	2		**12**	

Table 2: Confusion matrix of the experiment on the bark dataset with a trainings set size of 30 images. The tree names on the top are the estimated classes, the names on the left side the true classes. Classes where no images are left for testing are skipped in the first column.

	Ash	Beech	Black pine	Fir	Hornbeam	Larch	Mountain oak	Scots pine	Spruce	Swiss stone pine	Sycamore maple
Ash	**3**										
Black pine			**40**			16		15	1	3	1
Fir				**45**		3		9	7	2	1
Larch			12	1		**71**		21		5	
Mountain oak	4		1			2	**20**	8	4	7	1
Scots pine			11	2		17		**64**	2	3	1
Spruce	4	3		6	12	4	2	3	**83**	2	4
Swiss stone pine			1	2		1		5	1	**7**	1

Table 3: Confusion matrix of the experiment with combined features of the GLCM and wavelets. The trainings set contained maximal 30 images per class. Classes where no images are left for testing are skipped in the first column.

which confirms that the three pines and the larch are hard to identify. This was already shown in Figure 3.

5. Conclusion

This paper presented a method for an automated identification of tree species from images of the bark based on the method for the identification of leaves. The method described uses local features to describe the texture since local features can keep up with texture classification methods [13]. No method has been found for the classification of the needle images. A method has been presented to distinguish between fir and spruce needles but the images has to be in good quality because segmentation is needed and the endings are analyzed.

The proposed method consisted of three steps. First the images were transformed into a normalized gray scale image. There the SIFT features were calculated and the neighborhood of these keypoints are then described using orientation histograms. Features of the trainings set are clustered. For each feature in an image the nearest cluster center is searched and the histogram of the occurrences can then be used to train a one-vs-all SVM. When classifying a

	Ash	Beech	Black pine	Fir	Hornbeam	Larch	Mountain oak	Scots pine	Spruce	Swiss stone pine	Sycamore maple
Ash	**8**		1								
Beech		**8**							1		
Black pine			**8**							1	
Fir				**8**					1		
Hornbeam					**7**		2				
Larch			6			**-**		2	1		
Mountain oak	3						**5**	1			
Scots pine			3					**6**			
Spruce								1	**8**		
Swiss stone pine			1					1		**7**	
Sycamore maple	1		4	1					1	2	**-**

Table 4: Confusion matrix of the experiment on the bark dataset used for the experiments with the experts. The tree names on the top are the estimated classes, the names on the left side the true classes.

new image the SIFT features are calculated and the histogram of the nearest cluster centers is used as input for the SVM.

Experiments and results have been presented datasets of leaf and bark images. The classification rate for the leaf dataset was 93.6%. When applying the proposed method on the bark dataset the classification rate was 69.7%. A subset of the bark images were generated and experiments with two experts were made. The classification rates of the two experts were 56.6 respectively 77.8%. When applying the proposed method to this subset the classification rate was 65.6% which is approximately in between.

The disadvantage of the proposed method is that the calculation of the SIFT features is computational intensive and due to the clustering for the bag of word model online learning is not possible. The advantage is that the proposed method can be applied to leaf and bark images. Thus, a preprocessing step can be introduced to distinguish between bark and leaf images without calculating new features.

Acknowledgements

We would like to thank the “Österreichische Bundesforste AG” (“Austrian federal forests”) for their support and for providing the images.

References

[1] C. H. Chen, L. F. Pau, and P. S. P. Wang, editors. *Handbook of Pattern Recognition and Computer Vision*. World Scientific Publishing Co., Inc., 2000. 2

[2] Z. Chi, L. Houqiang, and W. Chao. Plant species recognition based on bark patterns using novel Gabor filter banks. In *Proc. International Conference on Neural Networks and Signal Processing*, volume 2, pages 1035–1038, 2003. 2

[3] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. *In Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004. 3

[4] S. Fiel and R. Sablatnig. Leaf classification using local features. In L. M. Blauensteiner P., Stöttinger J., editor, *Proceedings of 34th annual Workshop of the Austrian Association for Pattern Recognition (AAPR)*, pages 69–74, Zwettl, Austria, 2010. 1, 2, 3, 4

[5] J.-D. Godet. *Bäume und Sträucher bestimmen und nachschlagen*. Eugen Ulmer KG, 2007. 1

[6] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural Features for Image Classification. *IEEE Transactions on Systems, Man and Cybernetics*, 3(6):610 –621, nov. 1973. 2

[7] Z.-K. Huang. Bark Classification Using RBPNN Based on Both Color and Texture Feature. In *IJCSNS - International Journal of Computer Science and Network Security, Vol. 6 No. 10 pp. 100-103*, 2006. 2

[8] Z.-K. Huang, C.-H. Zheng, J.-X. Du, and Y. Wan. Bark Classification Based on Textural Features Using Artificial Neural Networks. *Third International Symposium on Neural Networks, 2006, Proceedings, Part II*, pages 355–360, 2006. 2

[9] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 3

[10] S. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674 –693, jul. 1989. 2

[11] J. Song, Z. Chi, J. Liu, and H. Fu. Bark classification by combining grayscale and binary texture features. In *Proc. International Symposium on Intelligent Multimedia, Video and Speech Processing*, pages 450–453, 2004. 2

[12] Y.-Y. Wan, J.-X. Du, D.-S. Huang, Z. Chi, Y.-M. Cheung, X.-F. Wang, and G.-J. Zhang. Bark texture feature extraction based on statistical texture analysis. In *Proc. International Symposium on Intelligent Multimedia, Video and Speech Processing*, pages 482–485, 2004. 2

[13] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study. In *Proc. Conference on Computer Vision and Pattern Recognition Workshop*, page 13, June 17–22, 2006. 3, 6

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

Automatic Tree Detection and Diameter Estimation in Terrestrial Laser Scanner Point Clouds

Anita Schilling, Anja Schmidt and Hans-Gerd Maas
Institute of Photogrammetry and Remote Sensing,
Dresden University of Technology, Germany
anita.schilling@tu-dresden.de

Abstract. *We present a method to detect trees in 3D point clouds of forest area acquired by a terrestrial laser scanner. Additionally, a method to determine the diameter at breast height of the detected trees is shown. Our method is able to process large data sets bigger than* 20 GB *in a reasonable amount of time. Results from scans on our test site with different seasonal vegetation are shown. Tree diameters can be reliably determined from the same trees in different scans.*

1. Introduction

Terrestrial laser scanners gained widespread popularity in the last years because point cloud representations of 3D objects can be acquired rapidly and easily. Applications cover a wide range from documentation of cultural heritage sites or accidents to environmental change detection or industrial engineering. We are interested in the development of methods to extract forestry related parameters from scans of forest area. These so-called inventory parameters for a particular forest site, e.g. tree height, diameter at breast height, crown diameter and basis, are important for forest monitoring and management. Usually, a sample set of trees is measured manually by time intensive methods to determine values for a forest. In some cases destructive methods cannot be avoided to obtain reliable results.

Laser scanning is especially attractive for this kind of tasks since it allows fast capturing of scenes in a non-destructive way. The scene analysis can then be performed off-site and already acquired point clouds can always be processed again if other parameters are needed.

Our aim here is the automatic generation of a map of the trees within the laser scanned scene. The actual number of trees within the area is unknown. Each tree has to be characterized by its diameter at breast height defined at $1.3m$ w.r.t. the lowest tree trunk point. Furthermore, the tree position is considered to be the center point of the circle from which the diameter is obtained.

A lot of work has also been done on detecting and segmenting trees in airbourne laser scans as reported in [11]. Our focus lies on terrestrial laser scanning within the scope of our ongoing project to recover the 3D forest structure, from which we present preliminary results. Similar work on tree detection and diameter estimation was described in [1], but the studied test site was less than half the size of ours. Our study site consists of a birch stock covering an area ($160m \times 80m$) of about $1.3ha$. The site was captured from 12 separate scanning positions in winter and spring 2010. The scenes show substantial seasonal changes in vegetation. Because of the size of the test area, our focus is on developing a robust method that can calculate the tree diameter reliably with different understorey vegetation present.

The paper is organized as follows: Section 2 gives an overview of laser scanner techniques and the data specifications. In section 3, the investigated methods are explained in detail. Following, experimental results using scans from our test site are presented in section 4. Finally, section 5 summarizes our findings.

2. Data Acquisition

A terrestrial laser scanner determines the distance to an object by emitting a laser pulse and measuring the time of flight until the reflection of an object is observed at the device or by a phase comparison of the reflection to the initial value. Usually, a laser in the near-infrared is utilized. The strength of the reflected laser pulse affects the measurement accuracy and is dependent on the incident angle and object material properties.

Most scanners work in their own polar coordinate systems with the scanning mechanism as origin. The vertical and horizontal directions are divided by an angular sampling interval of α_r degrees obtaining a spherical grid around the scanner head. A laser beam is sent through each of the spherical grid points (θ, ϕ). The distance d to the first object hit by the laser beam is measured. Thus, a particular object can only be measured if the line of sight between scanning mechanism and object is unobstructed. For this reason lower trunk parts are occasionally insufficiently represented due to understorey vegetation which is closer to the scanner than the targeted trees. Additionally, the laser exhibits a beam divergence resulting in an increasing beam diameter with distance. Therefore, several objects might be hit by one laser beam resulting in multiple reflection at the scanner. Some scanner models utilizing phase-comparison average the range values from several observed reflections ([1]), which decreases the accuracy of the scene representation.

The point cloud acquired in polar coordinates (θ, ϕ, d) is then converted to Cartesian coordinates (x, y, z). The resulting point cloud is a sampled representation of the object surfaces around the scanner. To represent an object from all sides, several scans have to be acquired providing full object coverage. The separate scans need to be registered to the same coordinate system using natural or artificial markers. Although the basic principle of laser scanners is straightforward and provides 3D coordinates of the objects around the device, accuracy depends on the characteristics of the utilized device as well as the object properties. An in-depth description of terrestrial laser scanning can be found in [11].

We used the terrestrial laser scanner Imager 5006i from Zoller+Fröhlich to capture the test site. The scanner uses a phase comparison technique which can resolve distances up to maximal $79m$ ([12]). Object points which are hit further away are treated as if they would lie within the maximum distance, i.e. $d = d - 79m$, resulting in *ghost points*. As reported in [1], these points have usually a very low reflection strength and can be removed by applying a suitable threshold. Therefore, we set the threshold value for the reflection strength to 0.005. The reflection strength of the measurements is in the interval $[0 \ldots 1]$. Only points with a reflection strength greater than the threshold are used. This reduces the point cloud sizes by about 15% to 29%. Since natural materials, e.g. bark or leaves with low incident angle, also yield low reflection strengths, it cannot be ruled out that a fraction of those are removed as well.

	Session 1	Session 2
time of scan	March	May
binary file size	$24\,GB$	$27.6\,GB$
total no. points	1,738,900,000	2,005,000,000
no. of points used	1,269,056,557	1,471,058,980

Table 1: Laser scanner data specifications. One point consists of 3D coordinates and a value indicating the reflection strength of the particular measurement as floats. The number of points used denotes points with an reflection strength greater than 0.005.

The angular resolution used was $0.0018°$ resulting in $20,000$ range measurements per $360°$. The field of view in the vertical direction is limited to $310°$ due to the scanner tripod. The test site is captured by 12 scans from fixed positions as indicated by figure 1. The 12 separate scans for each scanning session were co-registered by fixed spherical markers mounted on same trees. Registration was performed manually with the Zoller+Fröhlich scanner software. Each separate point cloud was limited to a radius of $37m$ around the scanner and exported as 3D Cartesian coordinates. The data specifications are summarized in table 1. The first session was scanned in March 2010 when there was no foliage on the trees and the understorey vegetation had been freshly pruned. In May 2010, the second session was acquired when the vegetation had grown significantly and trees were covered by foliage again.

3. Methods

The generation of a Digital Terrain Model (DTM) is necessary to determine the lowest trunk point of each tree. The DTM represents the ground as 2D matrix containing height values as elements. For the DTM generation a method presented in [3] is applied. The actual detection of trees within the point clouds is based on the assumption that the highest density of scan points is on the tree trunks. This was also exploited in [5], [4], [8] and [1]. A problem of the tree detection is the possible mutual occlusion of trees and other vegetation in the scans at different heights. Therefore, the detection method needs to consider several different heights. The targeted birch

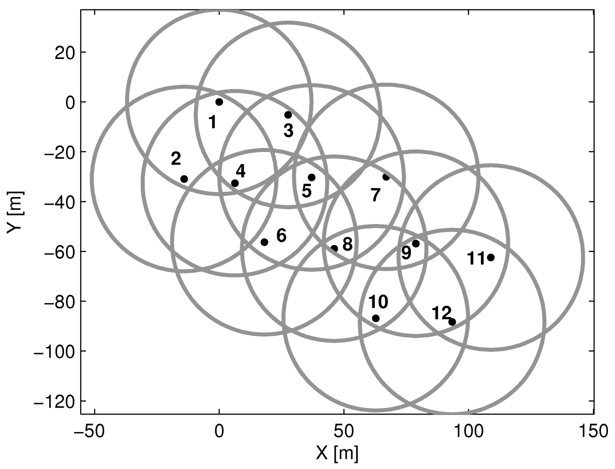


Figure 1: Distribution of the 12 scanner positions per session with radius of $37m$.

trees in the area are $38m$ in height. Smaller trees and some coniferous trees are also present within the area as well as shrubs of different extent. As the number of trees within the test site is unknown, the detection method needs to be robust enough so that no birch trees are missed.

Following tree detection, the points contributing to single trees are analysed separately to find points in breast height. Based on the previously determined DTM plane, the breast height of a particular tree is computed. Then, points at breast height are used to fit a circle to obtain the trunk diameter. Since a tree usually does not grow up perfectly straight, it can hardly be completely located using one 3D point. In spite of this, we use the center point of the fitted circle to indicate the position of a tree. The tree position is used to create an overview map of the test site. For further processing an ample radius around the reported position has to be considered.

The main issue is to determine the boundary of the tree trunk in breast height. This is complicated by the fact that in lower heights many scan points represent other vegetation partially obstructing the trunks. In [1], this task was performed with only few trees on a very small test site, taking about $10h$ processing time. We present a method to achieve reliable results in a reasonable amount of time for a comparatively large data set. The method is summarized in algorithm 1. The position and diameter at breast height values for the trees are eventually summarized as a map of the trees on the test site.

```
1. determine DTM plane $g_{DTM}$ for 3D point set $E$ from all positions of a scan session
2. detect trees and calculate a set of tree position estimates $T$ (see algorithm 2)
3. for each tree position $t \in T$
    (a) load 3D points within bounding volume from $E$,
        $P_t = \{p \in E : t_x - b_x \le p_x \le t_x + b_x \wedge t_y - b_y \le p_y \le t_y + b_y\}$
    (b) if $|P_t|$ is sufficient determine DTM plane $t_{DTM}$ from $P_t$, otherwise use $g_{DTM}$
    (c) calculate circle estimate $c$ in height $h_s$ (see algorithm 3)
    (d) compute lowest trunk point height $k_z^1$ by projecting the circle center onto the DTM plane
    (e) calculate circle update $c$ (see algorithm 3)
    (f) compute new lowest trunk point height $k_z^2$
    (g) if $|k_z^1 - k_z^2| > \epsilon$ then repeat starting at step (c) with $h_s = h_s + h_o$
    (h) log resulting diameter at breast height $t_{dbh} = 2 \cdot c_{radius}$ and tree position
        $t_p = (c_x, c_y, c_z)$ for the tree
```

Algorithm 1: Scheme of subtasks for tree detection and diameter calculation.

3.1. Digital Terrain Model generation

The method to generate the DTM that is summarized here, was originally presented in [3]. The xy-plane is partitioned into a 2D grid with cell size s_c. When projected onto this plane, several 3D points lie in the same cell. For each single cell, the z-axis is divided in several bins each covering a height interval of s_l. Points which are located within the current cell are counted in the bins corresponding to their z coordinate. Thus a height histogram is built for each cell from the point numbers. The histogram bin with the highest number is assumed to be the ground and the bin height is assigned to the current cell.

If a tree trunk was occluded by vegetation closer to the scanner, then there are hardly any points at the real ground height. In this case lower histogram bins are empty because the trunk points are only contributing to higher bins of the particular cell. The

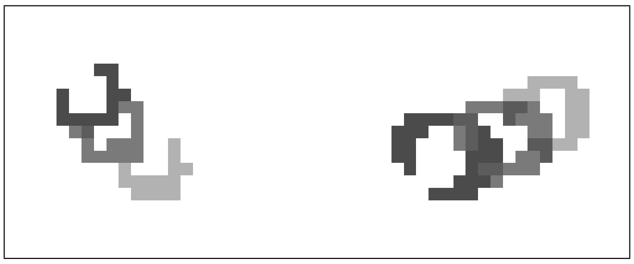


Figure 2: Arc- or circle-like shapes caused by tree trunks in different height layers indicated by gray value.

maximum bin is determined far to high resulting in a false height value. Therefore, the grid cell heights need to be filtered. If a cell height is too high in comparison to its neighbouring cells and a determined threshold then the cell value is removed.

Afterwards, cells with missing height values are interpolated using neighbouring grid cell heights. The 2D index of each cell is converted to (x, y) coordinates in the point cloud coordinate system with the cell height as z coordinate. Finally, an adjusted plane is fitted to this 3D point set.

A DTM is generated for each point cloud of one scan session separately. To obtain a general DTM for the entire scan session, the separate DTMs are merged. The DTMs of separate point clouds are overlapping in several parts of the test area. In these cases uninterpolated height values were preferred and averaged if multiple values were available.

3.2. Tree Detection

Our tree detection method is presented in algorithm 2. As already mentioned, it is based on the assumption that in the forest area the highest density of scan points are located at the tree trunks. To benefit from the nearly full trunk coverage in the overlapping parts of the point clouds, the entire scan session needs to be processed at once. Therefore a height slice of the scan session is considered. Points within that slice are projected to a 2D grid that partitions the xy-plane. For each cell, the number of points within the cell is counted. Grid cells with a point count less than a defined threshold $minNbPoints$ are cleared. If a suitable threshold is applied, the non-zero cells are likely to correspond to positions at the tree trunks. The trunk boundaries appear as components with an arc- or circle-like shape as shown in figure 2, though the cross section of a trunk is rarely a perfect circle. A more detailed analysis of the trunk points is necessary for each tree in any case, therefore determining approximate coordinates of the tree location is sufficient for the tree detection step.

```
1. at different heights $h_i$, slices of thickness $t$, project all points within onto a plane $l_i$ parallel to $xy$-plane
2. partition $l_i$ by a 2D grid $g_i$, count no. of points in each grid cell
3. grid $g_i$ represented by an $m \times n$ matrix $\mathbf{I}^i$ where
   $\mathbf{I}^i(m,n) = \left\{ \begin{array}{ll} 1 & g_i(m,n) > minNbPoints \\ 0 & otherwise \end{array} \right\}$
4. concatenate matrices $\mathbf{I}^i$ with OR operation, thus
   $\mathbf{K}(m,n) = \left\{ \begin{array}{ll} 1 & \mathbf{I}^i(m,n) = 1 \\ 0 & otherwise \end{array} \right\}$
5. dilate $\mathbf{K}$ with square structure element of size $s \times s$
6. find and uniquely label components in $\mathbf{K}$ by connected component labelling
7. find components in $\mathbf{I}^i$ by connected component analysis, join components $c$ by component number from $\mathbf{K}$ thus
   $M[\mathbf{K}(c_m, c_n)] = M[\mathbf{K}(c_m, c_n)] \cup (c_m, c_n)$
8. for each index list in $M$ calculate 2D centroid from indices, convert to point cloud coordinate system, resulting 2D coordinates are tree position estimate
```

Algorithm 2: Detection of trees in point clouds.

It is possible that a tree does not appear on the 2D grid of a particular height because of occlusions. Hence, several different heights have to be analysed. The components in each of the 2D grids are detected by a connected component labelling algorithm ([9]). Because of the skewed tree growth, components corresponding to the same tree in grids of different heights do not necessarily cover the same grid cells. But components of the same tree are inevitably close together and are joined to clusters. Seldomly, components resulting from branches with high scan coverage produce separate clusters, which are at the moment treated as valid detections as well. The 2D centroid of each cluster is computed and constitutes the tree position.

Finally, for each estimated tree position, all points located within a bounding volume are exported to a separate file. The bounding volume is a box of square base with the position estimate at its center. The generation of these smaller point clouds for each presumed tree is the most time intensive part using standard hardware, because of the high number of read and write operations. If sufficient memory, i.e. at least 30 GB, could be provided such that all point clouds of a scan session can be hold within memory, the creation of temporary point clouds for the tree position estimates would be unnecessary.

3.3. Tree Location and Diameter Determination

For tree location and diameter determination, each point cloud section belonging to an estimated tree position is processed separately. First, a DTM is calculated for the point cloud section. If this fails because of an insufficient number of points, an adjusted plane is used instead that is fitted to the 3D points of the respective section of the session DTM.

A first computation of the trunk circle center is necessary to determine the lowest trunk point height accurately. The largest aggregation of 3D points within a circular slice at height h_s around the estimated position is assumed to be the trunk. This subset of points can be found by taking the maximum and neighbouring bins greater than a predefined threshold from histograms of point numbers along the x and y axis as indicated in figure 3. A circle is calculated with Kasa method ([6]) using the 3D point subset. The circle equation is rearranged to

$$-2c_x x - 2c_y y + c_x^2 + c_y^2 - c_r = -(x^2 + y^2) \quad (1)$$

and transformed with the given point set to matrix representation

$$\mathbf{A}_{n\times 3} \cdot \mathbf{k}_{3\times 1} = \mathbf{l}_{n\times 1} \quad (2)$$

with n denoting the number of the considered points. The solution vector $\mathbf{k}$

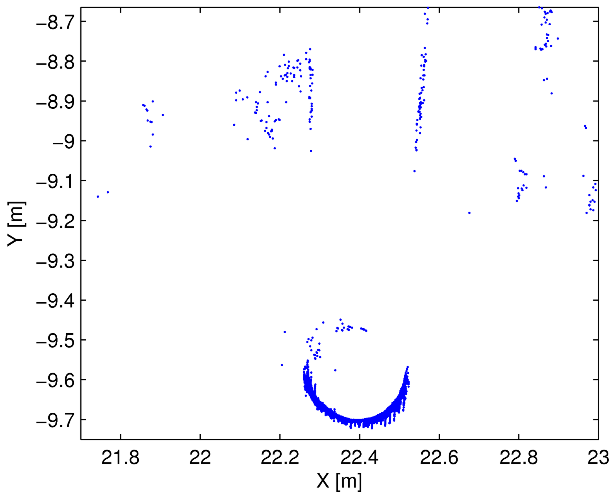
$$\mathbf{k} = \begin{bmatrix} -2c_x & -2c_y & c_x^2 + c_y^2 - c_r^2 \end{bmatrix}^T \quad (3)$$

is obtained by least-squares minimization

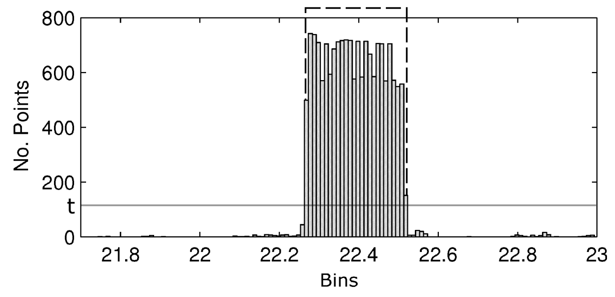
$$\mathbf{k} = (\mathbf{A}^T\mathbf{A})^{-1} \cdot \mathbf{A}^T \cdot \mathbf{l} \quad (4)$$

of the algebraic distances. Following, the elements of $\mathbf{k}$ have to be solved for the circle parameters.

The 2D center point (c_x, c_y) is projected onto the DTM plane to calculate the trunk point height k_z^1. In



(a) Plot of points on xy-plane



(b) Histogram with bin size of $0.01m$ along x axis.

Figure 3: Determination of tree circle estimate using a histogram of point amounts along the x and y axis with predefined threshold t.

a defined height of $1.3m$ w.r.t. the lowest trunk point, a new set of points within a circular slice around the calculated circle center is considered as summarized in algorithm 3. To find a cluster of points in the set resembling a circle the Circular Hough Transform as reported in [10] is utilized. The Circular Hough Transform is based on the fact, that the distance of every point on the perimeter of a circle c_m with known radius r is r. When a circle c_p of the same radius r is drawn around each perimeter point, all circles c_p will necessarily meet at the center point of circle c_m as shown in figure 4.

In [1] the Circular Hough Transform was applied to the non-empty cells of a 2D grid. Previously, the point set was projected onto this grid partitioning the xy-plane and thresholded like explained in section 3.2. Present on the 2D grid are arc- or circle-like shapes from trunks, but also components caused by branches. The accumulation of circles around the component cells on the grid results in only weak support for a particular circle. Instead of the few number of non-empty grid cells, we use each 3D point of the considered set for the circle accumulation. We initialize an empty 2D grid partitioning the xy-plane. The 3D points of the considered slice are projected onto the grid and a circle of size r is drawn around each of the points. For each cell the number of circles passing through it are counted.

In this way many more points are voting for the same circle center resulting in a distinct peak in the grid. Because only an estimate c_r of the precise radius is known, the Circular Hough Transform is applied several times with an increasing radius r. The maximum peak on the grid over all iterations denotes the new circle center (c_x, c_y). The circle radius c_r is updated with the radius r of the corresponding iteration. Finally, a new set of 3D points S is considered containing only points at the previously determined height $h_b \pm \frac{t}{2}$ within a radius defined by c_r with an additional offset d_3. Again, the algebraic circle fit of equation 1 to 4 is used to calculate values

$$\mathbf{c} = (c_x \; c_y \; c_r)^T \tag{5}$$

for the circle parameters. The circle is then fitted ([7]) by a least-squares minimization as in equation 4 with

$$\mathbf{A} = \begin{bmatrix} -\frac{x-c_x}{c_r} & -\frac{y-c_y}{c_r} & -\mathbf{1} \end{bmatrix}_{n\times 3} \tag{6}$$

and

$$\mathbf{l} = \begin{bmatrix} c_r - \sqrt{(x-c_x)^2 + (y-c_y)^2} \end{bmatrix}_{n\times 1} \tag{7}$$

to minimized the geometric distances. The resulting improvements in vector $\mathbf{k}$ are added to the circle parameters. The center point of the adjusted circle is the location of the tree t_c. The diameter $t_{DBH} = 2 \cdot c_r$ is obtained from the circle radius.

The lowest trunk point is determined again by projecting the circle center point onto the DTM. If the resulting trunk point differs from the previously defined height in comparison to a suitable threshold, then a new iteration of the method is performed unless the maximum number of iterations is reached. In this case the first circle estimate is determined anew, starting at a height of $h_s = h_s + h_o$.

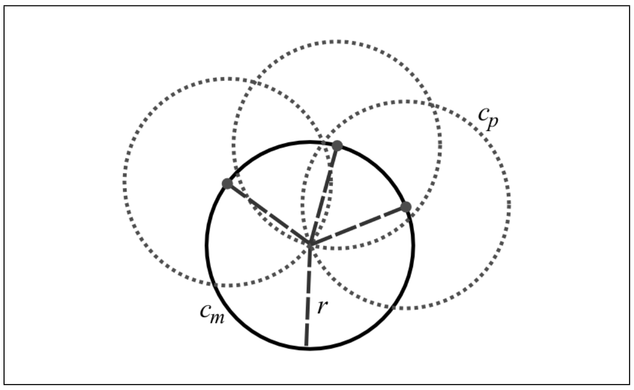


Figure 4: Circular Hough Transform

```
1. create 3D point set
   $S = \{p \in P_t : h_b - \frac{t}{2} \le p_z \le h_b + \frac{t}{2} \wedge d(p, t_c) < c_r + d_2\}$
2. adjust $r_{min}, r_{max}, r_{step}$ according to current circle radius estimate
3. for $r = r_{min},\ r < r_{max},\ r = r + r_{step}$
   (a) project all 3D points of $S$ onto a plane $l$ parallel to $xy$-plane
   (b) draw a circle with radius $r$ around each point $s \in S$
   (c) partition plane $l$ by a 2D grid $g$ with cell size $s_c$, in each cell count no. of circles passing through
4. determine radius $r$ of iteration with maximum cell value in grid $g$
5. convert grid indices to point cloud coordinates $(c_x, c_y)$ for circle $c$ and update circle radius $c_r$ with $r$
6. recreate 3D point set
   $S = \{p \in P_t : h_b - \frac{t}{2} \le p_z \le h_b + \frac{t}{2} \wedge d(p, t_c) < c_r + d_3\}$
7. update circle $c$ with a new circle estimate using $S$
8. calculate adjusted circle fit with $c$ and $S$
```

Algorithm 3: Determination of tree points in breast height and calculation of radius by circle fitting.

4. Experiments

We applied our method to both scan sessions of the test site. The results are summarized in table 2 and processing times are reported in table 3. We are not able to assess the results of the tree detection method regarding its completeness, because the number of birch trees actually present on the test site is not documented. For this reason, the number of false negatives, i.e. trees which were not detected, is also unknown.

We evaluated the results of the detection method manually. 99% of all detected items in session one and 97% in session two are actual trees present on the test site. 91% and 92% of all detections in the respective session are the targeted birch trees, while 8% and 5% are other small or coniferous trees. 1% and respectively 3% of all cases are false positives, which means that structures have been detected which are not trees. These detections were caused by branches with high scan coverage. In the second session more items were detected falsely which is probably due to the foliage present on branches.

We do not have ground truth values for the DBHs of the trees. The evaluation of the DBH only on basis of the computed values is not reliable. Tree diameters are quite variable, which makes the definition of a particular interval difficult. Furthermore, a circle fitted wrongly to a set of points belonging to a shrub nearby the sought-after trunk can also yield a diameter value, which is typical for birch trees.

For this reason, it was verified visually whether the points used for the calculation of the DBH are actually located at the respective tree trunk. For 95% of all detected birch trees in the first and 92% in the second session sufficient correctly located points were selected and therefore an accurate DBH value could be calculated. The averaged standard deviation of the point sets to the fitted circles is $7mm$ and $8mm$. For 5% and respectively 8% of the birch trees the trunk was not sufficiently covered by scan points or the points used for DBH calculation were not localized on the trunk yielding an invalid DBH value.

We are interested in the seasonal change of the vegetation. Before a comparison of the tree appearance in both sessions is possible, the scan sessions have to be registered to each other. The scan sessions exhibit a rotation to each other, but the correspondences and coordinates of the sphere targets are known. The sphere targets were previously used to register separate point clouds of one session to each other. We applied the Iterative Closest Point algorithm ([2]) to obtain a transformation matrix $\mathbf{M}$ using sphere target correspondences. With $\mathbf{M}$ the tree positions of the second scan session could be transformed to the coordinate system of the first session. Then we established correspondences between tree positions from both sessions manually.

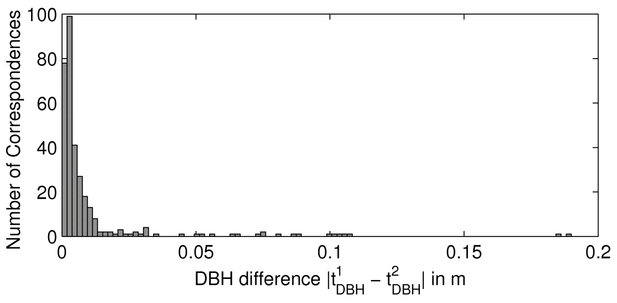


Figure 5: Histogram of DBH differences between the corresponding birch trees.

	session 1	session 2
total detections	363	368
detected birch trees	331	325
false detections	3	11
other detected trees	29	32
valid point set for DBH	316	299
correspondences	323	

Table 2: Results of first and second scan session. All detections were manually checked. The number of false detections is caused by branches which were interpreted as separate vegetation structures.

total processing times	session 1	session 2
DTM generation	$9min$	$9min\ 28s$
tree detection	$4min\ 57s$	$5min\ 46s$
tree separation	$147min$	$157min$
DBH calculation	$20min$	$17min$

Table 3: Processing times for the first and second scan session.

A total of 323 distinct birch tree correspondences were found. For this set the DBH values were compared. The absolute differences of the DBH values of each pair $\epsilon = \left|t^1_{DBH} - t^2_{DBH}\right|$ were calculated and are shown in figure 5 as histogram. 90% of the correspondences exhibit a DBH deviation of less than $2cm$ and even 63% of less than $5mm$. Regarding a maximum DBH difference of $1cm$, the DBH value could reliably determined in both scan sessions for an amount of 268 birch trees present on the test site.

5. Conclusion

We have shown that the generation of a map of trees on a comparably large test site is feasible in a reasonable amount of computation time. Although we cannot entirely evaluate the detections on the test site concerning their completeness, the results look promising. The greatest amount of detections are the targeted birch trees and their diameters at breast height could be determined precisely from the available terrestrial laser scanner point clouds.

There are still a lot of possibilities for improvements. The tree detection method needs to be evaluated whether actually all trees are detected. Further processing would profit from a more detailed analysis by which kind of vegetation structure the detection was caused. False detections from branches or smaller, unwanted trees on the test site could be avoided. Additionally, it is necessary to improve on the diameter calculation. The diameter at breast height has not been accurately calculated for all birch trees though the scan coverage was sufficient.

The DTM generation from scans with dense understorey vegetation is more error-prone, because the actual ground is not sampled enough. We will try to use the DTM from winter scans as basis for all sessions to obtain more reliable height values. The mutual registration of the scan sessions will be necessary for that. The calculation of an appropriate transformation matrix might be improved by utilizing the established tree position correspondences as well.

We have two more scan sessions captured in July and October 2010 exhibiting considerably more seasonal change in comparison to the first session. Therefore, the changing of parameter values is probably not appropriate and a way to adaptively adjust parameter values of the processing step would be beneficial. Furthermore, the representation of the tree location as a single 3D point is in fact not sufficient. Instead, we will aim to capture the topology of a tree directly.

References

[1] T. Aschoff and H. Spiecker. Algorithms for the automatic detection of trees in laser scanner data. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(8/W2), 2004. 1, 2, 3, 5

[2] P. Besl and N. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992. 7

[3] A. Bienert, S. Scheller, E. Keane, G. Mullooly, and F. Mohan. Application of terrestrial laser scanners for the determination of forest inventory parameters. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36, 2006. 2, 3

[4] J. G. Henning and P. J. Radtke. Detailed stem measurements of standing trees from ground-based scanning lidar. *Forest Science*, 52:67–80, 2006. 2

[5] C. Hopkinson, L. Chasmer, C. Young-Pow, and P. Treitz. Assessing forest metrics with a ground-based scanning lidar. *Canadian Journal for Forest Research*, 34:573–583, 2004. 2

[6] I. Kasa. A curve fitting procedure and its error analysis. *IEEE Transactions on Instrumentation and Measurement*, 25:8–14, 1976. 5

[7] T. Luhmann. *Nahbereichsphotogrammetrie: Grundlagen, Methoden und Anwendungen*. Wichman Verlag, Heidelberg, 2000. 6

[8] H. Maas, A. Bienert, S. Scheller, and E. Keane. Automatic forest inventory parameter determination from terrestrial laserscanner data. *International Journal of Remote Sensing*, 29(5):1579–1593, 2008. 2

[9] L. G. Shapiro and G. C. Stockmann. *Computer Vision*. Prentice Hall, 2001. 4

[10] M. Sonka, V. Hlavac, and R. Boyle. *Image processing, analysis, and machine vision*. PWS Publishing, second edition, 1998. 5

[11] G. Vosselman and H. Maas, editors. *Airborne and Terrestrial Laser Scanning*. Whittles Publishing, 2010. 1, 2

[12] Zoller+Fröhlich GmbH. Imager 5006i - An improved system based upon the highly regarded Z+F IMAGER 5006 `http://www.zf-laser.com/BROSCHUERE%20Z+FIMAGER_5006I_E_01.07.09.kompr.pdf`, July 2009. 2

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

Measuring Ball Spin in Monocular Video

Alexander Szép
Institute of Computer Technology, Vienna University of Technology
Gußhausstr. 27-29/384, A-1040 Vienna, Austria
alexander.szep@tuwien.ac.at

Abstract. *This work presents an objective method to measure ball spin in monocular video data. We apply this method to objectively classify racket sports equipment. Therefore, we observe the ball impact on a racket and compare spin differences measured prior to and after the impact. The method combines ball center tracking with surface corner tracking to calculate ball spin. Because our method's application has real-time constraints our spin measurements are fully automatic and without user intervention. Finally, we validate our approach with experimental results and prove that racket classification is feasible based on visual spin measurements.*

1. Introduction

We present a visual method for measuring ball spin aimed for the ball sports domain. Knowledge about ball spin enables a range of applications for sports where spin plays a crucial role like in table tennis, tennis, soccer, baseball, golf, bowling, and billiard. Three application domains in racket sports motivate us: Our primary domain is racket equipment classification (*Domain-1*). The amount of spin a racket imparts on a ball is a significant classification factor. Such classifications can be used in two ways: First, athletes can make objective and deliberate decisions to purchase equipment. Second, sports federations can classify illegal equipment which does not conform to the rules. *Domain-2* is training feedback analysis. Feedback based on ball spin is a useful pointer to improve an athlete's technique. *Domain-3* are virtual replays for television broadcasts of ball sports events. Showing spectators significant ball spin characteristics in virtual replays makes a sport more "tangible" and thereby potentially arouses more interest in the audience. This work focuses only on equipment classification (Domain-1). However, the other two domains are clear long-term goals even though their realization requires an adapted or new approach.

Whereas lots of research (like e.g. [5], [7], and [3]) has been done focused on ball tracking to obtain trajectory paths less past work has dealt with ball spin analysis. Previous work on spin analysis was done in following sports domains: tennis [2], soccer [6], table tennis [9], and baseball [10]. The authors of [2] measure the spin of tennis balls based on high-speed image sequences but favor manual spin measurement over computer vision methods because of higher accuracy. Neilson *et al.* [6] measure the spin of a soccer ball. Their results are based on a unique color pattern on the ball surface where each 2D view of the ball identifies its 3D position. Our approach in contrast works with arbitrary corner features on a ball's surface. Tamaki *et al.* [9] measure ball spin of table tennis balls. Their approach is based on image registration in addition to depth information from a manually fitted 3D sphere model. The work of Boracchi *et al.* [1] examines spin by analyzing blurred images. For the general case of a spinning and translating ball they propose a semi-automatic user-assisted approach. Both [9] and [1] require manual user intervention whereas our approach is fully automatic. Theobalt *et al.* [10] determine the spin of baseballs based on multi-exposure stereo images. Their approach relies on 3D depth data of predefined tracked color markers. We instead only use a single camera and do not need depth information.

Our contribution is a fully automated spin measurement without user intervention. High-speed cameras, as used in our acquisition setup, usually deliver gray scale image data. Therefore, our method relies solely on arbitrary corner features in gray scale image data. We provide measurement results within less than three seconds for 20 processed frames—

sufficient for racket classification application. Further, our method is independent from any motion model, works with uncalibrated, monocular camera data, and does not require a certain ball size. We point out that we only measure spin with a rotational axis perpendicular to the image plane—which makes our approach inappropriate for assessing spin in real game rallies. Although ball trajectory analysis reveals additional discriminative data for racket classification we neglect it in this paper and focus solely on spin measuring.

We explain our video data acquisition setting in Section 2 followed by implemented method details in Section 3. In Section 4 we present and discuss experimental results which are compared to existing approaches in Section 5. Finally, we revise our contribution and give an outlook in Section 6.

2. Video Data Acquisition

We assume our scene under orthography by a large distance between object and image plane and by a large focal length—thereby, perspective distortions are negligible. Hence, motion components towards the image plane cannot be measured with our approach. This poses no limitation for the envisioned racket classification application because we can control the spin axis position in our acquisition setting.

We use rotating table tennis balls as a test environment. Compared to tennis, soccer, baseball, and golf we can reproduce and verify results with less effort due to a simpler data acquisition setting, depicted in Figure 1. A similar setting with a rigidly mounted racket is described in [2]. We use an automatic ball feeder (on the left of Figure 1) to obtain repeatable preconditions. The feeder propels the balls with backspin (3800 ± 100 revolutions per minute (rpm)) towards the rigidly mounted racket from a short distance (0.5 m)—we capture the ball before and after impact on the racket with a high-speed camera. The image plane is parallel to the translational ball motion and the camera observes the ball from 2 m distance (focal length 100 mm). We light the scene with three 1000 W floodlights to achieve enough contrast on the ball contour and on the ball surface features for further processing. The main light direction of all three floodlights is positioned perpendicular to the image plane. The frame rate is 1000 frames per second (fps), the exposure time is $\frac{1}{7000}s$ to minimize motion blur, and the captured image sequences have a resolution of 1280×512 pixels (landscape). Every certified table tennis ball has a printed logo of the manufacturer on its surface. Such a logo covers at most 20% of the whole ball surface. For this reason a single logo might potentially be occluded from the camera viewpoint during an observed scene. Hence, we augment the ball surface with additionally painted arbitrary corner features to ensure visible texture in every captured frame. To make clear, the corner features of a manufacturer logo are good enough for our algorithm as long as they are visible in the scene (we prove this by an example shown in Figure 12 detail (f)).

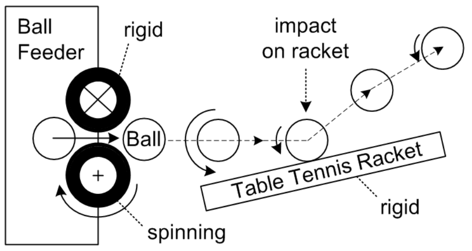


Figure 1. Video data acquisition setting

3. Spin Calculation

Figure 2 depicts the measuring principle with four superimposed frames of a sequence—the ball moves from left to right as in Figure 1. The first two frames are taken prior to the ball impact whereas the last two frames are taken after the impact. For better visibility a yellow dot marks a particular surface corner which is tracked in all four frames (this yellow dot only augments Figure 2 and does not exist on the ball itself). The spin results from the angle the dot has traveled between two frames within an elapsed time. Blue dashed lines mark the ball center in each frame and solid red lines indicate the current angle of the tracked dot with reference to the current ball center. We calculate spin rates for the sequence between frames 13 and 19 as well as between frames 36 and 56. The lower part of the figure sums up the interpretation and calculation: An angle difference of 138,5° within 6 frames corresponds to 3847 rpm whereas an angle difference of 17° within 20 frames corresponds to 141 rpm.

Our basic idea is the calculation of displacements between corresponding corners in two subsequent frames. As visualized in Figure 3 the spin calculation approach consists of the following six steps:

1. Segment the ball from the background.

2. Find the ball contour and calculate its center po-

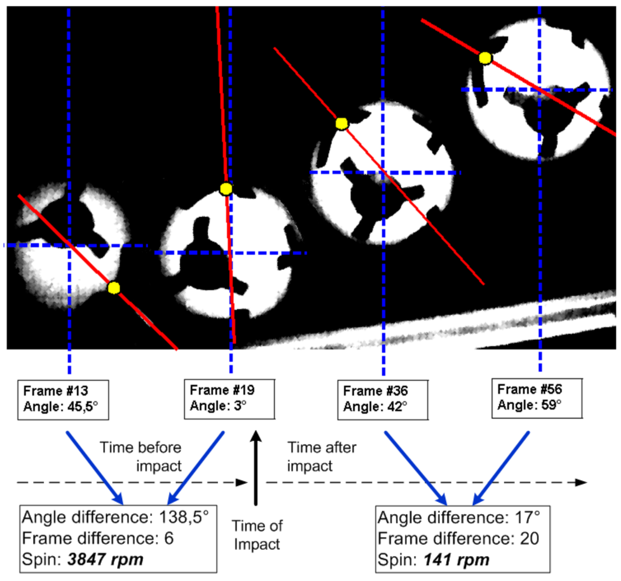


Figure 2. Measurement principle

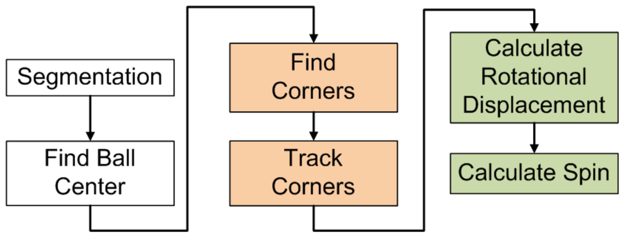


Figure 3. Proposed method for spin calculation

sition.

3. Identify corners within the ball contour.

4. Track identified corners between consecutive frames.

5. Calculate displacement vectors of ball center and displacement vectors of corresponding corners to obtain pure rotational corner displacements by vector subtraction.

6. Measure the angles spanned by rotational corner displacements and calculate spin based on angle per time frame.

In the following we detail the above steps of our approach and explain them according to Figure 4:

Step 1: Segmentation of the ball from the background: In Figure 4 this step transforms the original input image in detail (a) into a binary image in detail (b). To do this, we learn a background model based on frames before a ball becomes visible in the scene. During this learning phase we observe a certain intensity range for each image pixel. After the learning phase a pixel is considered as fore-

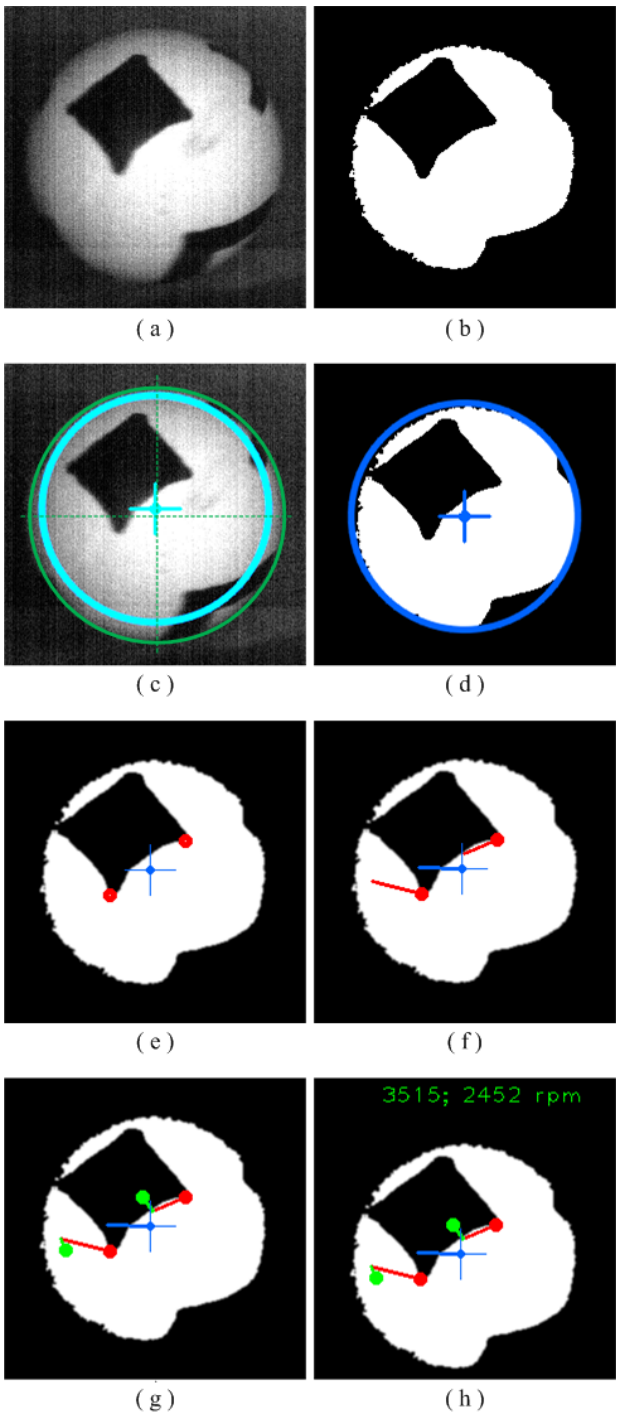


Figure 4. Illustrated method: (a) Original image, (b) segmented image, (c) original image with two different superimposed fitted circles and centers, (d) circle fitted to segmented image, (e) found corners, (f) trace of tracked corners and center, (g) rotational displacements, (h) calculated spins

ground when this pixel's intensity value is outside the learned intensity range. This segmentation method is implemented in *OpenCV* and derived from [4].

Step 2: Finding the ball contour and calculating its center position: The result of this step is visible in Figure 4 when we compare details (b) and (d). In detail (d) we observe an overlaid blue fitted circle to the ball contour as well as the identified ball center. We compute the ball center position in three small steps: First we calculate the convex hull of the segmented ball, second we fit a bounding box around the contour, and third we fit a circle into the bounding box. Finally, the center is obtained from the circle's radius. Detail (c) highlights a problem of accurate center finding: It shows the input image with two superimposed circles and ball centers where the smaller inner circle corresponds to the fitted circle in detail (d). The larger outer circle visualizes the true ball contour; the true ball center is also shown by the dashed cross-hair. In our acquisition setting the ball surface cannot be lit uniformly during the whole sequence. Hence, the contour border has varying contrast with the background. After segmentation the ball contour appears smaller and shifted and thus, the circle is fitted inexactly. The different contour size alone has no negative effect on the accurate center position but a shifted contour also shifts the center position. In Step 5 we will detail why an accurate center position is crucial for an accurate spin calculation.

Step 3: Identifying corners within the ball contour (see red dots in Figure 4 detail (e)): According to the criterion for "good" corners in [8] we identify corners where both eigenvalues of the second moment matrix are above a certain threshold. We set the threshold to 80% of the best found corner's lower eigenvalue. This threshold has been evaluated empirically and ensures "good corner quality".

Step 4: Tracking identified corners between consecutive frames: Figure 4 detail (f) shows the tracked trace of identified corners between subsequent frames with red lines and the tracked center with a blue line. We apply the Kanade-Lucas-Tomasi algorithm [11] for tracking corresponding corners.

Step 5: Calculating rotational displacement: Figure 4 detail (g) highlights the rotational displacement vectors with green elements. Step 2 revealed the ball center displacement and Step 4 revealed displacements of corresponding surface corners. Based on these displacements Figure 5 explains the calculation of the pure rotational displacement based on vector subtraction: Two consecutive frames (top left and middle) are virtually superimposed (top right frame). Solid blue crosses mark the ball center and dashed red crosses mark the tracked corner. The vector subtraction is depicted in the superimposed frame and enlarged below: The ball translation vector in blue is subtracted from the corner displacement vector in red. This results in the pure rotational displacement highlighted in green. The lower left part illustrates this displacement based on the first frame—this is the corner displacement without translation.

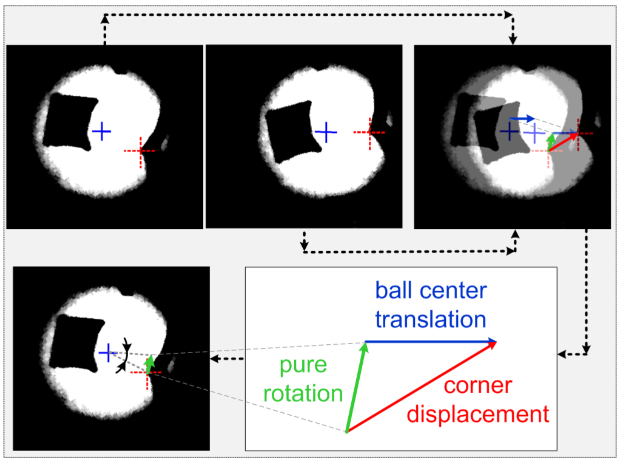


Figure 5. Calculating rotational displacement

Step 6: Calculating spin: Figure 4 detail (h) shows two calculated spin values. A rotational displacement vector together with the ball center spans a certain angle (see lower left frame in Figure 5) which is measured with straightforward trigonometry. The ball rotates by this angle within the time frame of $\frac{1}{1000}s$ (see frame rate). So the spin (in *rpm*) is calculated according to: $spin = \frac{angle \cdot 60s}{360° \cdot timeFrame}$ (angle measured in degrees).

4. Experimental Results

Obtaining ground truth data from real image sequences is a tedious task. Therefore, we generated synthetic image sequences where ground truth is known. Figure 6 visualizes a snapshot of an analyzed synthetic image sequence where the simulated spin is 3667 rpm prior to impact. Three corners of the square-like region are automatically chosen and tracked. The upper right image corner contains the three computed corresponding spin values. Ideally, all three values should be the same, the difference between them indicates inaccuracy. Seven vectors with three different colors are visible, their end points are marked with dots of the same color. According to Figure 5 the ball center translation is shown in blue, the tracked corners' general displacements are shown in red, and the pure rotational corner displacements after vector subtraction of the center translation are

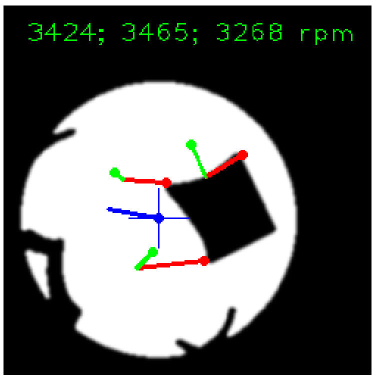


Figure 6. Spin calculation (synthetic image)

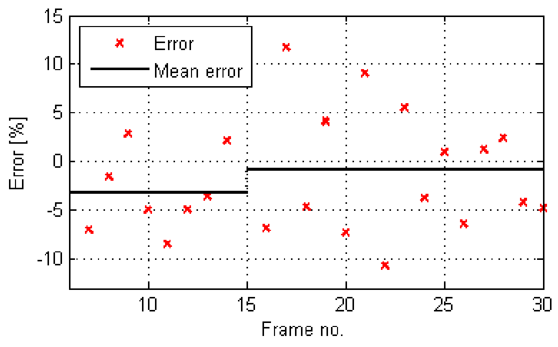


Figure 8. Errors of synthetic sequence

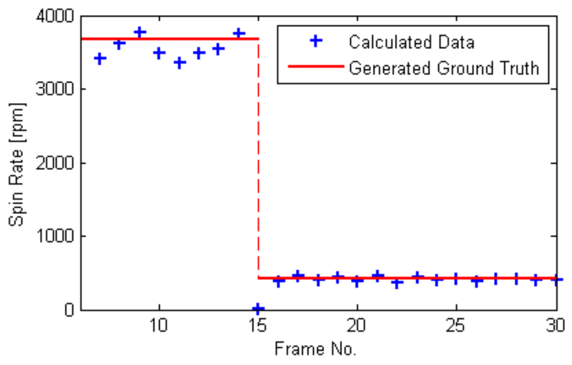


Figure 7. Results of synthetic sequence

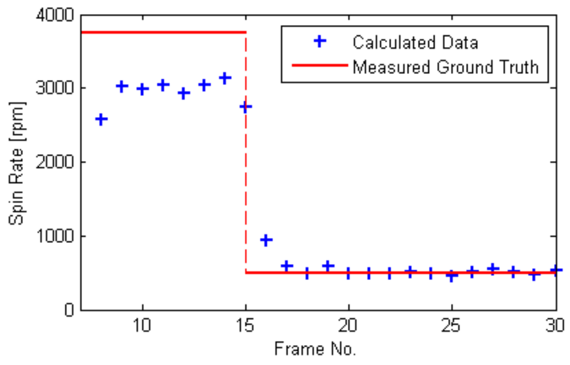


Figure 9. Results of real sequence

shown in green. In this particular snapshot we obtain a mean error of -7.7%, which means that the true spin rate is underestimated.

Figure 7 shows the calculated spins of the synthetic sequence where ground truth spin is 3667 rpm prior to impact and 417 rpm after impact, marked with red lines. Measured values are marked with blue crosses. The value close to zero spin at frame 15 is due to the simulated impact with momentarily zero motion. The values in this diagram represent average values calculated over the number of tracked corners—with reference to Figure 6 this is an average over three values. Of course this simple averaging includes also outliers but we want to show the mean error variation. Figure 8 quantifies the errors of measured spins with reference to the ground truth of the synthetic sequence, detailed in Figure 7. The mean measurement error prior to impact is -3.2% and after impact -0.9%. The relative errors seem to be almost independent from the absolute spin magnitude—the errors are scattered between +12 and -12%.

Figure 4 detail (h) depicts a snapshot of an analyzed real image sequence where the spin is 3750 rpm prior to impact. The mean error of this snapshot is -20.5% and results mainly from inexact center computation because of non-uniform lighting—we explained this in Section 3 (Step 2). Furthermore, although the image acquisition process is completely under our control we cannot ensure a spin axis constantly ideally perpendicular to the image plane in the whole scene. This fact has the most negative effect prior to impact—when the ball's translational and rotational displacement are maximal. None of these two negative effects is existent in synthetic sequences. Between Figure 4 detail (h) and Figure 6 we notice the apparently less smooth ball contour shape of the real snapshot. This results from varying contrast between the projected real ball contour and the background in the real sequence. Only two corners are tracked in the real sequence due to corner correspondence quality. Figure 9 shows the calculated spins of the real sequence. We obtain the ground truth by manually measuring angle differences between corresponding corners in the sequence on a computer display. The spin prior to impact is 3750 rpm and after impact 500 rpm. Figure 10 quantifies the errors of measured spins with reference to the ground truth of the real sequence. As mentioned above the large errors prior to impact result from inexact center computation due to non-uniform

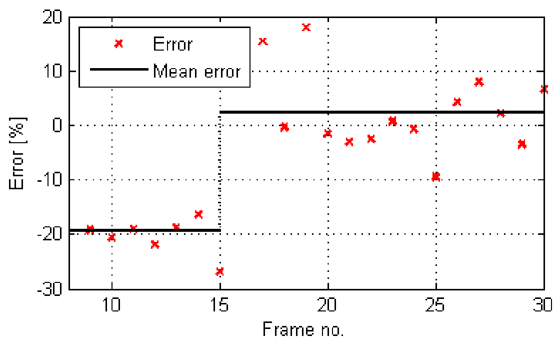


Figure 10. Errors of real sequence

lighting and slightly varying spin axis. The mean measurement error prior to impact is -19.3% but is simultaneously less important. Prior to impact we can assume the ball feeder to generate constant spin through all captured sequences—therefore, spin prior to impact needs not to be measured accurately because no changes are expected. In contrast, after impact, when we expect differences caused by different rackets, the mean error magnitude descends significantly to 2.4%.

We captured eight sequences with five different rackets with results similar to Figure 7 and Figure 9 (these measurements are not shown in detail). We call the measured spin of a racket after impact its *spin response* to a certain spin before impact. Measured spin responses revealed an average range per sequence between 200 and 1250 rpm depending on the racket. Figure 11 illustrates these measurements where Racket 3 corresponds to results of Figure 9 after impact. In the case of these five rackets the spin response is discriminative but insufficient to uniquely classify the rackets. Further measurements based on ball trajectory can enhance the discriminative power of the classification but these rather straightforward measurements are not the scope of this paper.

In Figure 12 we prove that our method can cope with different surface patterns in real image sequences—ten processed snapshots are depicted. Details (a) to (f) have been obtained with the setting described in Section 2. In contrast details (g) to (j) correspond to an earlier experimental setting in which balls move from right to left (observe the center trace) towards a not rigidly fixed racket. Detail (f) is exceptional because we have not added artificial surface patches. Nevertheless, we are able to calculate the spin based solely on the visible part of the manufacturer logo on the surface.

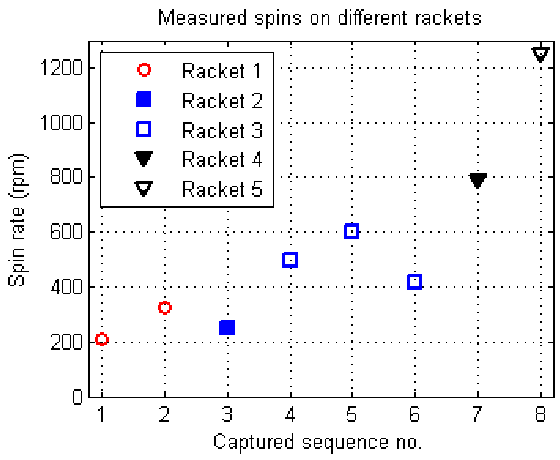


Figure 11. Spin responses with five different rackets

5. Comparison

To reveal our strengths and limitations we compare our results with two approaches ([1], [10]): The blur approach of Boracchi *et al.* [1] requires a feature to have an observed angle displacement of at least 3.6°—our method does not have a lower bound. In [1] the authors assessed only cases without translation where the mean error was 3 - 11% for a spin range 833 - 1666 rpm. In contrast to them we cope with additionally superimposed translations. Theobalt *et al.* [10] state an error of 0.4 - 2.5% for spins of 1258 - 1623 rpm for their stereo based approach. In contrast to both approaches we cope with a larger spin range between 0 - 3750 rpm. On the other hand our mean error magnitude can increase to about 20% but only during less important measurements prior to impact as mentioned above in connection with Figure 9.

6. Conclusion and Outlook

We have shown a motion analysis approach focused on the measurement of ball spin. Experiments proved different spin responses on different rackets which makes this method's results feasible for racket classification based on spin measurements. A sequence of 20 captured frames is sufficient for a significant racket classification. The execution time for processing 20 frames is about 3 seconds (run on an Intel Core i7 L620, 2 GHz processor)—this delay is acceptable for an application like on site classification of illegal rackets during sports events. However, the same delay might be an upper limit for application domains like training feedback and virtual replays for sports broadcasts.

We identify two major future steps:

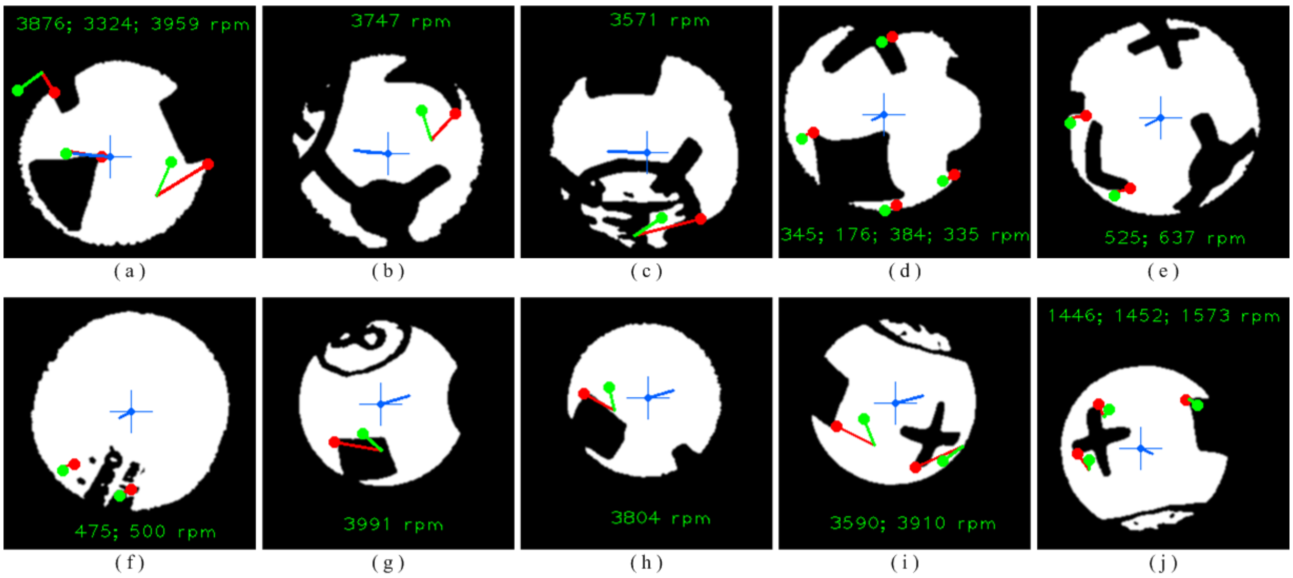


Figure 12. Snapshots of different sequences: (a) to (c) Results prior to impact, (d) to (f) results after impact, (g) to (i) results prior to impact (different setting), (j) results after impact (different setting)

- Most importantly, we strive for measuring spin without restrictions on the spin axis position. Thus, an adapted or new approach is crucial for the intended two long-term applications, training feedback and virtual replays.

- Our method should be sufficiently robust to measure spin based on any two subsequent frames. We will successively challenge this robustness by decreasing the number of artificial surface features.

Acknowledgements

We would like to thank Professor Arnold Baca and Philipp Kornfeind who supported and helped with the video data acquisition which enabled our measurements. All videos have been captured at the laboratory of the Department of Biomechanics, Kinesiology and Applied Computer Science at the University of Vienna.

References

[1] G. Boracchi, V. Caglioti, and A. Giusti. Estimation of 3d instantaneous motion of a ball from a single motion-blurred image. In *VISIGRAPP*, pages 225–237, 2009. 1, 6

[2] S. R. Goodwill and S. J. Haake. Ball spin generation for oblique impacts with a tennis racket. *Experimental Mechanics*, 44(2):195–206, 2004. 1, 2

[3] A. Guéziec. Tracking pitches for broadcast television. *IEEE Computer*, 35(3):38–43, Mar. 2002. 1

[4] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis. Real-time foreground-background segmentation using codebook model. *Real-Time Imaging*, 11(3):172 – 185, 2005. Special Issue on Video Object Processing. 3

[5] T. Kim, Y. Seo, and K.-S. Hong. Physics-based 3d position analysis of a soccer ball from monocular image sequences. In *Sixth International Conference on Computer Vision (ICCV'98)*, pages 721–726, Jan. 1998. 1

[6] P. Neilson, R. Jones, D. Kerr, and C. Sumpter. An image recognition system for the measurement of soccer ball spin characteristics. *Measurement Science and Technology*, 15(11):2239–2247, 2004. 1

[7] G. Pingali, A. Opalach, and Y. Jean. Ball tracking and virtual replays for innovative tennis broadcasts. In *International Conference on Pattern Recognition (ICPR'00)*, pages 152–156, 2000. 1

[8] J. Shi and C. Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 593–600, 1994. 4

[9] T. Tamaki, T. Sugino, and M. Yamamoto. Measuring ball spin by image registration. In *Proceedings of the Tenth Korea-Japan Joint Workshop on Frontiers of Computer Vision*, pages 269–274, 2004. 1

[10] C. Theobalt, I. Albrecht, J. Haber, M. Magnor, and H.-P. Seidel. Pitching a baseball - tracking high-speed motion with multi-exposure images. In *Proceedings of ACM SIGGRAPH*, 2004. 1, 6

[11] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision (IJCV)*, 9(2):137–154, 1992. 4

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

Robustifying the Flock of Trackers

Jiří Matas and Tomáš Vojíř
The Center for Machine Perception, FEE CTU, Prague
Karlovo namesti 13, 121 35 Praha 2, Czech Republic
{matas, vojirtom}@cmp.felk.cvut.cz

Abstract. *The paper presents contributions to the design of the Flock of Trackers (FoT). The FoT trackers estimate the pose of the tracked object by robustly combining displacement estimates from local trackers that cover the object.*

The first contribution, called the Cell FoT, allows local trackers to drift to points good to track. The Cell FoT was compared with the Kalal et al. Grid FoT [4] and outperformed it on all sequences but one and for all local failure prediction methods.

As a second contribution, we introduce two new predictors of local tracker failure - the neighbourhood consistency predictor (Nh) and the Markov predictor (Mp) and show that the new predictors combined with the NCC predictor are more powerful than the Kalal et al. [4] predictor based on NCC and FB.

The resulting tracker equipped with the new predictors combined with the NCC predictor was compared with state-of-the-art tracking algorithms and surpassed them in terms of the number of sequences where a given tracking algorithm performed best.

1. Introduction

Tracking is an important task in computer vision. Given two consecutive frames and the position of an object in the first frame, the task is to estimate the pose of the object in the second frame. In a video sequence, tracking is the task of estimation the full trajectory of the object.

There are many approaches addressing the problem. This paper focuses on the so-called *Flock of Trackers* (FoT). The Flock of Trackers is a tracking approach where the object motion is estimated from the displacements, or, more generally, transformation estimates, of a number of independent local trackers covering the object.

Each local tracker is attached to a certain area specified in the object coordinate frame. The local trackers are not robust and assume that the tracked area is visible in all images and that it all undergoes a simple motion, e.g. translation. The Flock of Trackers object motion estimate is robust if it is obtained by a combination of local tracker motions which is insensitive to failures.

This idea was utilized in the Median-Flow (MF) [4] tracker and was shown to be comparable to state-of-the-art trackers. In [4], the FoT is based on local trackers placed on a regular grid, i.e. the local trackers cover the object uniformly. The object motion, which is assumed to be well modelled by translation and scaling, is estimated by the median of a subset of local tracker responses.

Theoretically, the median is robust up to 50% of outliers for translation and $100 \times (1 - \sqrt{0.5})\%$ for scale estimation which is based on pairs of correspondences. In practice, the outlier tolerance is often higher since the outlier do not "conspire" and are not all above or below the median. Nevertheless, in challenging tracking scenarios, the inlier percentage was not sufficient and the median failed.

In order to robustify the FoT, [4] proposed several local tracker filtering methods which perform the task of finding and removing probable outliers before the median estimation. The outlier subset of local trackers is selected by a failure-predicting procedure which takes into account the following quantities: the normalised cross-correlation of the corresponding patches (NCC), the sum of squared differences (SSD) and the consistency of the so called forward-backward procedure (FB). The forward-backward procedure runs the Lucas-Kanade tracker [5] twice, once in the forward direction and then in the reverse direction. The probability of being an oulier is a function of the distance of the starting point and the point reached by the FB procedure.

The paper present two contributions to the design of the Flock of Trackers. First we show that superior tracking results are achieved if the failed local trackers are not reset to the grid position and they are allowed to track those areas of the object they drifted to. This process can be viewed as a new method for selecting ”good points to track” on the object which can be described as ”good points to track are those the tracker drifted to”.

Second, we propose two new predictors of local tracker failure - the neighbourhood consistency predictor (Nh) and the Markov predictor (Mp). The Nh predictor is based on the idea that a correct local tracker will return a displacement similar to its neighbours. The Markov predictor exploits temporal consistency, a local tracker that performed well in the recent past is likely to perform well on the current frame and vice versa.

We show the new predictors combined with the NCC predictor are more powerful than the Kalal et al. [4] predictor based on NCC and FB. Moreover, the new predictors are efficiently computed, at a cost of about 10% of the complete FoT procedure whereas the forward-backward procedure slows down tracking approximately by a factor of two, since the most time consuming part of the process, the Lucas-Kanade local optimization, is run twice. With the proposed precise failure predictors, a FoT with much higher robustness to local tracker problems is achieved with negligible extra computational cost.

The rest of the paper is structured as follows. Section 2 discussed placement of the local trackers in FoT and present the cell placement strategy. Section 3 propose two new predictors of local tracker failure. Finally, Section 4 evaluate proposed improvements and compare resulting MF tracker with the state-of-the-art tracking algorithms and conclusion is given in Section 5.

2. Local tracker placement in FoT

The task of object tracking is usually decomposed into two steps. First, interesting points to track are found (e.g. ”the good features to track” [8]). Next, the selected points are tracked. In the case of FoT, Kalal et al. [4] omit the first step and chooses the points to evenly cover the object of interest - the local trackers are laid out on a regular grid. It is clear that not all local trackers will be placed at location suitable for tracking. The poorly placed local trackers drift away from the original position on the grid. The Grid FoT (as proposed in [4]) resets, after estimating the global object motion, all local trackers to their original place in grid.

We argue that this is a suboptimal approach, because local trackers reinitialized to the same position unsuitable for tracking will drift again. Instead, we propose the Cell FoT, where each local tracker is allowed to ”find” a suitable offset from its default position in the grid, see Fig. 1. The local trackers are forced to stay in their cells, and thus guaranteeing to evenly cover the tracked object, but within the cell the local trackers are let to assume the best position for tracking.

To avoid tracking points that are near each other, which could make the local trackers dependent and likely to fail simultaneously, the cells within which a local tracker must stay may not completely cover the object, as depicted in Fig.1. In preliminary experiments, we observed that the cell parameters c_w and c_h have a noticable influence on the results. However, to keep the method simple, in experiments reported in this paper they are set to the grid resolution, i.e. the local trackers are allowed to assume any position on the object (the cells cover the object).

The improvement of the FoT achieved by the Cell method is demonstrated in experiments in Section 4.1.

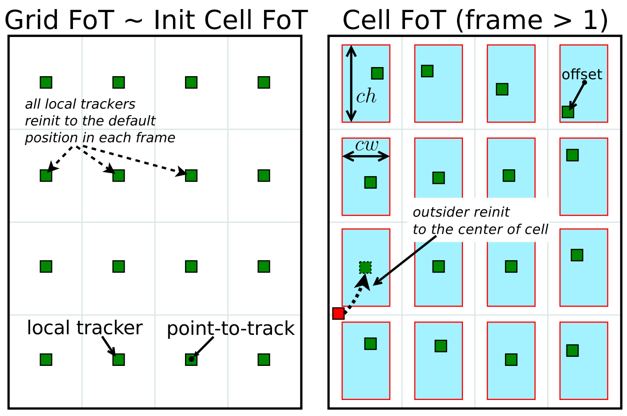

Grid FoT ~ Init Cell FoT
Cell FoT (frame > 1)
all local trackers reinit to the default position in each frame
local tracker
point-to-track
ch
cw
offset
outsider reinit to the center of cell

Figure 1: A comparison of the Grid and the Cell FoT. In the Grid FoT, in every frame, local trackers are placed on a regular grid. In the Cell FoT, after the first frame, the location the local tracker drifted to is tracked - the offset with respect to the grid positions is stored. Only local trackers that drifted away form their cells are reset.

3. New failure prediction methods

In this section, two novel methods for the local tracker failure prediction are presented together with a method that combines them with a predictor based on normalised cross-correlation and achieves very high accuracy at a very low computational cost. This new Median-Flow tracker with the new T_Σ combined predictor is evaluated in Section 4.2. The section is structured as follows: Section 3.1 describes the Neighbourhood constraint, Section 3.2 present an orthogonal predictor based on temporal local trackers behaviour.

3.1. Neighbourhood consistency constraint predictor

The idea of the neighbourhood constraint predictor Nh is that the motion of neighbouring local trackers is very similar, whereas failing predictors return a random displacement. The idea corresponds to the *smoothness assumption* which is commonly used in optic flow estimation, e.g. [7].

The Nh predictor was implemented as follows. For each point i, a neighbourhood N_i is defined. In all experiments, N_i was the four neighbourhood of i (three points are used on the edge, two in the corner of the grid). The neighbourhood consistency score S_i^{Nh}, i.e. number of the neighbourhood local trackers that have similar displacement, is calculated for each point i as follows:

$$S_i^{Nh} = \sum_{j \in N_i} [\| \Delta_j - \Delta_i \|^2 < \varepsilon]$$
$$\text{where} \quad [expression] = \begin{cases} 1 & \text{if } \textit{expression} \text{ is true} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

and where ε is the displacement difference threshold, and Δ_i is the displacement of local tracker *i*. A local tracker is defined consistent if $S_i^{Nh} \geq \theta$. The value of θ was set to 1. The displacement difference threshold ε was set to 0.5 pixels. The process is visualised in Fig. 2).

3.2. The Markov predictor

The Markov predictor (Mp) exploits a simple model of the past performance of a local tracker. The model is in the form of a Markov chain (Fig. 3) with two states, state = $\{$*inlier=1*, *outlier=0*$\}$.

The predicted state of the local tracker depends on its state in the previous time instance and the transition probabilities. Transition probabilities are computed incrementally, from frame to frame. Each local

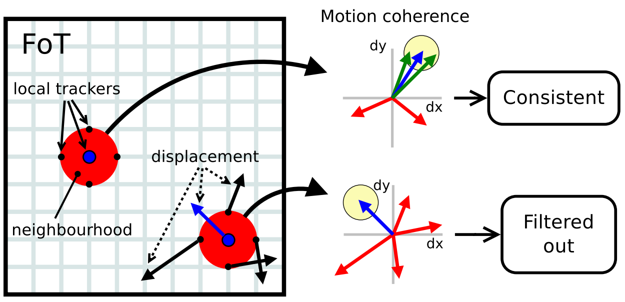
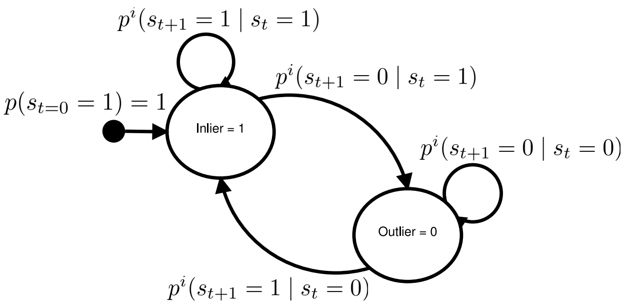


Figure 2: The neighbourhood consistency tracking failure predictor. The neighbourhood (red area) of the local tracker (blue circle) is explored for motion coherence. Blue arrow - center local tracker displacement, green arrows - local trackers with coherent motion, red arrows - local tracker with incoherent motion.

tracker *i* in time *t* is modeled as transition matrix $\mathbf{T}_t^i$ described in eq. 2.

$$\mathbf{T}_t^i = \begin{bmatrix} p^i(s_{t+1} = 1 \mid s_t = 1) & p^i(s_{t+1} = 1 \mid s_t = 0) \\ p^i(s_{t+1} = 0 \mid s_t = 1) & p^i(s_{t+1} = 0 \mid s_t = 0) \end{bmatrix} \tag{2}$$

where s_t is the current state of the local tracker and sums in columns are equal to 1. Prediction that cer-



$p^i(s_{t+1} = 1 \mid s_t = 1)$
$p^i(s_{t+1} = 0 \mid s_t = 1)$
$p(s_{t=0} = 1) = 1$
Inlier = 1
$p^i(s_{t+1} = 0 \mid s_t = 0)$
Outlier = 0
$p^i(s_{t+1} = 1 \mid s_t = 0)$

Figure 3: The state diagram of the Markov chain for the local tracker in the generalized form of two states probabilistic automaton with transition probabilities p^i, where *i* identifies the local tracker.

tain local tracker would be inlier (or outlier) is dual task to next state prediction in Markov chain. To predict next state in time *t*+1 we compute probability of crossing to state 1 with apriori of current state. This is done according to equation 3.

$$\begin{bmatrix} p^i(s_{t+1} = 1) \\ p^i(s_{t+1} = 0) \end{bmatrix} = \mathbf{T}_t^i \times \begin{bmatrix} p^i(s_t = 1) \\ p^i(s_t = 0) \end{bmatrix} \tag{3}$$

where $p(s_t = 1) = 1$ if current state is inlier, 0 otherwise (likewise for $p(s_t = 0)$). The left side of equation 3 are then probabilities that next state would be inlier (outlier) (e.g. if $p(s_{t+1} = 1) = 0.6$, then we considered local tracker as inlier in 60% of cases).

Model update is equal task to estimation of transition probabilities $p^i(s_{t+i} = 1 \mid s_t = 1)$ and $p^i(s_{t+i} = 1 \mid s_t = 0)$. These probabilities are updated in each frame as follow :

$$p^i(s_{t+1} = 1 \mid s_t = 1) = \frac{n^i_{11}}{n^i_1} \qquad p^i(s_{t+1} = 1 \mid s_t = 0) = \frac{n^i_{01}}{n^i_0} \tag{4}$$

where n_1 (n_0) are relative frequency for the local tracker *i* being inlier (outlier), and n_{11} (n_{01}) are relative frequency for event that the local tracker *i* was inlier (outlier) in the time *t* and inlier in the time *t*+1, for $t \in (0, t\rangle$. The current state of the local tracker being inlier (outlier) is obtained by fitting the local tracker correspondence to estimated global motion (by MF) and thresholding their errors.

4. Performance evaluation

The performance of the proposed FoT was tested on challenging video sequences with object occlusion (or disappearance), illumination changes, fast motion, different object sizes and object appearance variance. The sequences are described in Tab. 1.

In the experiments, the predictor of neighbourhood consistency (Nh) and the Markov predictor (Mp) were run as explained in Section 3. The sum of squared differences (SSD), normalized cross-correlation (NCC) and the forward-backward predictor (FB) rank local trackers by their score and treat the top 50% as inliers. Predictors are denoted by the names of their error measure, except for the combination Mp+NCC+Nh which is abbreviated to Σ.

A frame is considered correctly tracked if the overlap with ground truth is greater than 0.5, with the exception of experiment 4.4 where influence of the initialization of the tracker was assessed. Since in this case the bounding boxes are randomly generated and may not fully overlap the object, the threshold was lower to 0.3, see Fig. 6.

4.1. Cell FoT-MF vs. Grid FoT-MF

Two version of the FoT that differ by local tracker placement — the Cell FoT-MF and the Grid FoT-MF — were compared on sequences presented in Tab. 1.

The tests were carried out for all individual predictors and for most combination of local tracker failure predictors; some combinations, e.g. SSD and NCC, make no sense since the predictions are highly correlated.

The performance was first measured by the length of correctly tracked sub-sequences (Tab. 2) and, on a single sequence, by the overlap of the estimated object pose and the ground truth bounding box (Fig. 4).

According to both criteria, the Cell FoT outperform Grid FoT for almost all sequences and failure prediction methods. We therefore conclude that the Cell FoT is superior to the Grid FoT.

Finally, the Grid and the Cell trackers were compared by the fraction of local trackers that are inliers to the global object motion, see Tab. 3. Again, the Cell method dominates. The interpretation of this result is not straightforward since the median and mean inlier rates are calculated on the whole correctly tracked sub-sequences, which are different for different method.

4.2. Comparison of failure prediction methods

We compared performance of individual predictors and combinations FB+NCC (as proposed in [4]) and Σ within the MF tracker on sequences presented in Tab. 1. Performance was measured in terms the length of the subsequence that was correctly tracked and by the number of sequences where a given tracker failed last (Tab. 2 last row). All parameters for Nh was fixed for all sequences, as described in Section 3.1. The proposed local tracker failure predictor Σ outperform FB+NCC on all tested sequences.

4.3. Comparison of T_{FB+NCC} and T_Σ speed

The MF tracker is intended for real-time performance and thus the speed of local tracker predictor is important. The experiment was performed on a subset of sequences listed in Tab. 1 (where the trackers successfully tracked the whole sequence) and then the results were averaged. Speed was measured as the average time needed for frame-to-frame tracking (Tab. 4). Processing time for I/O operations, including image loading, and other tasks not relevant to tracking, was excluded. The MF with Σ predictor performs 58% faster than FB+NCC. Moreover, the Σ overhead is negligible compared to reference MF tracker (i.e. MF without any predictor).

4.4. Robustness to bounding box initialization

For a tracker, it is highly desirable not to be sensitive to the initial pose specified by the object bound-

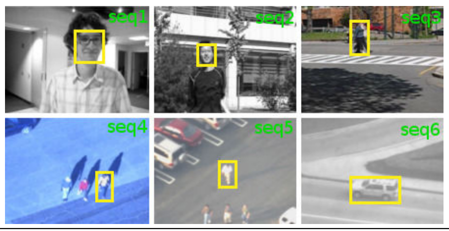
Sequence	name	frames	First appeared in	preview
1	David	761	D.Ross et al., IJCV'08 [6]	
2	Jumping	313	Q. Yu et al., ECCV'08 [9]	
3	Pedestrian 1	140	S. Avidan, PAMI'07 [1]	
4	Pedestrian 2	338	Q. Yu et al., ECCV'08 [9]	
5	Pedestrian 3	184	Q. Yu et al., ECCV'08 [9]	
6	Car	945	Q. Yu et al., ECCV'08 [9]	

Table 1: The description of test sequences and sample images with the selected object of interest.

Sequence	$T^g_\emptyset/T^c_\emptyset$	$T^g_{\mathrm{SSD}}/T^c_{\mathrm{SSD}}$	$T^g_{\mathrm{NCC}}/T^c_{\mathrm{NCC}}$	$T^g_{\mathrm{FB}}/T^c_{\mathrm{FB}}$	$T^g_{\mathrm{Nh}}/T^c_{\mathrm{Nh}}$	$T^g_{\mathrm{Mp}}/T^c_{\mathrm{Mp}}$	T^g_Σ/T^c_Σ	$T^g_{\mathrm{FB+NCC}}/T^c_{\mathrm{FB+NCC}}$
1	296/476	63/529	133/479	**761/761**	**761/761**	597/700	453/**761**	**761/761**
2	36/36	78/**79**	49/56	45/76	33/34	36/36	76/76	36/36
3	14/12	20/26	29/33	34/38	15/27	28/27	125/**140**	45/49
4	90/90	33/33	90/90	**264**/90	90/90	90/90	153/**264**	90/90
5	**52/52**	**52/52**	**52/52**	**52/52**	**52/52**	**52/52**	**52/52**	**52/52**
6	389/345	290/374	**510/510**	**510/510**	**510/510**	**510/510**	**510/510**	**510/510**
best	1/1	1/2	2/2	4/3	3/3	2/2	2/5	3/3

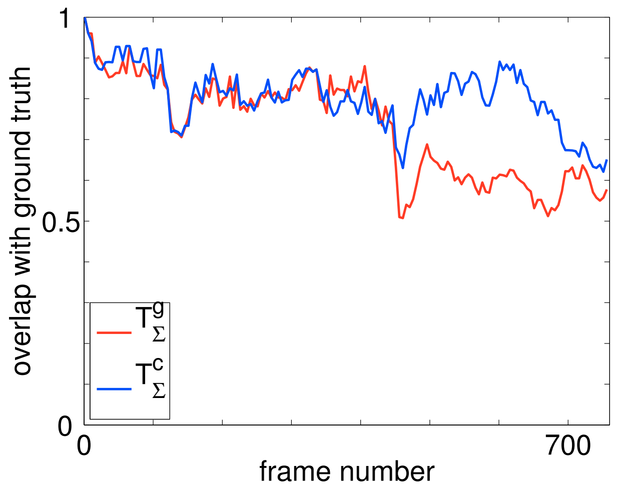
Table 2: A comparison of the Grid (T^g) and the Cell (T^c) FoT-MF with different local tracker failure predictors in term of the length of the successfully tracked subsequences. Best results for each sequence are in bold. Row *best* shows the number of sequences where the tracking method perform best.The total number of a frames in the sequences are listed in Tab. 1 The T^c_Σ clearly dominates.

Sequence	$T^g_\emptyset/T^c_\emptyset$	$T^g_{\mathrm{SSD}}/T^c_{\mathrm{SSD}}$	$T^g_{\mathrm{NCC}}/T^c_{\mathrm{NCC}}$	$T^g_{\mathrm{FB}}/T^c_{\mathrm{FB}}$
1	0.41(0.45)/**0.51(0.55)**	0.12(0.13)/**0.52(0.55)**	0.26(0.25)/**0.50(0.53)**	0.57(0.61)/**0.60(0.64)**
2	0.64(0.76)/**0.66(0.77)**	0.47(0.45)/0.47(0.45)	**0.55(0.63)**/0.52(0.55)	**0.60(0.70)**/0.50(0.54)
3	0.33(0.33)/**0.35**(0.33)	**0.40(0.43)**/0.39(0.40)	0.41(0.40)/**0.43(0.43)**	0.42(0.40)/**0.43**(0.40)
4	0.52(0.53)/**0.53(0.54)**	0.47(0.48)/**0.54(0.56)**	0.52(0.51)/**0.53(0.52)**	0.52(0.53)/0.53(0.52)
5	0.68(0.71)/0.68(**0.72**)	0.70(**0.74**)/0.70(0.73)	0.67(0.69)/**0.68(0.73)**	0.68(0.71)/0.68(**0.73**)
6	0.74(0.81)/**0.75**(0.81)	0.74(0.81)/0.74(0.81)	0.74(0.81)/**0.75(0.82)**	0.74(0.81)/0.75(0.82)
# better	0/**6**	2/2	1/**5**	1/**4**
Sequence	$T^g_{\mathrm{Nh}}/T^c_{\mathrm{Nh}}$	$T^g_{\mathrm{Mp}}/T^c_{\mathrm{Mp}}$	T^g_Σ/T^c_Σ	$T^g_{\mathrm{FB+NCC}}/T^c_{\mathrm{FB+NCC}}$
1	0.58(0.62)/**0.61(0.66)**	0.47(0.50)/**0.53(0.57)**	0.56(0.60)/**0.59(0.63)**	0.54(0.57)/**0.56(0.59)**
2	**0.68**(0.78)/0.65(0.78)	0.65(0.76)/**0.66**(0.76)	0.48(**0.49**)/0.48(0.44)	**0.65(0.79)**/0.64(0.76)
3	0.40(0.40)/0.41(0.39)	0.40(**0.39**)/0.40(0.38)	0.34(0.33)/**0.35(0.35)**	0.43(0.43)/0.43(0.43)
4	0.53(0.52)/**0.54(0.55)**	0.52(0.51)/0.52(**0.52**)	0.51(0.51)/**0.52**(0.51)	0.50(0.49)/**0.51**(0.49)
5	0.68(0.72)/0.68(0.72)	0.68(0.72)/0.68(0.72)	0.67(0.71)/**0.68**(0.71)	0.67(0.70)/0.67(0.70)
6	0.74(0.81)/**0.75**(0.81)	0.74(0.81)/**0.75(0.82)**	0.74(0.80)/**0.75(0.82)**	0.74(0.81)/**0.75(0.83)**
# better	1/3	1/4	1/5	1/3

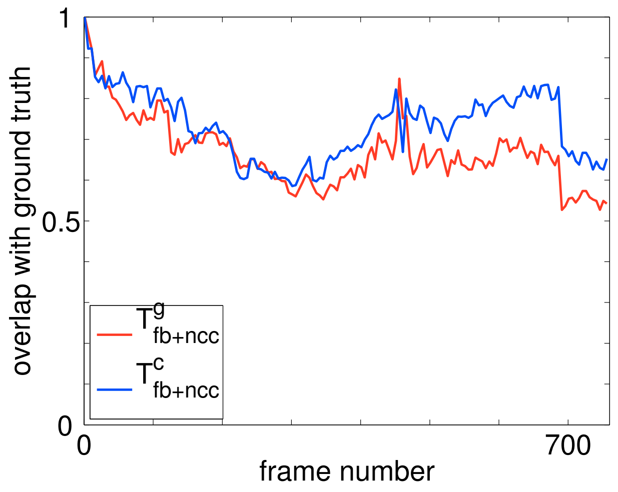
Table 3: A comparison of the Grid (T^g) and the Cell (T^c) FoT-MF in terms of inliers rates, i.e. the fraction of local trackers consistent with the estimated global object motion. Entries are in the following format: *mean*(*median*) of inliers rates for the correctly tracked sub-sequences. Row *# better* shows the number of sequences where the grid/cell methods perform better then the other one.

ing box as it is often selected manually, with unknown precision.

If part of the bounding box does not cover the object, the Mp predictor should soon discover that the local trackers are consistently in the outlier set. The Mp predictor can be used to define the object more precisely, e.g. as the set of cells that are likely to be inliers, according to Mp. Also, with Mp, there



(a)



(b)

Figure 4: A comparison of the Grid and the Cell trackers with (a) T_Σ (b) $T_{\text{FB+NCC}}$ in terms of the overlap with ground truth as a function of time for the sequence 1 (David).

Method	f[Hz]	T [ms]
$T_\emptyset \approx T_{\text{SSD}}$ [ref]	227	4.41
$T_{\text{FB+NCC}}$	131	7.63
T_Σ	207	4.83

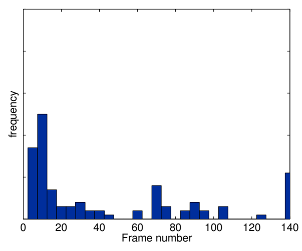
Table 4: A comparison of the speed of tracking failure prediction methods for the MF tracker.

is hope that the global tracker will be insensitive to initialization.

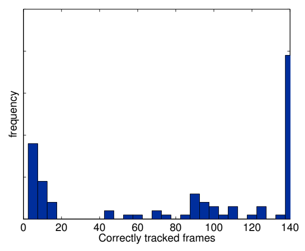
This experiment tested this assumption on the challenging sequence 3 (Pedestrian 1). We generated randomly 100 initial bounding boxes overlapping the object of interest (Fig. 6) and count the correctly tracked frames (Tab. 5). In this experiment, the frame was declared correctly tracked if overlap with ground truth was greater than 0.3. The T_Σ tracker perform about 90% better than $T_{\text{FB+NCC}}$ and was able to track the object correctly up to frame 85 in average. Figs. 5a and 5b show the histograms of the number of correctly tracked frames and fig. 5c show 2D histogram of the corresponding numbers of correctly tracked frames.

	Score	mean (median)
$T_{\text{FB+NCC}}$ [ref]	4493	45 (21)
T_Σ	8438	84.4 (99.5)

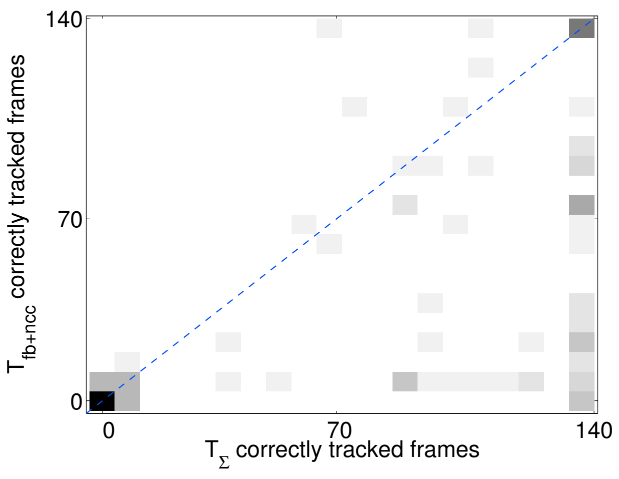
Table 5: Evaluation of filtering methods in terms of the number of correctly tracked frames with randomly initialized bounding box (see. Fig. 6). Score is the total number of correctly tracked frames, the mean and the median of the same quantity is presented in the right column.



(a)



(b)



(c)

Figure 5: Histograms of the number of correctly tracked frames for (a) $T_{\text{FB+NCC}}$, (b) T_Σ and the (c) 2D histogram of the corresponding numbers of correctly tracked frames for different bounding box initializations.

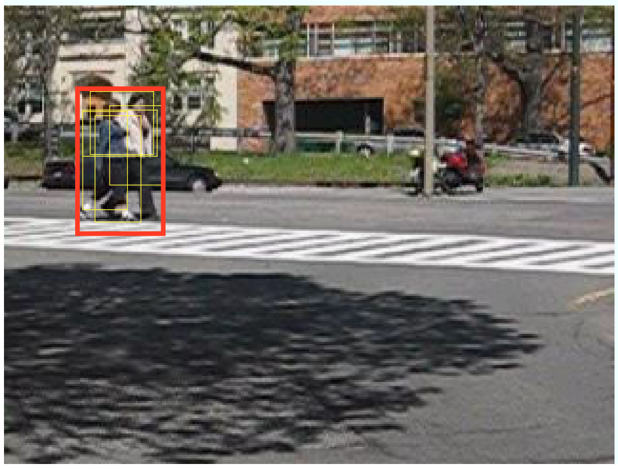


Figure 6: Exmaples of randomly generated initial bounding boxes (yellow) which were randomly generated within the red rectangle.

4.5. Comparison with state-of-the-art approaches

The proposed method was compared with the state-of-the-art algorithms. Results of the experiment are presented in Table 6. The key results is presented in the bottom (denoted *best*). The number is defined as number of sequences where the given tracking algorithm perform best. Results for algorithms [6, 3, 1, 2] were obtained from [4]. The experiment follows [4] - each frame was considered tracked correctly if overlap with ground truth was bigger than 0.5. Object initialization was done by the ground truth. The proposed Σ predictor with MF tracker outperform state-of-the-art algorithms and proof to be superior in speed and robustness to FB+NCC (as was shown in sections 4.3, 4.4), which perform similarly.

sequence	[6]	[3]	[1]	[2]	[4]	T_Σ
1	17	n/a	94	135	**761**	**761**
2	75	**313**	44	**313**	170	76
3	11	6	22	101	**140**	**140**
4	33	8	118	37	97	**264**
5	50	5	**53**	49	52	52
6	163	n/a	10	45	**510**	**510**
best	0	1	1	1	3	**4**

Table 6: A comparison of the proposed MF T_Σ tracker with recently published tracking methods.

5. Conclusions

This paper presented an improvement of the Flock of Trackers, the so called Cell FoT that allows local trackers to drift to points good to track. The Cell FoT was compared with the Grid FoT [4] and outperformed it on all sequences but one and for all local failure prediction methods.

As a second contribution, two new local tracker failure predictors were introduced - the neighbourhood consistency predictor and the Markov predictor. Together with the NCC predictor, they formed a very strong predictor Σ. The Σ predictor was compared with the FB+NCC predictor in the framework of the Median Flow. The Σ predictor outperformed the FB+NCC in all criteria, ie. in terms of speed, the length of correctly tracked sequences and the robustness to bounding box initialization.

The MF tracker equipped with Σ predictor was compared with state-of-the-art tracking algorithms and surpassed them in terms of the number of sequences where a given tracking algorithm performed best.

Acknowledgements

The authors were supported by Czech Science Foundation Project P103/10/1585 and by Czech Government reseach program MSM6840770038

References

[1] S. Avidan. Ensemble tracking. *PAMI*, 29(2):261–271, 2007. 5, 7

[2] B. Babenko, M.-H. Yang, and S. Belongie. Visual Tracking with Online Multiple Instance Learning. In *CVPR*, 2009. 7

[3] R. Collins, Y. Liu, and M. Leordeanu. On-line selection of discriminative tracking features. *PAMI*, 27(1):1631 – 1643, October 2005. 7

[4] Z. Kalal, K. Mikolajczyk, and J. Matas. Forward-Backward Error: Automatic Detection of Tracking Failures. *ICPR*, 2010. 1, 2, 4, 7

[5] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision (ijcai). In *IJCAI*, pages 674–679, April 1981. 1

[6] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *IJCV*, 77(1-3):125–141, 2008. 5, 7

[7] P. Sand and S. J. Teller. Particle video: Long-range motion estimation using point trajectories. *IJCV*, 80(1):72–91, 2008. 3

[8] J. Shi and C. Tomasi. Good features to track. In *CVPR*, pages 593 – 600, 1994. 2

[9] Q. Yu, T. B. Dinh, and G. G. Medioni. Online tracking and reacquisition using co-trained generative and discriminative trackers. In *ECCV*, pages 678–691, 2008. 5

16th Computer Vision Winter Workshop
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

3D Geometric Shape Modeling by '3D Contour Cloud' Reconstruction from Stereo Videos

Kerstin Pötsch and Axel Pinz
Institute of Electrical Measurement and Measurement Signal Processing
Graz University of Technology
Austria
kerstin.poetsch@tugraz.at, axel.pinz@tugraz.at

Abstract. *We present a shape representation for objects based on 3D contour fragments that build a '3D Contour Cloud'. Our approach for the automatic reconstruction of such '3D Contour Clouds' from calibrated stereo image sequences includes a novel idea on the reconstruction of 3D contour fragments from stereo frame pairs and a robust Structure and Motion analysis. Moreover, we propose an extension of 2D shape context to 3D – 3D shape context – which we use for outlier analysis. We show results on a standard multi-view stereo dataset as well as on our own stereo dataset. We achieve qualitatively convincing '3D Contour Clouds' for various objects including the automatic generation of 3D bounding boxes and object-centered coordinate systems.*

1. Introduction

3D shape modeling gains more and more importance in computer vision and graphics in terms of shape retrieval, shape matching, object recognition and categorization. Especially in object categorization, shape features can provide powerful cues to represent object categories, and 2D shape information has successfully been used in several recent categorization systems [9, 12, 18]. These 2D shape models use silhouette contour fragments as well as inner contour fragments to model a category. However, such 2D shape models are sensitive regarding to pose changes. Consequently, several models for various aspects are needed [12], a drawback that could be eliminated by having just one 3D shape model per category.

But modeling of shape is not only required in computer vision and object categorization. Geometric modeling of 3D shape plays an important role in several research areas in vision and graphics and forms the basis for many applications. Methods to generate 3D models allow us to model the 3D nature of an object and this can provide additional information about shape and appearance of objects.

Point clouds, generated by laser range scanners, by stereo vision, or by Structure-from-Motion techniques, are probably the most obvious and simplest way to represent 3D shape. Often these point clouds are converted to triangle meshes or polygonal models. Extensive research has been done to generate, analyze, match, and classify such models. The shape representations vary from shape distributions [5, 10, 11, 13] to symmetry descriptors [7] or Skeletal Graphs [19]. Koertgen *et al.* [8] describe a similarity measure between 3D models based on 2D shape context which is similar to our idea. Iyer *et al.* [6] and Tangelder and Veltkamp [20] discuss several shape representation methods. Moreover, there exist 3D model databases such as the Princeton Shape Benchmark [17] and the ISDB [5].

So far, only few methods in 3D curve reconstruction exist. One method based on the usage of a double stereo rig was presented by Ebrahimnezhad and Ghassemian [3]. The authors describe a method for 3D reconstruction of object curves from different views and motion estimation based on these 3D curves. Park and Han [14] propose a method for Euclidean contour reconstruction including self-calibration. Unfortunately, their matching algorithm is not applicable to our data. Experiments using their matching algorithm with our data show that the resulting point correspondences are too inaccurate and thus many outliers exist. In contrast to our method their contour point correspondence algorithm is mainly based on epipolar information and local descriptors. Recently, Fabbri and Kimia [4] presented an approach for multi-view stereo reconstruction and calibration of curves. They concentrate on the reconstruction of contour fragments without motion analysis and their algorithm is mainly based on so called view-stationary curves, *e.g.* shadows, sharp ridges, reflectance curves. Therefore, it is well applicable for aerial images as they show in their results.

Motivated by these aspects, we aim to extend 2D contour fragment models towards 3D shape models for objects using 3D contour fragments. Our '3D Contour Clouds' have potential applications in various research areas including contour-based 3D shape retrieval, matching and categorization.

The main contribution of this paper is the automatic stereo reconstruction of '3D Contour Clouds' for individual objects from stereo image sequences. Section 2 describes in detail our '3D Contour Cloud' reconstruction method and introduces the idea of 3D shape context for 3D contour fragment matching, which we use for outlier reduction. In Section 3, we describe our own dataset containing calibrated stereo image sequences of humans and hand-held objects. We show reconstruction results on this dataset as well as on a standard multi-view stereo dataset.

2. '3D Contour Cloud' stereo reconstruction

The determination of long and salient 3D contour fragments is a rather difficult task. One main task in reconstructing 3D contour fragments is to find accurate point correspondences on 2D contour fragments of the left and the right stereo frame. Two problems may occur:

- Linking: Different linking of edges to longer contour fragments in different views (stereo frame pairs as well as consecutive frames) influences a matching procedure. Contour fragments may not have the same length, same start point, and the same end point.

- Shape deformation: The shape of contours changes significantly when viewing them from different poses, which makes it harder to track contours over time and to find stereo correspondences (visual rim changes).

A standard stereo correspondence algorithm would compute the intersection between epipolar line and contour fragments and search in a neighborhood of these intersection points for the corresponding point. In contrast to this method, our new approach integrates the well known 2D shape context with epipolar information in one single cost matrix.

Nevertheless, 2D contour fragments with similar shape may produce false correspondences of 2D contour fragments and contour points which result in incorrectly reconstructed 3D contour fragments. For outlier reduction we introduce 3D shape context as an extension of 2D shape context.

2.1. Stereo reconstruction of 3D contour fragments

2D shape context is very suitable for our contour fragment matching, but using shape context in combination with epipolar geometry has many more advantages. First, we can limit our search space to a subset of contours by only taking into account those contour fragments which lie in regions restricted by the epipolar lines. Second, we rely on point correspondences which are more precise than using just one of the methods. We apply the Canny edge detection algorithm (subpixel accuracy) to extract contour fragments in the left and the right frame of a stereo frame pair. Then, a linking algorithm based on smoothing constraints (Leordeanu et al. [9]) is used to obtain long, connected 2D contour fragments. For the point correspondences we generate a new cost matrix where we build a weighted combination of the shape context cost matrix and a cost matrix computed from epipolar information.

Let $\mathbf{p}_i$ be a point on a contour fragment in the left stereo image and $\mathbf{q}_j$ be a point on a contour fragment in the right stereo image. Then, the original cost matrix (see [1]) is given by

$$\mathbf{CS}_{ij} := \frac{1}{2}\sum_{k=1}^{K}\frac{[h_i(k)-h_j(k)]^2}{h_i(k)+h_j(k)}, \qquad (1)$$

where $h_i(k)$ and $h_j(k)$ are the normalized histograms (K[1] bins) at points $\mathbf{p}_i$ and $\mathbf{q}_j$. In shape context, each point $\mathbf{p}_i$ on a contour fragment is represented by a histogram which contains the relative position to all other points on the contour fragment. The cost matrix $\mathbf{CS}$ is simply the χ^2 measure between the point-histograms of two contour fragments. Hence, the cost matrix $\mathbf{CS}$ contains for each point $\mathbf{p}_i$ on one fragment the matching cost to each point $\mathbf{q}_j$ on the second contour fragment. For our application for

[1]In our experiments $K = 12 * 5$, where 12 is the number of bins for θ and 5 is the number of bins for r (cmp. [1])

stereo correspondences we create an additional cost matrix which contains for each epipolar line $\mathbf{e}_j$ of one point $\mathbf{p}_i$ of the left contour fragment the distance to each point $\mathbf{q}_j$ on the right contour fragment and vice versa. Consequently,

$$\mathbf{e}_j = \mathbf{F}\mathbf{p}_i \qquad \text{and} \qquad \mathbf{e}_i = \mathbf{F}^T\mathbf{q}_j, \tag{2}$$

where $\mathbf{F}$ is the Fundamental Matrix, $\mathbf{e}_j$ and $\mathbf{e}_i$ define the epipolar lines for points $\mathbf{p}_i$ and $\mathbf{q}_j$. This leads to the second cost matrix $\mathbf{CE}$, with

$$\mathbf{CE}_{ij} = \frac{(d(\mathbf{e}_j, \mathbf{q}_j) + d(\mathbf{e}_i, \mathbf{p}_i))^2}{\max((d(\mathbf{e}_j, \mathbf{q}_j) + d(\mathbf{e}_i, \mathbf{p}_i))^2)}, \tag{3}$$

where d is the Euclidean distance between the epipolar line and the point. The maximum $max((d(\mathbf{e}_j, \mathbf{q}_j) + d(\mathbf{e}_i, \mathbf{p}_i))^2)$ is the maximum over all entries of the matrix and denotes a normalization factor. We combine both cost matrices by

$$\mathbf{C}_{ij} = w_1 * \mathbf{CS}_{ij} + w_2 * \mathbf{CE}_{ij}, \tag{4}$$

where w_1 and w_2 are weighting factors. Empirically, we found $w_1 = 0.3$ and $w_2 = 0.7$ to be good choices for the weighting factors.

For a continuous reconstruction, we additionally use an ordering constraint on the contour fragments, because neighboring points should have neighboring correspondence points. By first achieving a clockwise ordering of the contour points, the corresponding points then are given by a sub-diagonal of the cost matrix. Based on these contour point correspondences we reconstruct the 3D contour fragments using the 'Object Space Error for General Camera Models' [15].

2.2. 3D Shape Context

Outliers - falsely reconstructed 3D contour fragments - may always occur due to false stereo correspondences or false matching over time. It is not always possible to detect falsely reconstructed 3D contour fragments only on the basis of 2D information. Therefore, we introduce a 3D shape representation and matching based on the idea of extending 2D shape context - we call it 3D shape context - to reduce the number of outliers in a '3D Contour Cloud'. A '3D Contour Cloud' CC consists of a number of 3D contour fragments

$$CC = \{F_l, l = 1...N\}, \tag{5}$$

where N is the number of 3D contour fragments in a cloud. Each of these 3D contour fragments has been seen in several frames and tracked over time, so that each 3D contour fragment F_l consists of a number of reconstructed fragments

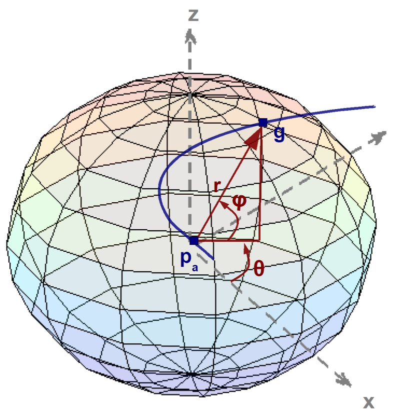


Figure 1. 3D shape context description: For each point on a 3D contour fragment (blue curve) the relative position (r, ϕ, θ) to all other points is computed.

$$F_l = \{f_i^l, i = 1...M\}, \tag{6}$$

where M is the number of frames in which F_l is tracked. We then use 3D shape context to verify those fragments f_i^l which have the most similar shape. Fragments which do not have a similar shape as the majority of F_l are rejected as outliers. The cost matrix on the fragment F_l is defined by

$$\mathbf{CF}_{ij} = sc\_cost\_3D(f_i^l, f_j^l) \qquad \forall i, j \in M, \tag{7}$$

where $sc\_cost\_3D(f_i^l, f_j^l)$ is the 3D shape context matching cost between fragment f_i^l and f_j^l. By analyzing the median and variance of this cost matrix we can identify those tracked 3D contour fragments f_n^l which are not similar to the other fragments of F_l.

The 3D shape context matching cost $sc\_cost\_3D(f_i^l, f_j^l)$ is defined in a similar way as in 2D. Let $\mathbf{p_a}$ be a 3D contour point of fragment f_i^l. Similar to 2D shape context we build a multi-dimensional histogram $h1_a$ by computing the relative positions to all other contour points $\mathbf{g}$ on fragment f_i^l in a 3D log-polar space. Hence, we compute the distance r, the azimuth θ and elevation ϕ. The basis for this computation builds an object-centered coordinate system. Figure 1 illustrates the principle.

Then, our 3D shape context is defined in the following way

$$h1_a(k) := \#\left\{\mathbf{g} \neq \mathbf{p}_a : (r, \theta, \phi) \in bin(k)\right\}. \tag{8}$$

In our experiments the number of bins $K = 12 * 12 * 5$ where 12 is the number of bins for θ and ϕ, and 5 defines the number of bins for r. Here, the cost matrix $\mathbf{CS3D}$ between to fragments is defined in the same manner as in the 2D case using the χ^2 measure

$$\mathbf{CS3D}_{ab} := \frac{1}{2}\sum_{k=1}^{K} \frac{[h1_a(k) - h1_b(k)]^2}{h1_a(k) + h1_b(k)}, \quad (9)$$

for points $\mathbf{p_a}$ and $\mathbf{q_b}$. Taking into account also the position on the object, we compute a second, very sparse histogram $h2_a$. To obtain the position of the contour points to a reference point, we compute the distance r, the azimuth θ and elevation ϕ relative to the defined reference point $\mathbf{oc}$ *e.g.* object center:

$$h2_a(k) := \#\left\{\mathbf{oc} \neq \mathbf{p}_a : (r, \theta, \phi) \in bin(k)\right\}. \quad (10)$$

This histogram is sparse in the sense, that for each point p_a we have just one entry - the bin which contains (r, θ, ϕ) for p_a. The cost matrix is given by

$$\mathbf{CP3D}_{ab} := \frac{1}{2}\sum_{k=1}^{K} \frac{[h2_a(k) - h2_b(k)]^2}{h2_a(k) + h2_b(k)}. \quad (11)$$

The overall cost matrix is then given by a weighted sum of both cost matrices:

$$\mathbf{C3D}_{ab} = w_1 * \mathbf{CS3D}_{ab} + w_2 * \mathbf{CP3D}_{ab}, \quad (12)$$

where we found empirically $w_1 = 0.6$ and $w_2 = 0.4$. The matching cost *sc_cost_3D* between two 3D contour fragments f_i^l and f_j^l is then computed by

$$sc\_cost\_3D(f_i^l, f_j^l) = max(\tfrac{1}{A}\sum_{a=1}^{A} \min_b \mathbf{C3D}_{ab}, \frac{1}{B}\sum_{i=1}^{B} \min_a \mathbf{C3D}_{ab}) \quad (13)$$

where A (B) is the number of contour points on fragment f_i^l (f_j^l). Contour fragments with a matching cost greater than a threshold wrt. to the majority of all other fragments are detected as outliers. Here, the threshold is defined by computing the median and variance of the cost matrix.

3. Experiments and Results

We evaluate our '3D Contour Cloud' reconstruction approach on our own dataset consisting of two types of videos and a standard multi-view dataset.

3.1. Datasets

Our own video datasets, Graz-Stereo-Base-Eye and Graz-Stereo-Base-30 [2] are taken by a calibrated stereo rig. However, we wish to point out that the underlying algorithmic components (structure and motion analysis) are well-suited to capture 3D object models from other, more complex video data in the future. Therefore, we also show experiments on a standard multi-view stereo dataset.

3.1.1 Graz-Stereo-Base-Eye

We captured in the lab several videos of small toy objects which are manipulated naturally by hand in front of a stereo camera system (see Figure 2). Each of these stereo videos typically contains an object, which is presented in a hand-held manner in front of homogeneous background. So, we avoid the controlled setting of turntables. The object is manipulated such that it is seen from all sides, showing all aspects. In order to reconstruct just contours of the hand-held objects and not contours of the hand, we first have to mask the hand in the stereo videos of hand-held objects. Here, we use a segmentation system based on variational methods (see [21]), which gives us a precise hand segmentation. Features that belong to the hand are subsequently ignored. The motion analysis is based on the approach by Schweighofer et al. [15]. The system is able to reconstruct Structure and Motion of stationary scenes and it is robust if there are at least 50% of the features in the stationary scene, foreground motion is detected as outliers. In our case, because the majority of the interest points is located on a rigid object, the system assumes that the stereo rig is moving around the object although we manipulate the object in front of the cameras. This leads to an 'object-centered' representation as shown in Figure 6.

3.1.2 Graz-Stereo-Base-30

We capture several videos of humans moving in front of a stereo camera system (see Figure 3). People rotate around their vertical axis in front of homogeneous background and show several aspects to the stereo rig. Again, the motion estimation is done using the approach by [15].

[2]The video data are available for download from `http:\\www.emt.tugraz.at/~pinz/data`

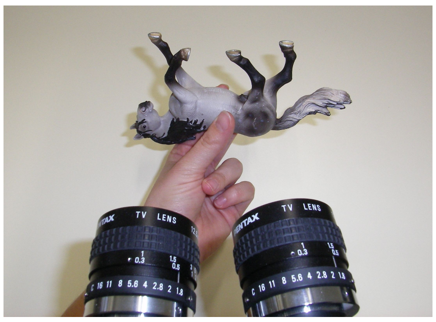


Figure 2. Stereo rig to capture the Graz-Stereo-Base-Eye database. We use two μEye 1220C cameras and Cosmicar/Pentax lenses with a focal length of 12.5 mm. The baseline is approximately 6 cm (human eye distance), the vergence angle 5.5^o. The frame rate is 15 Hz. The size of the images is 480x752 px. For the calibration of the stereo rig we use the Camera Calibration Toolbox for Matlab [2]

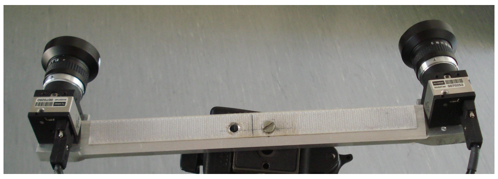


Figure 3. Stereo rig to capture the Graz-Stereo-Base-30 database. Focal length: 6.5 mm, Baseline: 30 cm, vergence angle: 6.5^o.

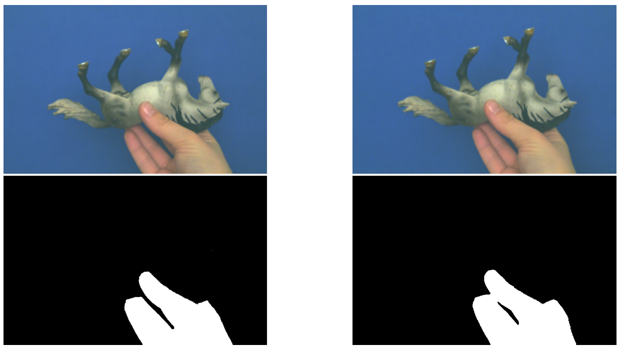


Figure 4. One example stereo frame pair of the database Stereo-Graz-Stereo-10-Base-Eye. The hand is masked using the method by [21]

3.1.3 Multi-view stereo dataset

The multi-view dataset of [16] consists of two image sequences (Dino-dataset and Temple-dataset). The datasets were generated by sampling several views on a hemisphere around the objects. The camera calibration parameters and camera poses are available.

3.2. 3D Contour Clouds of Graz-Stereo-Base-Eye

The Graz-Stereo-Base-Eye consists of stereo videos of small hand-held objects which are manipulated in front of the stereo rig (see Figure 4).

First, we want to demonstrate the '3D Contour Cloud' reconstruction method on one example stereo video of our dataset. The stereo video we choose shows a small toy horse with the dimensions $length = 175$mm, $height = 95$mm, and $width = 45$mm (see Figure 4). Figure 5 shows the output for the stereo reconstruction process - 3D contour fragments for single stereo frame pairs. We can see how the 3D shape changes over time when rotating the object.

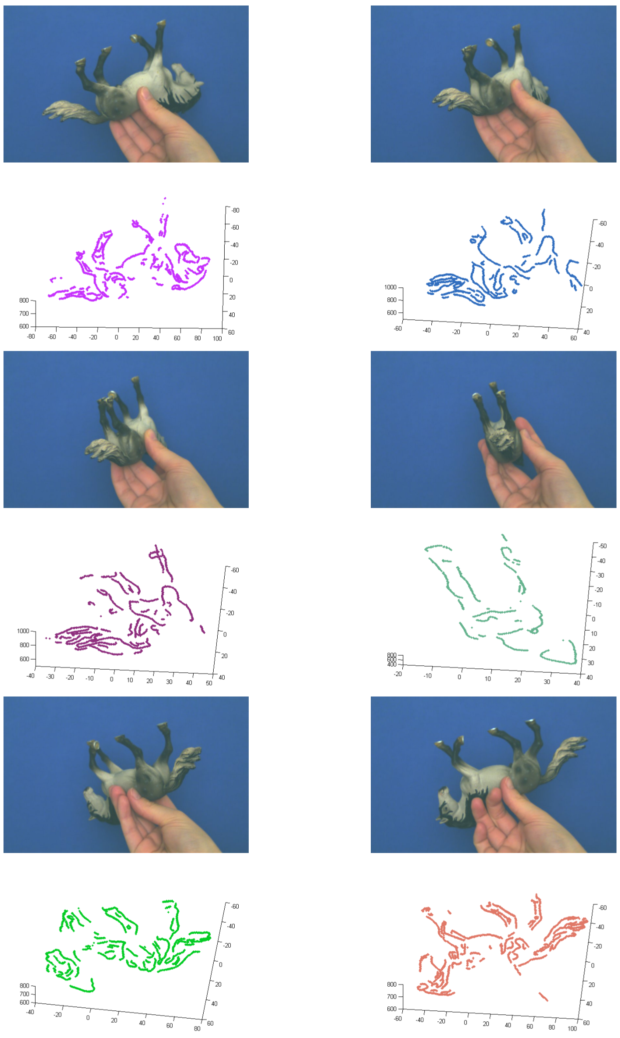


Figure 5. 3D contour fragments per stereo frame pair for a small subset of stereo frame pairs over 180^o. For space-saving only the left frames of stereo pairs are shown, above their corresponding reconstructions.

Figure 6 shows the estimated camera poses of the stereo rig around the object in an object-centered coordinate system. We can see that we manipulate the object by first rotating it for approximately 360^o around the vertical axis of the horse and then by about 180^o around the horizontal axis. Figure 7 shows a '3D Contour Cloud' of the horse for 201 frames without reducing the outliers, Figure 8 shows

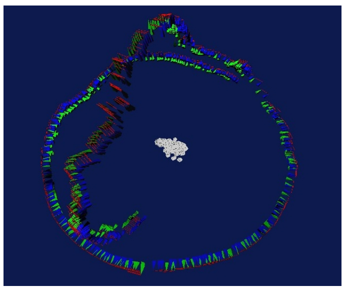


Figure 6. Camera poses of the stereo rig (green and blue triangle connected by a red line) estimated around the horse using [15] represented in an object-centered coordinate system.

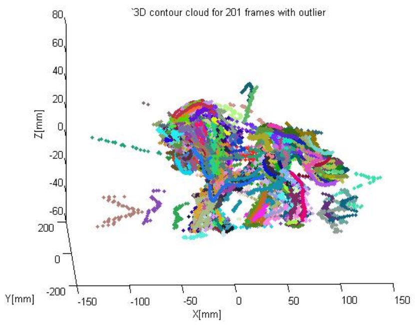


Figure 7. '3D Contour Cloud' of a horse for 201 frames. In this reconstruction outliers have not been removed using 3D shape context and all 3D contour fragments are drawn.

the same '3D Contour Cloud' by reducing the outliers and only drawing 3D contour fragments which were tracked for at least 3 frames. 201 frames correspond to a rotation of the object of approximately 270^o around the vertical axis of the object. In the reconstruction we choose every fifth frame. Instead of visualizing one representative contour fragment, we show all registered contours in corresponding color, which explains the width of the visualized contour fragments. We see that a high number of outliers can be reduced so that the shape of the horse is clearly visible. We see that there is no contour for the back of the horse, which is caused by the fact that in the first part of manipulating the horse, the back is occluded by the hand (comp. Figure 4).

Figure 9 shows a reconstructed '3D Contour Cloud' with an automatically generated 3D Bounding Box and object-centered coordinate system. We choose the longest dimension of the 3D Bounding Box as x-axis. For the purpose of visualization, we choose a '3D contour cloud' with a reduced number of 3D contours where we reconstruct the '3D contour cloud' just for a short subsequence of the stereo video.

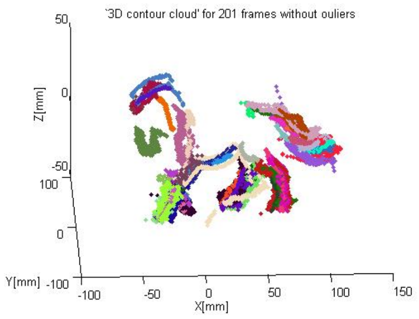


Figure 8. '3D Contour Cloud' of a horse for 201 frames. Outliers have been removed using 3D shape context and the median as threshold. Only those 3D fragments are visible which have been tracked over ≥ 3 frames.

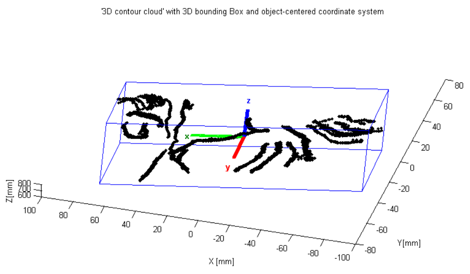


Figure 9. '3D contour cloud' with automatically generated 3D Bounding Box and object-centered coordinate system

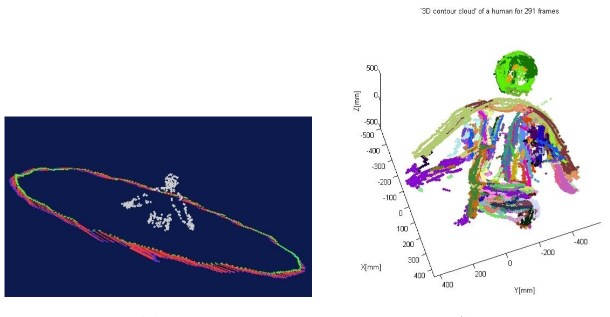
3.3. 3D Contour Clouds of Graz-Stereo-Base-30

The Graz-Stereo-Base-30 consists of calibrated stereo videos showing humans that rotate around their vertical axis (see Figure 10). Figure 11(a) shows the estimated camera poses of the stereo rig around the object in an object centered coordinate system. We can see that the human rotates approximately 360^o around his vertical axis. Figure 11(b) shows a '3D Contour Cloud' of the human. We can clearly identify the shape of the human. Again, instead of visualizing one representative contour fragment, we show all registered contours in corresponding color, which explains the width of the visualized contour fragments. We can see at the example of the green 3D contour fragment on the arm how the visual rim is tracked over some frames, when it is similar to other poses. Figure 12 shows the results for another human stereo video.

3.4. 3D Contour Clouds of the Multi-view stereo dataset

To show that our method is also applicable to standard datasets, we apply our reconstruction method to the multi-view dataset of [16], which consists of the Dino-dataset (see Figure 13) and the Temple-dataset

Figure 10. Example images for one stereo video of the database Graz-Stereo-Base-30. For space-saving only the left frame of a stereo pair is shown.



(a) (b)

Figure 11. Camera poses of the stereo rig (green and blue triangle connected by a red line) estimated around the human using [15] represented in an object-centered coordinate system.

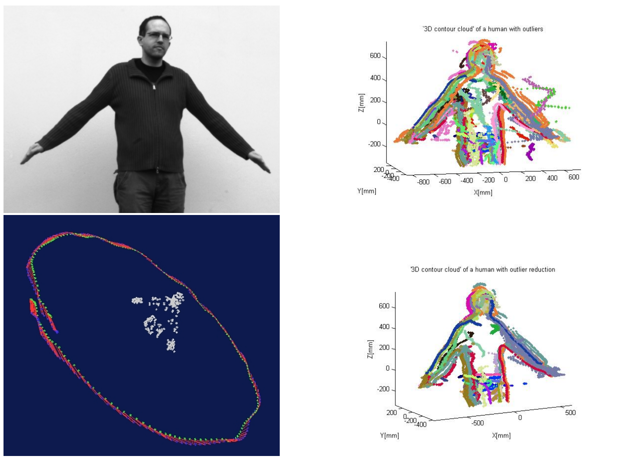


Figure 12. Stereo reconstruction of a human: left frame; estimated camera poses; '3D Contour Cloud' with outliers (top); '3D Contour Cloud' without outliers and 3D contour fragments which are tracked over $\geq$ 2 frames (bottom)

(see Figure 14), including camera parameters and camera poses. For these datasets, views sampled on a hemisphere or on a ring are available.

Figure 15 shows a '3D Contour Cloud' for a subset of images of the Dino-dataset, and Figure 16 for the Temple-dataset. For the visualization we choose a small subsample of images captured from the same aspect (side view of the dino and the temple). Similar to the other datasets, we show all registered contours in corresponding color, which explains the width of the visualized contour fragments. We can see that the plates of the stegosaurus result in many different 3D contour fragments because of different illumination effects, and that the striation on the pillars delivers many non-distinguishable inner contour fragments.

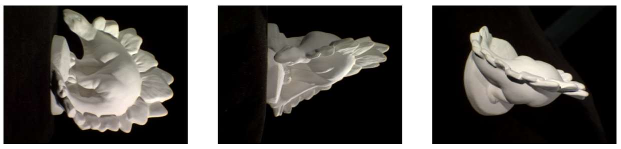


Figure 13. Sample images of the Dino-dataset. 363 views are sampled on a hemisphere.

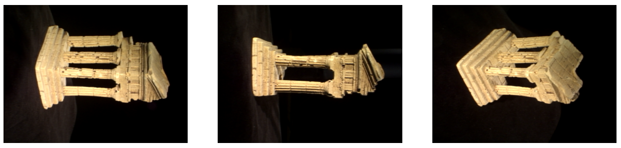


Figure 14. Sample images of the Temple-dataset. 312 views are sampled on a hemisphere.

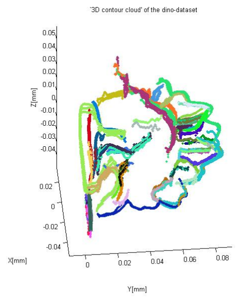


Figure 15. '3D Contour Cloud' of the dino.

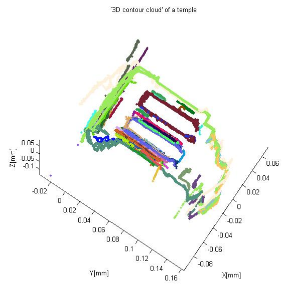


Figure 16. '3D Contour Cloud' of the temple.

4. Conclusion and Outlook

We have presented an automatic reconstruction method that can generate '3D Contour Clouds' for several objects and various databases. Furthermore, the system generates 3D bounding boxes and object-centered coordinate systems. Although most of our experimental results are obtained on our own stereo datasets, we show that the approach is also applicable to a standard multi-view stereo dataset. Moreover, the paper presents an extension of standard 2D shape context towards 3D, a contribution that will be useful in various other 3D shape matching applications.

Various applications may benefit from the '3D Contour Clouds' presented in this paper, but our main interest is in object categorization using pose-invariant 3D shape models. Here, the focus does not lie in a precise reconstruction, rather in a qualitatively convincing representation of 3D shape of a category by salient contour fragments. We are aware that an exact silhouette reconstruction is not possible based on two views where different silhouette contours may be seen on the 'visual rim' of the object. On the other hand, there is the advantage that all aspects of the object's 'visual rim' are integrated into one single 3D model.

The representation by '3D Contour Clouds' constitutes a powerful method for geometric modeling of 3D shape. In addition, 3D contour fragments are more discriminative than standard point cloud representations.

Acknowledgment

This work was supported by the Austrian Science Fund (FWF) under the doctoral program Confluence of Vision and Graphics W1209. We thank Markus Unger for help with segmentation [21].

References

[1] S. Belongie, J. Malik, and J. Puzicha. Shape Matching and Object Recognition Using Shape Contexts. *IEEE Trans. on PAMI*, 24(4):509–522, 2002. 2

[2] J.-Y. Bouguet. Camera calibration toolbox for matlab. 5

[3] H. Ebrahimnezhad and H. Ghassemian. Robust motion and 3D structure from space curves. In *Proc. Intern. Conf. on Information Sciences, Signal Processing and their Applications*, 2007. 1

[4] R. Fabbri and B. B. Kimia. 3D curve sketch: Flexible curve-based stereo reconstruction and calibration. In *Proc. CVPR*, 2010. 2

[5] R. Gal, A. Shamir, and D. Cohen-Or. Pose-Oblivious Shape Signature. *IEEE TVCG*, 13(2):261–271, 2007. 1

[6] N. Iyer, S. Jayanti, K. Lou, Y. Kalyanaraman, and K. Ramani. Three-dimensional shape searching: state-of-the-art review and future trends. *Computer-Aided Design*, 37(5):509–530, 2005. 1

[7] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Symmetry Descriptors and 3D Shape Matching. In *Symposium on Geometry Processing*, 2004. 1

[8] M. Körtgen, G. J. Park, M. Novotni, and R. Klein. 3D Shape Matching with 3D Shape Contexts. In *Proc. Central European Seminar on Computer Graphics*, 2003. 1

[9] M. Leordeanu, M. Hebert, and R. Sukthankar. Beyond Local Appearance: Category Recognition from Pairwise Interactions of Simple Features. In *Proc. CVPR*, 2007. 1, 2

[10] M. Mahmoudi and G. Sapiro. Three-dimensional point cloud recognition via distributions of geometric distances. *Graph. Models*, 71(1):22–31, 2009. 1

[11] R. Ohbuchi, T. Minamitani, and T. Takei. Shape-similarity search of 3D models by using enhanced shape functions. *International Journal of Computer Applications in Technology (IJCAT)*, 2005:70–85. 1

[12] A. Opelt, A. Pinz, and A. Zisserman. A Boundary-Fragment-Model for Object Detection. In *Proc. ECCV*, 2006. 1

[13] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. *ACM Trans. Graph.* 1

[14] J. S. Park and J. H. Han. Euclidean reconstruction from contour matches. *PR*, 35:2109–2124, 2002. 1

[15] G. Schweighofer, S. Segvic, and A. Pinz. Online/realtime structure and motion for general camera models. In *Proc. WACV*, 2008. 3, 4, 6, 7

[16] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms. In *Proc. CVPR*, 2006. 5, 6

[17] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser. The princeton shape benchmark. In *Proc. SMI*, 2004. 1

[18] J. Shotton, A. Blake, and R. Cipolla. Multi-Scale Categorical Object Recognition Using Contour Fragments. *IEEE Trans. on PAMI*, 30(7):1270–1281, 2008. 1

[19] H. Sundar, D. Silver, N. Gagvani, and S. Dickinson. Skeleton Based Shape Matching and Retrieval. In *Proc. SMI*, 2003. 1

[20] J. Tangelder and R. Veltkamp. A survey of content based 3D shape retrieval methods. *Multimedia Tools and Applications*, 2007. 1

[21] M. Unger, T. Mauthner, T. Pock, and H. Bischof. Tracking as Segmentation of Spatial-Temporal Volumes by Anisotropic Weighted TV. In *Proc. EMM-CVPR*, volume 5681, 2009. 4, 5, 8

16th *Computer Vision Winter Workshop*
Andreas Wendel, Sabine Sternig, Martin Godec (eds.)
Mitterberg, Austria, February 2-4, 2011

Detection of unseen patches trackable by linear predictors

David Hurych
hurycd1@cmp.felk.cvut.cz

Karel Zimmermann
zimmerk@cmp.felk.cvut.cz

Tomáš Svoboda
svoboda@cmp.felk.cvut.cz

Center for Machine Perception, Dept. of Cybernetics
Czech Technical University in Prague

Abstract.

Linear predictors (LPs) are being used for tracking because of their computational efficiency which is better than steepest descent methods (e.g. Lucas-Kanade). The only disadvantage of LPs is the necessary learning phase which hinders the predictors applicability as a general patch tracker. We address this limitation and propose to learn a bank of LPs off-line and develop an on-line detector which selects image regions that could be tracked by some predictor from the bank. The proposed detector differs significantly from the usual solutions that attempt to find the closest match between a candidate patch and a database of exemplars. We construct the detector directly from the learned linear predictor. The detector positively detects the learned patches, but also many other image patches, which were not used in LP learning phase. This means, that the LP is able to track also previously unseen image patches, the appearances of which are often significantly diverse from the patches used for learning. We propose a fast LP-structure-based detection method, which is in computational cost comparable with standard appearance-based detectors and is easy to construct directly from a trained LP without any further learning.

1. Introduction

Many computer vision techniques (e.g. 3D reconstruction, simultaneous localization and mapping) and applications (e.g. surveillance, robot's visual navigation) require real-time, reliable and accurate tracking algorithms. The most natural approach to tracking is scanning around the last known position for the maximum response of a criterion function. Since an exhaustive scanning through the whole image is time consuming, the search space is often restricted. If it is known, that the last position is not too far from the current position, a local estimation methods, like steepest descend methods [10, 3] or regression-based methods [9] are often used. Unfortunately, locality of such methods is unavoidable. Each method has a limited range within which it works. The range is usually determined by the maximal inter-frame object displacement. A detector is essentially needed for any tracking application in order to resolve cases when the track is lost. For example the Lucas-Kanade tracker usually re-starts on features that are good to track [12].

Linear predictor is one of the simplest yet powerful regression-based tracking method. Linear predictor (LP) is a linear regression function which maps observed image intensities to motion parameters. It has been shown in [17] that LPs outperform steepest descend method in both the size of the basin-of-attraction and the speed of tracking. However, the off-line learning stage limits their practical usage as a general tracker in an open world environment. To avoid this drawback we propose to pre-train a database of LPs and equip each LP by a detector that finds trackable patches. A naive solution, which would check the neighbourhoods of all image points for LP convergence, would make the usage of LPs prohibitively time consuming. We propose a way how to build the LP-structure-based detector, which detects all LP trackable points, rejects most of the other points and preserves computational cost comparable with standard appearance-based detectors.

The rest of the paper is organized as follows: Section 1.1 introduces state-of-the-art in linear predictors, Section 2 explains the functionality and construction of the detector and finally Section 3 shows experimental evaluation of the method.

1.1. State-of-the-art

Given the initial position $\mathbf{s}_0 \in S$, where S is the set of all 2D coordinates in the current image $\mathbf{I}$, a tracker estimates motion[1] $\mathbf{t}$ of the object by some function $\varphi(\mathbf{I}, \mathbf{s}_0)$:

$$\mathbf{t} = \varphi(\mathbf{I}, \mathbf{s}_0). \qquad (1)$$

The most common way of tracking is repeated minimization of some image similarity (criterion) function $f(\mathbf{t}; \mathbf{I}, \mathbf{s}_0)$ given an image $\mathbf{I}$ and previous object position $\mathbf{s}_0$

$$\mathbf{t}^* = \operatorname{argmin} f(\mathbf{t}; \mathbf{I}, \mathbf{s}_0) = \varphi(\mathbf{I}, \mathbf{s}_0), \qquad (2)$$

where $\mathbf{t}^*$ is the estimate of the object's motion. Criterion $f(\mathbf{t}; \mathbf{I}, \mathbf{s}_0)$ includes implicit or explicit model of object's possible appearances and optionally some relation to $\mathbf{s}_0$. Criterion f could be for example obtained as a similarity function as well as a classifier or foreground/background probability ratio learned from training examples.

By *optimization-based tracking* we understand an on-line optimization technique solving problem (2). While some approaches [11, 2, 8, 4] exhaustively search for object in a subset of object positions S with a classifier approximating $f(\mathbf{t}; \mathbf{I}, \mathbf{s}_0)$, another approaches [10, 3, 14, 12] use a gradient optimization of some criterion.

Regression-based tracking methods attempt to model explicitly the relationship between image observations and the optimal motion $\mathbf{t}^*$ without the necessity of defining criterion $f(\mathbf{t}; \mathbf{I}, \mathbf{s}_0)$. They learn function $\varphi(\mathbf{I}, \mathbf{s}_0)$ (used in equation (1)) in a supervised way from synthesized training data [9, 5, 15]. We outline main principle of the regression-based methods.

The regression-based methods [5, 9, 15] estimate the motion $\mathbf{t}$ *directly* from locally observed intensities on some set of pixels $X \subset S$ called *support set* (pixels coordinates spread over the object) instead of optimizing the criterion function $f(\mathbf{t}; \mathbf{I}, \mathbf{s}_0)$. Such an approach requires a learning stage. Pairs of motions $\mathbf{t}$ and corresponding vector of image intensities $\mathbf{I}(\mathbf{t} \circ X)$, observed in coordinates X moved by vector $\mathbf{t}$, are collected and a mapping $\varphi\colon \mathbf{I} \to \mathbf{t}$ minimizing some error on these examples is estimated, see Figure 1,

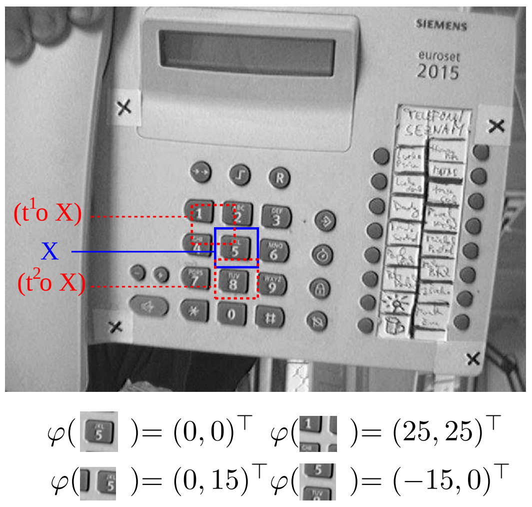
$$\varphi^* = \operatorname*{argmin}_{\varphi} \sum_{\mathbf{t}} \left\|\varphi\Big(\mathbf{I}(\mathbf{t} \circ X)\Big) - \mathbf{t}\right\|, \qquad (3)$$

In the tracking stage, the learned mapping φ^* directly estimates motion parameters without the necessity of an on-line optimization of any criterion function.

Notice that Lucas-Kanade tracker [10] solves a similar optimization task in each frame, where it needs to compute the image gradient, Jacobian of the warp and pseudo-inverse of the Hessian. This process can be replaced by using a regression matrix $\mathtt{H}$ learned on a set of synthesized examples. Matrix $\mathtt{H}$ forms a linear mapping between intensities $\mathbf{I}(\mathbf{t} \circ X)$ and motion $\mathbf{t}$,

$$\mathbf{t} = \varphi\Big(\mathbf{I}(\mathbf{t} \circ X)\Big) = \mathtt{H}\Big(\mathbf{I}(\mathbf{t} \circ X) - \mathbf{I}(X)\Big), \qquad (4)$$

In the tracking procedure, the motion parameters $\mathbf{t}$ are simply computed as a linear function $\mathtt{H}(\mathbf{I}(\mathbf{t} \circ X) - \mathbf{I}(X))$ of the object intensities. We call such method *Linear Predictor*. In the following, the least squares



$(t^1 o\ X)$
X
$(t^2 o\ X)$
SIEMENS
euroset 2015
$\varphi(\;) = (0,0)^\top$
$\varphi(\;) = (25,25)^\top$
$\varphi(\;) = (0,15)^\top$
$\varphi(\;) = (-15,0)^\top$

Figure 1. **Learning of a linear mapping** between image intensities and motion parameters. The synthetically created training examples (image patches) are collected in a close neighborhood of the object's position under known set of motions. A linear mapping φ between these image samples and motion parameters is than computed using the least squares method.

[1]For simplicity, we talk about *2D position* and *2D motion* but the method also generalizes to more complex transformations.

learning of the LP is described, since it is the most commonly used.

Let us suppose we are given template $\mathbf{J} = \mathbf{I}(X)$ and collected training pairs $(\mathbf{I}^i = \mathbf{I}(\mathbf{t}^i \circ X),\ \mathbf{t}^i)$ $(i = 1 \dots d)$ of observed vectors of intensities $\mathbf{I}^i$ and corresponding motion parameters $\mathbf{t}^i$, which aligns the object with the current frame, see Figure 1. Then the *training set* is an ordered pair $(\mathtt{I}, \mathtt{T})$, such that $\mathtt{I} = [\mathbf{I}^1 - \mathbf{J}, \mathbf{I}^2 - \mathbf{J}, \dots \mathbf{I}^d - \mathbf{J}]$ and $\mathtt{T} = [\mathbf{t}^1, \mathbf{t}^2, \dots \mathbf{t}^d]$. Given the training set, LP's coefficients minimizing the square of Euclidean error on the training set are computed as follows:

$$\mathtt{H}^* = \mathtt{T}\underbrace{\mathtt{I}^\top(\mathtt{I}\mathtt{I}^\top)^{-1}}_{\mathtt{I}^+} = \mathtt{T}\mathtt{I}^+. \qquad (5)$$

Since the regression method is very effective it is widely applied in tracking. In particular, Cootes et al. [5, 6] estimate the parameters of Active Appearance Model (AAM) - *i.e.* deformable model with the shape and appearance parameters projected into a lower dimensional space by the PCA. In [5] a linear predictor (4) learned by the least squares method (5) estimates all parameters of the AAM. Since the linearity holds only for a small range of parameters, the solution is iterated. Iterations are computed with the same matrix but the length of the optimization step is locally optimized.

This approach was later adapted by Jurie et al. [9] for tracking of rigid objects. Unlike Cootes et al. [5], Jurie's linear predictors estimate local 2D translations only. The global motion is estimated from local motions by the RANSAC algorithm, showing the method to be very efficient and robust. Williams et al. [15] extended the approach to the non-linear motion predictors learned by Relevance Vector Machine [13] (RVM). Agarwal and Triggs [1] used RVM to learn the linear and non-linear mapping for tracking of 3D human poses from silhouettes. Another extension was suggested by Zimmermann et al. [17] who proposed an optimal way to concatenate several regression functions into a sequential predictor. Different learning techniques have also been proposed, e.g. Drucker et al. [7] search for the regression function that has at most certain deviation from the actually obtained poses. Zhou et al. [16] proposed greedy learning for additive regression functions, using weak regressor formed of a linear combination of binary functions.

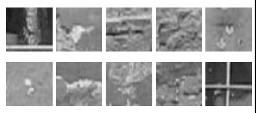
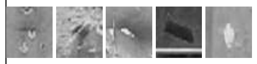
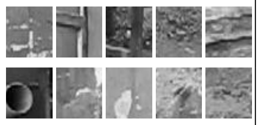
	All training patches	Trackable patches
LP1		
LP2		

Figure 2. **Examples of patches** used for learning (middle column) and some of *trackable patches* (right column) for 2 different LPs are shown for visual comparison. You may notice, that the trackable patches, which were found by our detector, are not visually similar to the patches used for training.

1.2. Contribution

We show that an LP allows to track many points it has not been trained for, for examples see Figure 2. Notice, that appearance of the set of LP trackable patches are very discrepant from the set of training patches. Such points could not be detected by any standard detector trained on the appearance of training examples – simply because the trackability rather stems from the structure of the LP than from the appearance of the training samples.

We propose an efficient detector of LP trackable points which (i) does not require any time consuming learning (ii) detects all trackable points and (iii) has computational costs comparable with standard appearance based detectors. The detector construction is described in the following section.

2. Detector of LP trackable points

Different forms of LPs provide different sensitivity to object appearance – the more degrees of freedom, the more general the LP is, but the longer learning is needed. In this paper, the two following forms of LPs are studied:

$$\textit{Basic LP:}\quad \mathbf{t} = \mathtt{H}\mathbf{I}(\mathbf{t} \circ X), \qquad (6)$$

$$\textit{Extended LP:}\quad \mathbf{t} = \mathtt{H}(\mathbf{I}(\mathbf{t} \circ X) - \mathbf{I}(X)). \qquad (7)$$

The *basic LP* uses directly the vector of observed image intensities $\mathbf{I}(\mathbf{t} \circ X)$ whereas the *extended LP* substracts the object template $\mathbf{I}(X)$ from $\mathbf{I}(\mathbf{t} \circ X)$. The extended version allows the tracker to be more variable, as will be seen in Section 3. Note, that although we speak about *Linear* predictors (because of histori-

(a) Convergence of the LP (b) Terminology

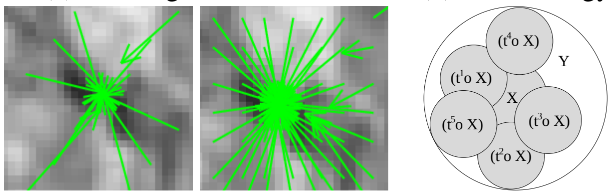


Figure 3. **(a) Convergence of the LP** from a neighbourhood of a single point (predictions are depicted by green arrows). Arrows point from the LP's initial position to the predicted object center. Left - depicts only a few of used votings for better visualization. Right - shows all the votings from the neighborhood R used in our algorithm. When all the arrows point to one pixel (or close enough), which is also the currently evaluated point, than the point is trackable. When the arrows point to some random directions, the point is not trackable by particular LP. **(b) Terminology:** X is the set of 2D coordinates, called support set. $(\mathbf{t}^j \circ X)$ is the support set transformed by local motion perturbation $\mathbf{t}^j$ and Y is union of all perturbations of the support set, i.e. $Y = \bigcup_{\mathbf{t}^j \in R}(\mathbf{t}^j \circ X)$.

cal reasons), huge class of possible non-linear extensions is at hand (e.g. polynomials can be treated as linear combinations of monomials).

The simplest way to detect the a trackable point would be to build a naive detector which evaluates LP's convergence (see Figure 3a) at every single point (rotation and/or scale) in the image. Let us consider that we want to check convergence of a simple LP with coefficients (regression matrix) $\mathtt{H}$ and support set X. Convergence of the LP means that each row $\mathbf{h}_i^\top$ of $\mathtt{H}$ and corresponding element t_i^j of every local perturbation $\mathbf{t}^j$ from the considered neighbourhood $R = \{\mathbf{t}^1, \mathbf{t}^2, \ldots, \mathbf{t}^n\}$ has to satisfy

$$\forall j \quad \mathbf{h}_i^\top \mathbf{I}(\mathbf{t}^j \circ X) = t_i^j + \mathbf{h}_i^\top \mathbf{I}(X) + \Delta^j \qquad (8)$$

for reasonably small prediction errors Δ^j, where $\mathbf{I}(\mathbf{t}^j \circ X)$ is a vector of intensities collected in the support set X transformed by local perturbation $\mathbf{t}^j$. For the sake of simplicity, row index i is further omitted.

This approach is, however, reasonably applicable just for LPs in the basic form and it can easily become prohibitively time consuming. Therefore, we propose sufficiently fast detection method of LP trackable points, applicable for more general forms of LPs, e.g. Equation (7) or other [18].

Main idea is based on the fact that checking the convergence of an LP on some region is almost equivalent to checking whether the mean and variance of the prediction errors Δ^j are close to zero. We first introduce the set of all pixels

$$Y = \bigcup_{\mathbf{t}^j \in R} (\mathbf{t}^j \circ X), \qquad (9)$$

used in linear system (8), see Figure 3b. Then the Equation (8) is rewritten to the intensity independent form as follows:

$$\begin{aligned} \forall j \quad \mathbf{h}^\top \mathbf{I}(\mathbf{t}^j \circ X) &= \mathbf{f}^{j\top} \mathbf{I}(Y) \\ &= t^j + \mathbf{h}^\top \mathbf{I}(X) + \Delta^j, \end{aligned} \qquad (10)$$

where $\mathbf{f}^j$ consists of suitable permutations of elements of $\mathbf{h}^\top$ and zeros. Since $\mathbf{I}(Y)$ contains all elements of $\mathbf{I}(X)$ we can express prediction error in the following form

$$\begin{aligned} \Delta^j &= [\mathbf{f}^{j\top} \mathbf{I}(Y)] - [t^j + \mathbf{h}^\top \mathbf{I}(X)] \\ &= \mathbf{w}^{j\top} \mathbf{I}(Y) - t^j. \end{aligned} \qquad (11)$$

If the point is *trackable* then the prediction errors Δ^j has distribution $\mathcal{F}(\mu, \sigma)$ with both mean $\mu(\Delta)$ and variance $\sigma^2(\Delta)$ *close to zero* which is further denoted as $\mu(\Delta), \sigma^2(\Delta) \approx 0$. If $\mu(\Delta)$ or $\sigma^2(\Delta)$ gets far from zero, then there exist local perturbation(s) around the selected point, which cannot be compensated by the LP. It means, that to reject the hypotheses that the point is trackable by a given LP, it is not necessary to check all the linear equations in system (10), but we can easily check *necessary but not sufficient condition*:

$$\begin{aligned} \mu &= \frac{1}{n} \sum_j \Delta^j \\ &= \underbrace{\left(\frac{1}{n} \sum_j \mathbf{w}^{j\top}\right)}_{\mathbf{w}^\top} \mathbf{I}(Y) - \underbrace{\left(\frac{1}{n} \sum_j t^j\right)}_{b} \\ &= \mathbf{w}^\top \mathbf{I}(Y) + b \approx 0, \end{aligned} \qquad (12)$$

the computational cost of which is the same as the computational cost of just one equation of the linear system (10). This condition will be insufficient for example in the case, where all LP predictions has got flipped signs or in the case where $\mathbf{I}(Y)$ is constant (i.e. gradient is zero). While the first case is very rare, the second case is quite often. Image areas with constant (or almost constant) intensity function are inherently not trackable and can be eliminated in advance.

Similarly the variance $\sigma^2(\Delta)$ can be expressed as the following sparse quadratic form:

$$\begin{aligned}
\sigma^2 &= \mathcal{E}((\Delta^j)^2) - \mathcal{E}(\Delta^j)^2 \\
&= \frac{1}{n}\sum_j \left(\mathbf{w}_j^\top \mathbf{I}(Y) - t^j\right)^2 - \mu^2 \\
&= \mathbf{I}^\top(Y) \underbrace{\left(\frac{1}{n}\sum_j \mathbf{w}^j \mathbf{w}^{j\top}\right)}_{\mathtt{A}} \mathbf{I}(Y) - \\
&\quad - \underbrace{\left(\frac{2}{n}\sum_j \mathbf{w}^{j\top} t^j\right)}_{\mathbf{b}^\top} \mathbf{I}(Y) + \\
&\quad + \underbrace{\left(\frac{1}{n}\sum_j (t^j)^2\right) - \mu^2}_{c} \\
&= \mathbf{I}(Y)^\top \mathtt{A}\mathbf{I}(Y) - \mathtt{b}^\top \mathbf{I}(Y) + c \approx 0, \quad (13)
\end{aligned}$$

where $\mathtt{A}$ is a sparse, positive-semidefinite and symmetrical matrix, $\mathbf{b}^\top$ is a vector and c is a scalar. Sparsity of $\mathtt{A}$, which is crucial for the computational efficiency, depends on the set of considered local perturbations R. We observed that around 80% of its elements are equal to zero. All the coefficients needed for μ-test, i.e. Equation (12), and σ-test, i.e. Equation (13), are off-line directly created from the elements of $\mathtt{H}$, the on-line detection means only a few hundreds of scalar multiplications per point.

The point trackability is determined by the resulting μ and σ, which should not be bigger than corresponding threshold values Θ_μ and Θ_σ. If none of the values (μ or σ respectively) is bigger than threshold (Θ_μ or Θ_σ respectively), than the point is trackable by a particular LP, othervise it is not trackable. Note, that in our experiments 99.8% of not trackable points were already filtered by the μ-condition and the σ-condition was usually evaluated on only a few points. Note, that the same detector can be constructed for the basic LP by omitting term $\mathbf{h}^\top\mathbf{I}(X)$ in Equation (11).

3. Experiments

Experiments were performed on two video sequences, 1548 frames and 2595 frames long. Moving camera captured walls with windows, see Figure 4 for few example frames. Ground-truth inter-frames homographies were computed from manually labeled 4-point correspondences in both sequences, in order to be able to detect the loss-of-track of tested

Figure 4. **Example images** from tested sequences. The sequences contained mainly 2D translations with small scale changes and rotations. The tracked objects were mainly planar walls, windows or doors.

LPs. The LPs used in our experiments predicted 2D motion only. The tested sequences contain mainly 2D motion with small scale changes and small in-plane rotations to see the robustness of tested LPs. We demonstrate the results of experiments by graphs of average track length L on horizontal axis for particular number of trackable patches N on vertical axis. Changing the thresholds $[\Theta_\mu, \Theta_\sigma]$ for detector changes the number and quality of detected and tracked points, which than generates various points in these $NL-$diagrams. We evaluate the results for one (Figs. 6 and 7) up to six LPs (Fig. 5) trained on different patches for various $[\Theta_\mu, \Theta_\sigma]$ thresholds.

Comparison of the basic and extended linear predictor: μ and σ conditions are in practice evaluated with respect to some thresholds Θ_μ and Θ_σ. The higher thresholds the more points are accepted but the lower is the average length of the track because of the worse local convergence of the LP. Figure 5 shows the threshold combinations for basic LPs (in blue) and extended LPs (in red). The results show the average LP performance computed from six distinct LPs over 2 sequences. Lines connect combinations with fixed Θ_μ. The ideal LP with ideal detector would have all threshold pairs on the most right vertical line, since it would allow to track all the points for as long as possible. The pareto-optimal threshold combinations, which are emphasized by the thick line show the best performance, which may be obtained with particular set of patches for two tested sequences. Both predictor types have

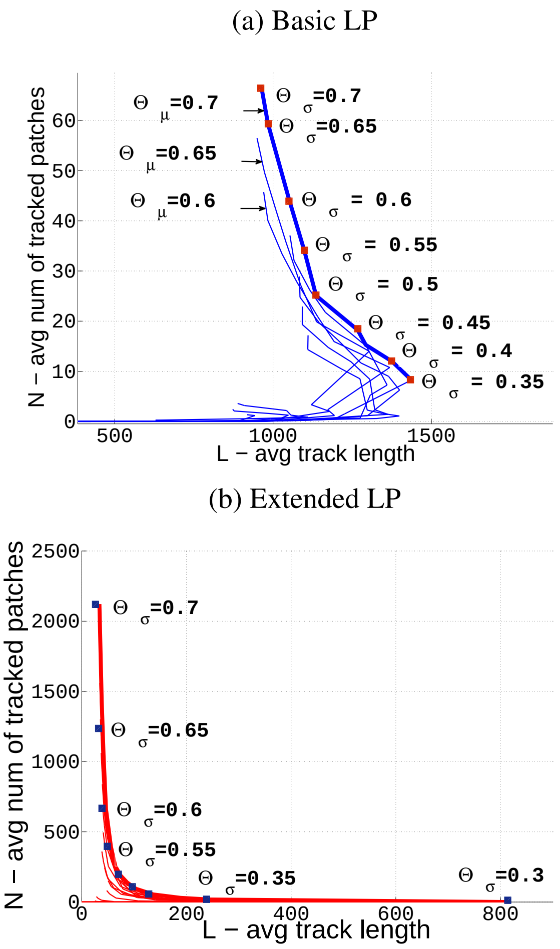


Figure 5. **Comparison of the (a) Basic LP and (b) Extended LP:** Lines connect threshold combinations with fixed Θ_μ, pareto-optimal threshold combinations are emphasized by the thick lines. Both predictor versions have different properties. The basic LP is able to track lower number of patches for more frames, while the extended LP is able to track a lot of patches, but looses the track more frequently.

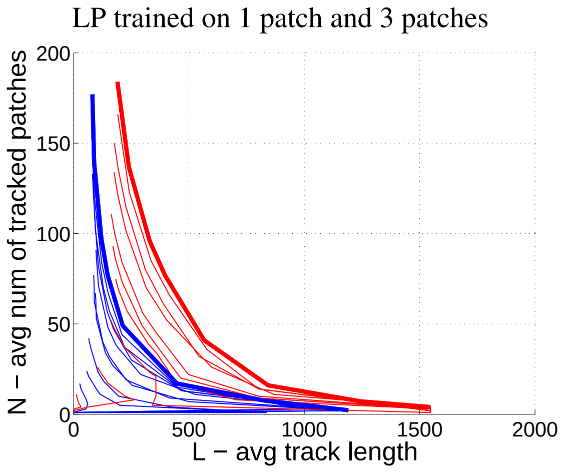


Figure 6. **Single versus multiple patch learning:** Comparison of the performance of single LP trained for one image patch (blue) and the same LP which was incrementally trained for another two patches. The aditional two training patches improved the predictor performance although the right number of training patches and the criteria for their selection is still not solved.

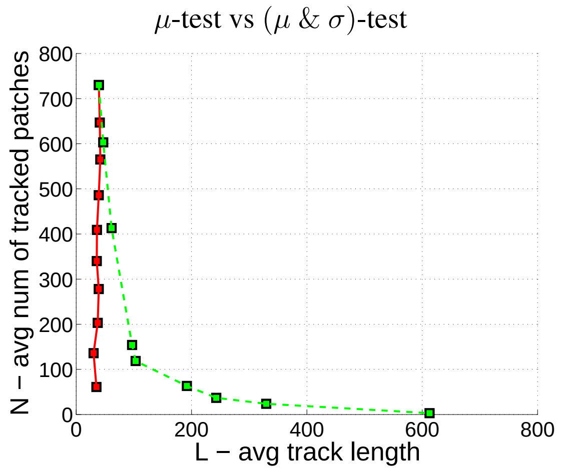


Figure 7. **Importance of σ-test:** NL-diagram of the points detected only by the μ-condition (in red) and points detected by both conditions with pareto-optimal thresholds (in green) for single LP.

different properties. The extended LP is alowes to track a high number of patches for a smaller number of frames (when averaged) than the basic LP, which tracks a lower number of patches for higher number of frames. The higher adaptibility of the extended LP on possible patch appearances, caused by the template $\mathbf{I}(X)$ substraction in equation 7. On the other hand the extended LP has zero mean on image regions with constant intensities (zero image gradient), which requires to compute the time consuming σ-test in more image points, than for the basic LP. So for the extended LP, it usually takes a little longer to detect the trackable points.

Using one or more patches to train the LP: On Figure 6 you may see the comparison of performance of LP trained for one patch (object) and the same LP, which was incrementally trained for another two patches. Adding a few training examples improves the LP's *generality*.

The numbers of patches for learning of the LPs used in our experiments vary from one to five. Some LPs (used in all experiments) were trained on patches, which appear in the tested sequences, but most of them were trained on completely different patches and objects, which do not appear in the tested sequences. The aditional training patches were selected manually and generally contained some distinctive visual features (e.g. harris corners or well textured areas). The automatic selection of training patches (and their number) for the most general LP is clearly an important issue and it will be the focus of our future work.

Importance of σ-test: In this experiment, we show the importance of the σ-test. We compare NL-diagram of the points detected only by the μ-condition (i.e., with $\Theta_\sigma = \infty$) and points detected by both conditions with pareto-optimal thresholds, showing that σ-test yields important improvement. Results are summarized in Figure 7.

4. Conclusion

We design a simple method for detection of LP trackable points. Depending on the μ and σ thresholding, one linear predictor may successfully track around 100 of different patches, for which the LP was not trained . The detector is very efficient and does not slow down the learing phase, because it is constructed directly from the learned LP. An interesting opened question is how to select the proper patches in the off-line training phase. For the moment we select examples by hand. In the future work we would like to optimize over wider range of possible patches.

We believe also that presented method can be extended to more general forms of LPs notably for the sequential LPs [17] or appearance parameterized LPs [18].

Acknowledgements

The 1st author was supported by the Czech Science Foundation Project P103/10/1585, and the 2nd author by the project P103/11/P700. The 3rd author was supported by EC project FP7-ICT-247870 NIFTi. Any opinions expressed in this paper do not necessarily reflect the views of the European Community. The Community is not liable for any use that may be made of the information contained herein.

References

[1] A. Agarwal and B. Triggs. Recovering 3d human pose from monocular images. *IEEE Transaction Pattern Analysis Machine Intelligence*, 28(1):44–58, 2006. 3

[2] S. Avidan. Support vector tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(8):1064–1072, 2004. 2

[3] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004. 1, 2

[4] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *9th European Conference on Computer Vision*, Graz Austria, May 2006. 2

[5] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. In *5th European Conference on Computer Vision-Volume II*, pages 484–498, London, UK, 1998. Springer-Verlag. 2, 3

[6] D. Cristinacce and T. Cootes. Feature detection and tracking with constrained local models. In 17^{th} *British Machine Vision Conference, Edinburgh, England*, pages 929–938, 2006. 3

[7] H. Drucker, C. Burges, L. Kaufman, A. Smola, and V. Vapnik. Support vector regression machines. *Advances in Neural Information Processing Systems*, 9:156–161, 1996. 3

[8] H. Grabner and H. Bischof. On-line boosting and vision. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 260–267, Washington, DC, USA, 2006. IEEE Computer Society. 2

[9] F. Jurie and M. Dhome. Real time robust template matching. In *British Machine Vision Conference*, pages 123–131, 2002. 1, 2, 3

[10] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International Journal of Computer Vision and Artificial Intelligence*, pages 674–679, 1981. 1, 2

[11] M. Özuysal, V. Lepetit, F. Fleuret, and P. Fua. Feature harvesting for tracking-by-detection. In *9th European Conference on Computer Vision*, volume 3953, pages 592–605, 2006. 2

[12] J. Shi and C. Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593 – 600, 1994. 1, 2

[13] M. E. Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001. 3

[14] L. Vacchetti, V. Lepetit, and P. Fua. Stable real-time 3D tracking using online and offline information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10):1385–1391, 2004. 2

[15] O. Williams, A. Blake, and R. Cipolla. Sparse bayesian learning for efficient visual tracking. *Pattern Analysis Machine Intelligence*, 27(8):1292–1304, 2005. 2, 3

[16] S. K. Zhou, B. Georgescu, X. S. Zhou, and D. Comaniciu. Image based regression using boosting method. In *International Conference on Computer Vision*, volume 1, pages 541–548, USA, 2005. 3

[17] K. Zimmermann, J. Matas, and T. Svoboda. Tracking by an optimal sequence of linear predictors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4):677–692, April 2009. 1, 3, 7

[18] K. Zimmermann, T. Svoboda, and J. Matas. Adaptive parameter optimization for real-time tracking. In *workshop on Non-rigid registration and tracking through learning*, Rio de Janeiro, Brazil, 2007. 4, 7

Index of Authors