Proceedings of the 21st Computer Vision Winter Workshop

Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)

Rimske Toplice, Slovenia
February 3-5, 2016

Proceedings of the 21st Computer Vision Winter Workshop
February 3-5, 2016, Rimske Toplice, Slovenia

© Slovenian Pattern Recognition Society, Ljubljana, February 2016
Volume Editors: Luka Čehovin, Rok Mandeljc, Vitomir Štruc
Publisher
Slovenian Pattern Recognition Society, Ljubljana 2016
Electronic edition
Slovenian Pattern Recognition Society, Ljubljana 2016
© SDRV 2016

CIP - Katatožni zapis o publikaciji
Narodna univerzitetna knjižnica, Ljubljana

004.93(082)(086.034.4)
004.8(082)(086.034.4)
COMPUTER Vision Winter Workshop (21 ; 2016 ; Rimske Toplice)
Proceedings of the 21st Computer Vision Winter Workshop, Rimske Toplice, Slovenia, February 3-5, 2016 [Elektronski vir] / Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.). - Electronic ed. - Ljubljana : Slovenian Pattern Recognition Society, 2016

ISBN 978-961-90901-7-6
1. Čehovin, Luka
283229440

Maloprodajna cena: 19,99 €

Message from the program chairs

It is our pleasure and privilege to welcome you to the 21st Computer Vision Winter Workshop (CVWW2016). This year the workshop is organized by the Slovenian Pattern Recognition Society (SPRS), and held in Rimske Toplice, Slovenia, from of February 3rd to February 5th, 2016. We hope that your experience at CVWW is both professionally and personally rewarding!

The Computer Vision Winter Workshop (CVWW) is an annual international meeting of several computer vision research groups, located in Ljubljana, Prague, Vienna, and Graz. The aim of the workshop is to foster interaction and exchange of ideas among researchers and PhD students. The focus of the workshop spans a wide variety of computer vision and pattern recognition topics, such as image analysis, medical imaging, 3D vision, human-computer interaction, vision for robotics, machine learning, as well as applied computer vision and pattern recognition.

CVWW 2016 received a total of 23 submissions from six countries. The paper selection was coordinated by the Program Chairs, and included a rigorous double-blind review process. The international Technical Program Committee consisted of 39 renowned computer vision experts, who conducted the review. Each submission was examined by at least three experts, who were asked to comment on the strengths and weaknesses of the papers and justify their recommendation for accepting or rejecting a submission. The Program Chairs used the reviewers' comments to render the final decision on each paper. As a result of this review process, 8 papers were accepted for oral presentation, and 6 papers were accepted for presentation in the form of a poster. Authors of the accepted posters were also given the opportunity to present their work in the form of short one-minute talks at a designated spotlight session. 8 papers were accepted for presentation at the workshop in the form of *invited presentations of on-going* *work* (6 orals and 2 posters), and are not included in the proceedings to avoid conflicts with potential future submissions of the presented material. The Program Chairs would like to thank all reviewers for their high-quality and detailed comments, which served as a valuable source of feedback for all authors, and most of all for their time and effort, which helped to make the CVWW2016 a success.

The workshop program included an invited talk by dr. Mario Fritz (Laboratory for Autonomous Intelligent Systems, Department of Computer Science, University of Freiburg), to whom we thank for his participation. We also extend our thanks to the Slovenian Pattern Recognition Society, through which the workshop was organized.

CVWW 2016 benefits from its sponsors; and we want to acknowledge and thank our supporters from KOLEKTOR and the Faculty of Computer and Information Science for their contributions. To all the sponsors and their representatives in attendance, thank you!

We hope that the 21st iteration of the Computer Vision Winter Workshop is a productive and enjoyable meeting for you and your colleagues, and inspires new ideas that can advance your professional activities.

Welcome and thank you for your participation!

Luka Čehovin, Rok Mandeljc, Vitomir Štruc
CVWW2016 Program Chairs
Ljubljana, Slovenia, January 2016

Committes

PROGRAM CHAIRS

- Luka Čehovin (FRI University of Ljubljana)
- Rok Mandeljc (FRI, FE University of Ljubljana)
- Vitomir Štruc (FE University of Ljubljana)

PROGRAM COMMITTEE

Csaba Beleznai	Stanislav Kovacic	Rene Ranftl
Horst Bischof	Matej Kristan	Daniel Prusa
Jan Cech	Walter Kropatsch	Peter Roth
Ondrej Chum	Vincent Lepetit	Robert Sablatnig
Ondrej Drbohlav	Jiri Matas	Radim Sara
Boris Flach	Martin Matousek	Walter Scheirer
Vojtech Franc	Mirko Navara	Alexander Shekhovtsov
Friedrich Fraundorfer	Tomas Pajdla	Danijel Skocaj
Margrit Gelautz	Peter Peer	Tomas Svoboda
Michal Havlena	Roland Perko	Peter Ursic
Yll Haxhimusa	Janez Pers	Tomas Vojir
Václav Hlaváč	Roman Pfugfelder	Andreas Wendel
Ines Janusch	Thomas Pock	Paul Wohlhart

Contents

7. *Hessian Interest Points on GPU* [PDF]
   Jaroslav Sloup, Jiri Matas, Michal Perdoch, Stepan Obdrzalek* (Czech Technical University in Prague)
8. *BaCoN: Building a Classifier from only N Samples* [PDF]
   Georg Waltner*, Michael Opitz, Horst Bischof (Graz University of Technology)
9. *Cuneiform Detection in Vectorized Raster Images* [PDF]
   Judith Massa, Bartosz Bogacz*, Susanne Krömker, Hubert Mara (University Heidelberg)
10. *2D tracking of Platynereis dumerilii worms during spawning* [PDF]
    Daniel Pucher*, Walter Kropatsch, Nicole Artner, Stephanie Bannister (Vienna University of Technology)
11. *Significance of Colors in Texture Datasets* [PDF]
    Milan Šulc*, Jiri Matas, (Czech Technical University in Prague)
12. *A Novel Concept for Smart Camera Image Stitching* [PDF]
    Hanna Huber, Majid Banaeyan*, Raphael Barth, Walter Kropatsch (Vienna University of Technology)
13. *A concept for shape representation with linked local coordinate systems* [PDF]
    Manuela Kaindl*, Walter Kropatsch (Vienna University of Technology)
14. *A Computer Vision System for Chess Game Tracking* [PDF]
    Can Koray*, Emre Sumer (Başkent University)
15. *Fast L1-based RANSAC for homography estimation* [PDF]
    Jonáš Šerých*, Jiri Matas, Ondrej Drbohlav (Czech Technical University in Prague)

Invited talk

Towards a Visual Turing Test: Answering Questions on Images

Mario Fritz
Max Planck Institute for Informatics and Saarland University

Abstract

We address the task of automatically answering questions on images by bringing together latest advances from natural language processing and computer vision. In order to quantify progress on this challenging problem, we have established the first benchmark for this challenging problem that can be seen as a modern attempt at a visual turing test. Our first approach to this problem follows a more traditional AI approach, where we combine discrete reasoning with uncertain predictions by a multi-world approach that models uncertainty about the perceived world in a bayesian framework. More recently, we build on the success of deep learning techniques and propose an end-to-end formulation of this problem for which all parts are trained jointly. Looking forward, we see these two approach as two ends of a spectrum ranging from symbolic representations to vector-based embedding that we are currently exploring.

Sponsors

University of Ljubljana
Faculty of Computer and Information Science

KOLEKTOR

21st *Computer Vision Winter Workshop*
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

A Longitudinal Diffeomorphic Atlas-Based Tissue Labeling Framework for Fetal Brains using Geodesic Regression

Roxane Licandro[1,2]
licandro@caa.tuwien.ac.at
[1] Institute of Computer Aided Automation, Computer Vision Lab, Vienna University of Technology,
http://www.caa.tuwien.ac.at/cvl

[2] Department of Radiology and Image-guided Therapy, Computational Imaging Research Lab,
Medical University of Vienna,
http://www.cir.meduniwien.ac.at

Georg Langs[2] Gregor Kasprian[2] Robert Sablatnig[1] Daniela Prayer[2]
Ernst Schwartz[2]

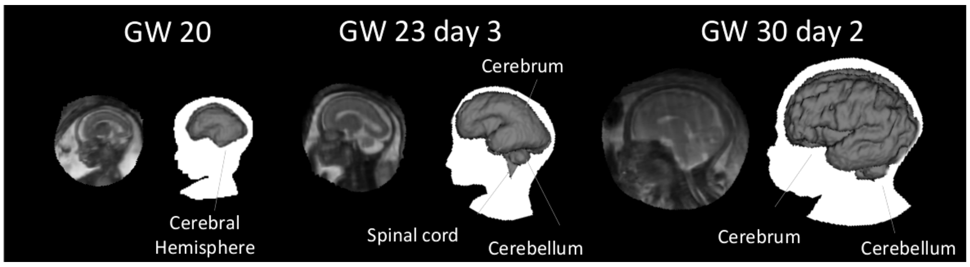
Abstract. *The human brain undergoes structural changes in size and in morphology between the second and the third trimester of pregnancy, corresponding to accelerated growth and the progress of cortical folding. To make fetal brains comparable, spatio-temporal atlases are used as a standard space for studying brain development, fetal pathology locations, fetal abnormalities or anatomy. The aim of this work is to provide a continuous model of brain development and to use it as base for an automatic tissue labeling framework. This paper provides a novel longitudinal fetal brain atlas construction concept for geodesic image regression using three different age-ranges which are parametrized according to the developmental stage of the fetus. The dataset used for evaluation contains* 45 *T2$-$weighted Magnetic Resonance (MR) volumes between Gestation Week (GW)* 18.0 *and GW* 30 *day* 2*. The automatic tissue labeling framework estimates cortical segmentations with a Dice Coefficient (DC) of up to* 0.85 *and ventricle segmentations with a DC of up to* 0.60.

1. Introduction

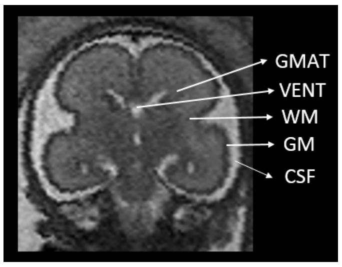
The aim of brain mapping experiments is to create maps (models), based on studies, to understand structural and functional brain organization. To this end, neuroimaging methods as well as knowledge of neuroanatomy and physiology are combined. Due to the fundamental changes occurring in the human fetal brain during pregnancy, a single map is not sufficient to model brain development [19]. Changes in size, according to accelerated growth, changes in morphology, due to the progress of cortical folding and deceleration of the proliferation of ventricular progenitor cells [16] occur and are illustrated in Figure 1a. Thus, a collection of brain maps is needed to describe these alterations as a function of time. For studying the brain organisation during its development, abnormalities, and locations of pathologies, brain maps are used as a reference model [18]. Newly acquired brain images are labelled to identify structures and possible abnormal changes or to find indicators for diseases. This labeling can be performed manually by annotating the images, which needs an expert, time and consequently leads to increased costs compared to an automatic labeling procedure [3]. In this case, labels for non annotated images are estimated automatically by software using a brain model for the mapping. Such an automated labeling procedure on the one hand and a reference model on the other form an atlas. To cover the time-dependent development of the fetal brain, time-varying reference models are considered for building spatio-temporal atlases.

1.1. State-of-the-Art

State-of-the-art approaches [8, 10, 13, 17, 21] for computing a spatio-temporal atlas combine registration methods and interpolation techniques to obtain



(a) Fetal Brain Development



(b) Observable Brain Structures

Figure 1: Left: MR imaging and schematic illustration of the fetal brain development at GW 20, 23 day 3 and 30 day 2. Right: Illustration of identifiable brain structures in a T2 weighted fast MR image acquired with a 1.5 Tesla scanner (Grey Matter (GM), White Matter (WM), the VENTricles (VENT) and the Germinal MATrix (GMAT) [21]). Also extraventricular Cerebro Spinal Fluid (CSF), Deep Grey Matter (DGM) and Non-Brain structures (NB), like skull or amniotic fluid are identifiable. MR images courtesy of Medical University of Vienna (MUW).

continuity in time. The use of an "all-to-one" approach (a single subject as reference) introduces substantial bias. The brain structures of fetuses cannot be described by one image, since it does not reflect occurring changes over time [10, 17]. Exclusive pairwise affine registration for image alignment results in blurred regions in the templates obtained by intensity averaging. Affine registration is not capable of compensating local inter-subject variability [17]. This leads to worse registration results between atlas-based segmentations and individual objects compared to non-rigid approaches, which show a higher level of detail [17]. An advantage of pairwise approaches lies in the registration of wider age-ranges between 15 to 18 Gestation Weeks (GW), compared to groupwise approaches, which are able to cover only small age ranges between 5 to 8 GW. A benefit of groupwise registration approaches is the template-free estimation of the initial reference space. The template is estimated and updated during the registration procedure [10]. The main limitations of groupwise registration lie in the lower level of anatomic definition [17]. Examples for pairwise approaches can be found in [10, 17] and for groupwise approaches in [8, 13, 21].

1.2. Challenges

Imaging of a fetus in utero is challenging, due to its constantly changing position, which causes image unsharpness and artefacts [5]. Thus, a main issue in fetal imaging lies in shortening the image acquisition time to 20 seconds and to use motion correction techniques [4]. The Magnetic Resonance (MR) imaging technique is used as an alternative to ultrasonography for prenatal diagnosis and is able to image a fetus in a non-invasive way. Distinguishable brain structures using this technique are illustrated in Figure 1b. A problem of MR imaging is the lack of comparability and constancy of gray-values. Thus, for the comparison of brains of adult patients, an atlas as a standard space is required, which avoids the gray-value discrepancies. The brains are mapped to a standardized coordinate system according to marked anatomical locations. However, the fetal brain is a developing structure. In comparison to building an atlas of an adult brain, the fast change of a fetal brain in shape and size has to be taken into account [10]. Also, fetal brains at a certain GW show differences in orientation shape and size. Possible reasons are the inaccuracy in determination of the gestational age, inter-patient variability or pathological growth processes [15]. The motivation for building a fetal atlas is the possibility to compare fetal brains for studying brain development, fetal pathology locations, fetal abnormalities or anatomy.

1.3. Contribution

We create a tissue labeling framework for cortical and ventricle structures in the fetal brain from GW 18 to GW 30. An automatic segmentation procedure including a longitudinal fetal brain atlas and a labeling procedure are considered. In our work we demonstrate that image regression is capable to build a spatio-temporal atlas of the fetal brain and is able to model a mean trajectory encoding the brain development in a single diffeomorphic defor-

mation, instead of calculating discrete age-dependent templates combined with interpolation. As found in literature [7, 9, 11], image regression for time-series data have been evaluated only using adult- and child-brain datasets, which record changes of brain structure over time. In the proposed work the local inter-subject variability is considered to be modelled continuously in time and non-rigidly in space by geodesic regression [1, 2]. The computed atlas is used as a prior of the Graph Cut (GC) approach for multi label segmentation proposed by Yuan et al. [20].
The paper is organized as follows. In Section 2 an overview of the methodology used and the concept of the tissue labeling framework proposed is presented. The results and the corresponding discussion are given in Section 3. This work concludes with a summary of the contributions in Section 4.

2. Methodology

The framework proposed is illustrated in Figure 2. The input represents a gray value image I_{new} at time point t_{new}, which is preprocessed in a first step, by performing motion correction, rigid alignment, image masking and image cropping. Subsequently, the longitudinal diffeomorphic fetal brain atlas is used to estimate a time point t_{new} corresponding diffeomorphic transformation for computing a time-dependent intensity image I_A and a time-dependent segmentation for ventricular and cortical tissue S_A^{tissue} in atlas space. In a pairwise registration procedure, a transformation T from the preprocessed input (*Aligned* I_{new}) to the atlas-based intensity image I_A is estimated. The inverse of the computed transformation T^{-1} is used to transform the atlas based segmentations S_A^{tissue} to the subject's space ($S_A^{tissue} \circ T^{-1} = S_{GC}^{tissue}$). As next step the transformed segmentations S_{GC}^{tissue} and I_{new} are used as input parameters for the multi label GC segmentation refinement. The output of the framework are segmentations for ventricular and cortical brain tissues S_{new}^{tissue} of the input image I_{new}.

2.1. Image Acquisition and Preprocessing

The time series MR image dataset used consists of 45 healthy fetal brains with an age range between 18 and 30 GW. The MR image acquisition is performed using an 1.5 Philips Gyroscan superconducting unit scanner performing a single-shot, fast spin-echo T2-weighted MR sequence: In-plane resolution = 0.78 - 0.9 pixels per mm, Slice thickness = 3 - 4.4mm, Acquisition matrix = 256×256, Field of view = 200 - 230mm, Specific Absorption Rate (SAR) = $<$ 100% /4.0W/kg, Image acquisition time = $\leq$ 20s, TE (Echo Time) = 100 - 140ms, TR (Repetition Time) = 9000 - 19000ms. The dataset of MR images used for atlas learning are preprocessed using the pipeline illustrated in Figure 2. First the images are motion corrected using the toolkit for fetal brain MR images published by Rousseau et al. [14]. Subsequently, the manual annotation of the cortex, left and right eye, ventricle and occipital foramen magnum is performed by an expert. After this step, the data is rigidly aligned, the surrounding mother tissue is excluded in a masking step and the volumes are cropped to reduce computational costs in the longitudinal registration procedure using a bounding box of size 90 × 140 × 140 voxels.

2.2. Spatio Temporal Atlas Learning

The algorithm used for Diffeomorphic Anatomical RegistraTion using Exponential Lie algebra (DARTEL) of Ashburner et al. [1, 2] for geodesic regression is integrated in the Statistical ParaMetric (SPM) tool box - release SPM8 [1]. This approach is used to encode the brain development in a single diffeomorphic deformation by optimising the energy term E expressed in Equation 1 [2].

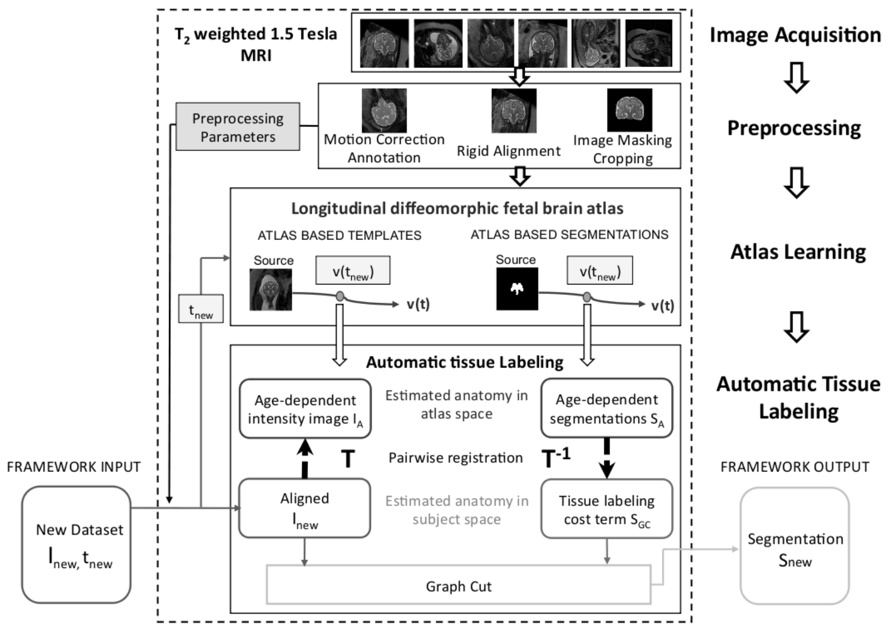
$$E = \frac{1}{2}\left\|Lv_0\right\|^2 + \frac{1}{2}\sum_{n=1}^{N}\left(\int_{x\in\Omega}\left\|I_{t_0} - I_{t_n}(\varphi_{t_n})\right\|^2 dx\right) \tag{1}$$

The term φ_{t_n} denotes the forward deformation from source I_{t_0} to target I_{t_n} at time point t_n, where $n = 1, \ldots, N$ and L represents a model of the "inertia" of the system, i.e. a linear operator which operates on a time-dependent velocity that mediates the deformation over unit time [2]. It is introduced to derive an initial momentum m_0 through an initial velocity v_0. The velocity field $v(x)$ learned at position x is parametrised using a linear combination of i basis functions. Such basis functions consist of a vector of coefficients c_i and a i^{th} first degree B-spline basis function $\rho_i(x)$ (cf. Equation 2) [1].

$$v(x) = \sum_i c_i\rho_i(x) \tag{2}$$

The aim of the DARTEL implementation is to estimate an optimized parametrisation of c. The energy

[1] `http://www.fil.ion.ucl.ac.uk/spm/`; [accessed 07 December 2015]



T2 weighted 1.5 Tesla MRI
Image Acquisition
Preprocessing Parameters
Motion Correction Annotation
Rigid Alignment
Image Masking Cropping
Preprocessing
Longitudinal diffeomorphic fetal brain atlas
ATLAS BASED TEMPLATES
ATLAS BASED SEGMENTATIONS
Source
v(tnew)
v(t)
tnew
Atlas Learning
Automatic tissue Labeling
Age-dependent intensity image IA
Estimated anatomy in atlas space
Age-dependent segmentations SA
T
Pairwise registration
T-1
FRAMEWORK INPUT
New Dataset Inew, tnew
Aligned Inew
Estimated anatomy in subject space
Tissue labeling cost term SGC
Graph Cut
Automatic Tissue Labeling
FRAMEWORK OUTPUT
Segmentation Snew

Figure 2: Fetal brain tissue labeling framework. MR images courtesy of MUW.

cost term E in Equation 1 is reformulated in terms of finding the coefficients of c for a given dataset D with maximum probability (cf. Equation 3). A maximization of the probability leads to the minimization of its negative logarithm and thus, is used to interpret registration of data D as a minimization procedure of the objective function $(-\log p(c, D))$ expressed in Equation 3, consisting of a prior term $(-\log p(c))$ and a likelihood term $(-\log p(D|c))$ [1].

$$-\log p(c, D) = -\log p(c) - \log p(D|c) \qquad (3)$$

The prior term denotes the prior probability $p(c)$. Ashburner et al. [1] use a concentration matrix (inverse of a covariance matrix) K to encode spatial variability. The parameters $[\lambda_1, \lambda_2, \lambda_0, \lambda, \mu]$, which have to be predefined to compute K, influence the behaviour of the deformation (bending energy, stretching, shearing) as well as the divergence and amount of volumetric expansion or contraction [1]. The term λ_0 encodes the penalisation of absolute displacements, λ_1 penalises the difference between two neighboured vectors by observing the first derivatives (linear term) of the displacements, λ_2 penalises the difference between the first derivatives of two neighboured vectors by observing the second derivatives of the displacements and λ denotes the variability of the spatial locations (divergence of each point in the flow field) with a constant value. Increasing λ leads to increasing smoothing of the flow vector field and preserves volumes during the transformation. The term μ encodes the variance according to symmetric components, rotations and the penalisation of scaling and shearing. The likelihood term encodes the probability of c given the data D [1] and corresponds to the mean-squared difference between a warped template deformed by the calculated transformation and the target image.

2.2.1 Optimisation Procedure

A Full Multi Grid (FMG) approach is used to solve the equation (cf. Equation 4) which is needed to update the vector field during a Gauß-Newton optimising procedure, where H^{iter} denotes the Hessian, g^{iter} the gradient and K the concentration matrix. Details regarding the computation of v_0^{iter+1} are explained in [1, 2].

$$v_0^{iter+1} = v_0^{iter} - \epsilon(K + H^{iter})^{-1}(Kv_0^{iter} + g^{iter}) \quad (4)$$

For this task images are observed in different scales. For every resolution level multigrid methods recursively estimate the field, starting at the coarsest scale and computing the residual to solve the update equations on the current grid. Subsequently, the solution is prolongated to the next finer grid [1].

2.3. Automatic Tissue Labeling using Graph Cuts

For tissue labeling, we use a continuous max flow formulation of a multi label GC [20]. Three input parameters are necessary for performing tissue segmentation. A data term (gray value volume I_{new}

at age t_{new}), a cost (unary) term, and a penalty (binary) term. For computing a unary term, atlas based segmentations for cortex and ventricle tissue $S_{tissue} = \{S_{cortex}, S_{ventricle}\}$ at age t are estimated and smoothed with a Gaussian filter G. The parameter δ is defined to weight the smoothed result with a constant factor. The unary term is illustrated in Equation 5, where $\star$ denotes the convolution operator.

$$C = \delta * (S_{tissue} \star G) \tag{5}$$

Three different binary terms are evaluated:
Penalty term 1 (P$_1$) is a weighted norm of the gradient of the data term D (cf. Equation 6), where δ denotes the same weighting term as used in Equation 5 and a, b are constant weighting parameters.

$$P_1 = \delta * \frac{b}{1 + (a * \|\nabla D\|)} \tag{6}$$

Penalty term 2 (P$_2$) denotes an intensity based term and is calculated separately for cortex and ventricle segmentation (cf. Equation 7). Tissue type specific gray values are modelled as Gaussian distributions N$\sim(\mu_{tissue}, \sigma_{tissue})$, which parameters μ_{tissue} and σ_{tissue} are estimated using the a-priori atlas segmentation. These parameters are used to calculate the probability of every pixel belonging to cortex or ventricle. Subsequently, the gradient of the resulting probability map P and its norm are computed and weighted by the parameters δ, a, b as shown in Equation 6.

$$P_2 = \delta * \frac{b}{1 + (a * \|\nabla P(\mu_{tissue}, \sigma_{tissue})\|)} \tag{7}$$

Penalty term 3 (P$_3$) represents an exponential formulation and is expressed in Equation 8. The parameter u is a constant and v a linear weighting parameter. The term w weights the norm of the image's D gradient non-linearly in the exponential term.

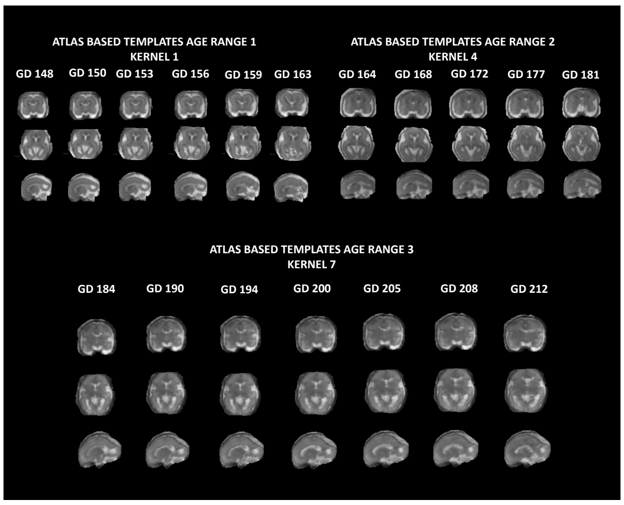
$$P_3 = u + v * exp\left(-\frac{\|\nabla D\|}{w}\right) \tag{8}$$

3. Results

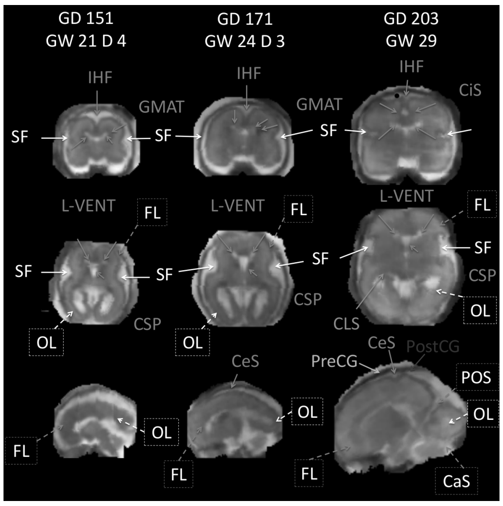
Evaluation of the proposed framework is performed using leave-one-out cross validation. In this paper a novel longitudinal registration procedure is formulated by dividing the data set into three age ranges, based on the developmental stage of the fetus. Age range 1 reaches from 20 GW day 6 (146 GD) to 23 GW day 3 (164 GD), age range 2 from 23 GW day 3 (164 GD) to 26 GW day 2 (184 GD) and age range 3 from 26 GW day 2 (184 GD) to 30 GW day 2 (212 GD). The first part of the evaluation documents the atlas learning results for each age range. Subsequently, the atlases computed are used to evaluate the tissue labeling procedure as a second part of the evaluation. Estimated atlas templates at the testing timepoint are pairwise registered to the test MR volume to obtain a transformation T. The inverse T^{-1} is used to transform the atlas based segmentation to the test-subject's space. As last step the segmentation of the test volume using the transformed atlas is computed. To evaluate our approach, we report the overlap between automatic- and manual segmentations of the fetal cortex and ventricles. In the leave-one-out cross validation, we compare the Dice Coefficient (DC) [6] between the groundtruth annotation and different automatic segmentations based on (1) the atlas, (2) the transformed atlas, and (3) the GC segmentation optimization.
Furthermore, we report the volume of cortex and ventricles, and the area of the cortical surface of the atlas based segmentations.

3.1. Results Spatio-Temporal Atlas Learning

The deformation behaviour of image regression using 21 different regularisation kernels $K\,[\lambda_1, \lambda_2, \lambda_0, \lambda, \mu]$ (cf. Section 2.2) is evaluated for every age range. Beside the DC also the behaviour of the regularisation of the volume expansion and changes of the area of cortical surface have to be taken into account, when choosing a suitable kernel. Atlas-based cortical and ventricle segmentations are studied. According to the evaluation results, kernel 1 ($K_1\left[0.01, 0.01, 9e^{-6}, 1e^{-5}, 1e^{-5}\right]$) is chosen as suitable regularisation for age range 1, kernel 4 ($K_4\left[0.01, 9e^{-6}, 9e^{-6}, 0.01, 1e^{-5}\right]$) for age range 2 and kernel 7 ($K_7\left[0.01, 0.01, 9e^{-6}, 0.01, 1e^{-5}\right]$) for age range 3. Figure 3a shows examples of the atlas templates learned and Figure 3b illustrates the anatomical details of these at age GW 21 day 4 (GD 151), GW 24 day 3 (GD 171) and GW 29 (GD 203). In both figures the growth of the brain structures is observable. The brain model at age range 1 is characterised by a smoother cortex surface in comparison to a brain at a higher age range. It also visualises the increase of the cortical folding grade. According to Pugash et al. [12], the ventricles achieve their thickest size in early gestation and regress in the third trimester, which is not visible. The regularisa-



(a) Atlas based templates



(b) Details Atlas based templates

Figure 3: Left: Atlas based templates of age range 1, 2 and 3 between GW 21 day 1 (GD 148) and GW 30 day 2 (GD 212). Right: Anatomical details of atlas based templates at age GW 21 day 4 (GD 151), GW 24 day 3 (GD 171) and GW 29 (GD 203). Coronal (first row), axial (second row) and sagital (third row) slices are illustrated. Denoted structures: Sylvian Fissure (SF), InterHemispheric Fissure (IHF), Germinal MATrix (GMAT), Lateral-VENTricle (L-VENT), Cingulate Sulcus (CiS), ColLateral Sulcus (CLS), Cavum of Septum Pellucidum (CSP), Occipital Lobe (OL), Frontal Lobe (FL), Central Sulcus (CeS), PreCentral Gyrus (PreCG), PostCentral Gyrus (PostCG), ParietoOccipital Sulcus (POS) and Calcarine Sulcus (CaS).

tion term for geodesic regression is not able to model location specific volume expansion and shrinkage at the same time. This leads to worse modelling results for ventricles, compared to cortical structure, since a kernel is chosen which models expansion. Additionally, the subject specific variability of age-dependent ventricle size in the dataset and the complex form of ventricles complicate the determination of a suitable kernel and consequently the registration procedure. Observable structures at every age range are Sylvian Fissures (SF), Lateral VENTricle (L-VENT), InterHemispheric Fissure (IHF), Cavum of Septum Pellucidum (CSP), Occipital Lobe (OL) and Frontal Lobe (FL). The SF show in the coronal and axial slices a smooth bending at age range 1 and develop to a deep fold at the lateral side of the brain at age range 3. Also the IHF shows a deeper folding at age range 3 with Cingulate Sulcus (CiS) as additional forming compared to age range 1. The Germinal MATrix (GMAT) is existent until age range 2 and disappears later in the third trimester of pregnancy. The Central Sulcus (CeS) formation starts at age range 2 and gets more apparent at age range 3 as well as the developing of the PreCentral Gyrus (PreCG) and PostCentral Gyrus (PostCG). The ColLateral Sulcus (CLS) is visible at age range 3 as well as the Calcarine Sulcus (CaS) and PreOccipital Sulcus (POS).

3.2. Results Automatic Tissue Labeling

For pairwise registration kernel A ($K_A\left[5e^{-3}, 5e^{-3}, 3e^{-5}, 1e^{-5}, 9e^{-6}\right]$) is used for regularisation. The DC distributions of segmentations of the cortex for age range 1, 2 and 3 are illustrated in Figure 4 on the top and for ventricle segmentations on the bottom. The DC distribution of atlas based and transformed atlas-based segmentations using pairwise registration are illustrated and the three dotted lines visualise the DCs of GC based segmentations computed using penalty terms 1, 2 and 3. For age range 1 the highest DC improvement from 0.727 to 0.771 at GD 158 is achieved by pairwise registration and GC refinement compared to atlas based segmentations. In contrast to this no improvement is reached at GD 151, but shows the highest DC of about 0.851. At GDs older than 154 the GC refining using penalty 1 and penalty 2 achieve a higher DC increase of about 0.02 compared to using penalty 3. At age range 2 no improvement of transformed atlas based segmentations is observed after pairwise registration, which leads to a decrease of the DC. It is observed that the labeling result of the pairwise registration

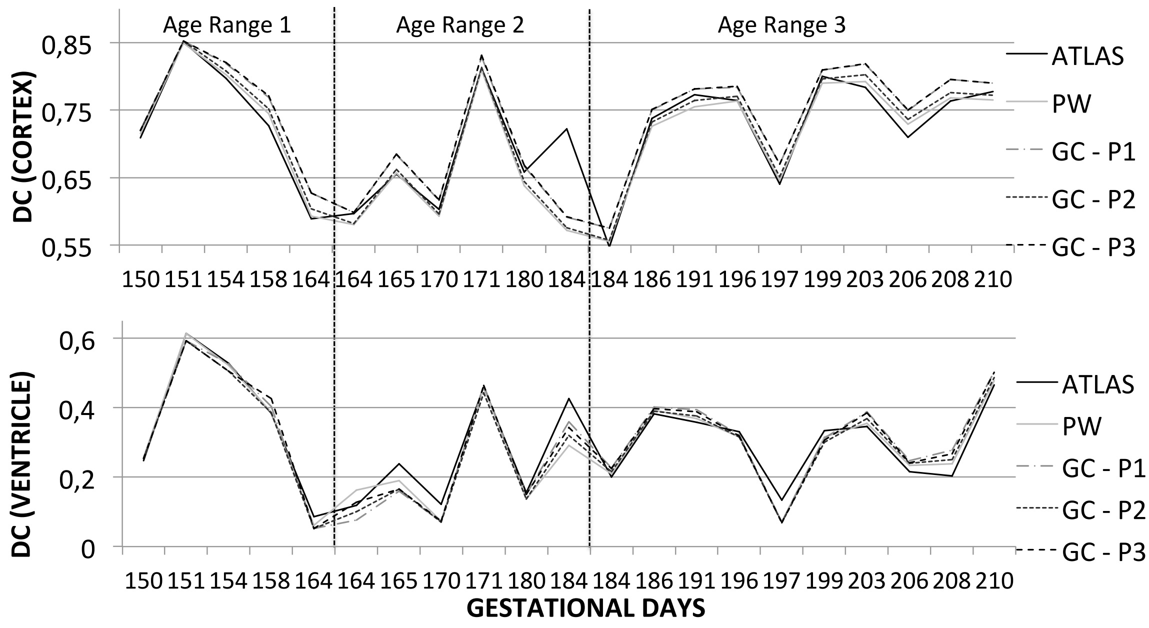


Figure 4: DCs of automatically estimated labels of the cortex and ventricle at age range 1, 2 and 3.

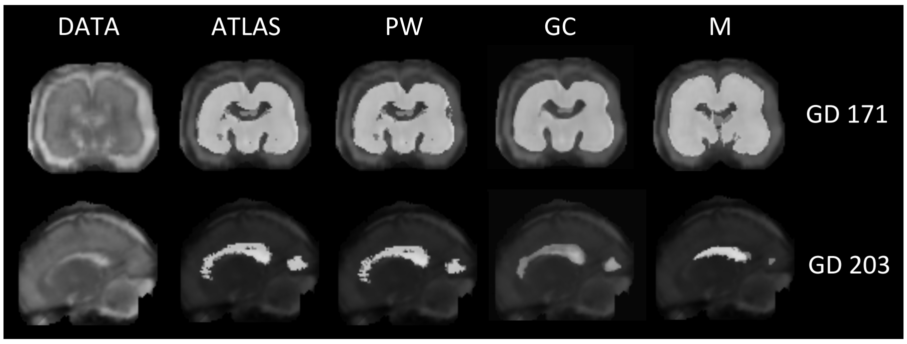


Figure 5: Top: Coronal view - segmentations of the cortex at GD 171 (GW 24 day 3), bottom: sagital view - segmentations of the ventricle at GD 203 (GW 29). Segmentations are illustrated estimated by the atlas (ATLAS), after the pairwise registration procedure (PW), estimated by the GC approach (GC) and manual annotations (M).

has an influence on the GC labeling since it acts as initialization of this procedure, best visible at GD 184. The GC refinement is able to compensate the results of the pairwise registration between GD 164 and 184 and shows an increase of the DC between atlas and graph-cut based segmentations in average of about 0.02. At age range 3 an increase of DC at every age range is achievable using GC refinement. The highest improvement between atlas-based segmentations and GC based segmentations is reached at GD 206 with a DC increase from 0.71 to 0.795. The highest DC at age range 3 of about 0.819 is achieved at GD 203 and the lowest of about 0.575 at GD 184. It is observable that pairwise registration is not capable to compensate differences in volume size or absolute displacements. If an estimated segmentation has a bigger volume than the structure to be segmented or is displaced, then the borders of neighboured tissue prevents the GC approach from cutting through regions of a high gradient, since this would lead to increasing costs in the energy minimisation procedure. Consequently, the GC is not capable to refine the segmentation. In Figure 5 an example for a misaligned segmentation and its deformation through the labeling procedure is illustrated. The displacement is observable at the IHF in the first column and the superior part of the anterior horn of the ventricle in the second column. Test data and corresponding estimated segmentations, transformed segmentations to subject's space and GC based segmentations of the cortex at GD 171 (top) and of ventricular tissue at GD 203 (bottom) are shown. The GC segmentations are computed using the penalty term 3, since it shows the best improvement between atlas-based and GC based segmentations.

4. Conclusion

In this paper an automatic fetal brain tissue labeling framework using geodesic image regression was presented and was identified to be suitable as registration approach to longitudinally model the changes of the brain during the 18^{th} and 30^{th} GW. The advantage is the provision of a time-dependent transformation from a source to a target brain, instead of combining a template building technique and interpolation technique to obtain continuity in time. A novel longitudinal registration scheme was proposed, using separate age ranges for flexible regularisation of the deformation behaviour due to the age range dependent changes. The atlas learned was evaluated using a leave-one-out cross validation approach for every age range and 21 different regularisation kernels were analysed according to their behaviour regarding volume expansion, modelling of cortical surface and Dice similarity to manual annotations. The fetal brain atlas proposed is not capable of modelling the thinning of ventricles from age range 1 to age range 3. Since the proposed method uses one regularisation kernel per age range, geodesic regression is not able to regularise location specific volume expansion and shrinkage at the same time. To overcome this issue, the usage of tissue specific regularisation and consequently the computation of separate ventricle atlases are a possible solution. In contrast to this, the increase of the cortical folding grade and of the volume over time are integrated in the proposed spatio-temporal model. The quality of transformed atlas based segmentations to subject's space using pairwise registration leads to the conclusion that the kernel for pairwise registration has to be defined differently according to the age range and also tissue type, for being able to improve the graph cut initialisation term. Additionally, it is shown that the quality of graph cut labeling is dependent on the initialisation cost term (atlas segmentation) and the penalty term. A false or displaced atlas segmentation hinders as cost term the refinement of the graph cut based labeling. Finally the proposed framework is able to estimate cortex segmentations with a DC up to 0.85 and ventricle segmentations up to 0.60. We show that image regression is capable to model the variability of fetal brains in time and is qualified to be used for building a spatio-temporal atlas as basis for fetal brain tissue segmentation. The evaluation of the cortical labeling results for age range 1, 2 and 3 show that a single kernel for pairwise registration for every age range is not suitable. Thus, a main focus of future work will lie in the improvement of the labeling procedure, by evaluating age range and tissue dependent regularisation, to improve the quality of graph cut based segmentation. Additionally, a combination of global rigid and local deformable pairwise registration could be analysed for transforming atlas based segmentations to the subject's space as extension to this work.

Acknowledgements

This work was co-funded by ZIT - Life Sciences 2014, grant number 1207843, Project Flowcluster, and by OeNB (15929).

References

[1] J. Ashburner. A fast diffeomorphic image registration algorithm. *NeuroImage*, 38(1):95–113, Oct. 2007. 3, 4

[2] J. Ashburner and K. Friston. Diffeomorphic registration using geodesic shooting and Gauss-Newton optimisation. *NeuroImage*, 55(3):954–967, Apr. 2011. 3, 4

[3] M. Becker and N. Magnenat-Thalmann. Deformable models in medical image segmentation. In N. Magnenat-Thalmann, O. Ratib, and H. Choi, editors, *3D Multiscale Physiological Human*, pages 81–106. Springer London, Jan. 2014. 1

[4] L. Breysem, H. Bosmans, S. Dymarkowski, D. V. Schoubroeck, I. Witters, J. Deprest, P. Demaerel, D. Vanbeckevoort, C. Vanhole, P. Casaer, and M. Smet. The value of fast MR imaging as an adjunct to ultrasound in prenatal diagnosis. *European Radiology*, 13(7):1538–1548, July 2003. 2

[5] M. Clemence. How to shorten MRI sequences. In D. Prayer, editor, *Fetal MRI*, Medical Radiology, pages 19–32. Springer Berlin Heidelberg, 2011. 2

[6] L. Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, July 1945. 5

[7] S. Durrleman, X. Pennec, A. Trouvé, J. Braga, G. Gerig, and N. Ayache. Toward a comprehensive framework for the spatiotemporal statistical analysis of longitudinal shape data. *International Journal of Computer Vision*, 103(1):22–59, May 2013. 3

[8] P. Habas, K. Kim, J. Corbett-Detig, F. Rousseau, O. Glenn, A. Barkovich, and C. Studholme. A spatiotemporal atlas of MR intensity, tissue probability and shape of the fetal brain with application to segmentation. *NeuroImage*, 53(2):460–470, Nov. 2010. 1, 2

[9] Y. Hong, Y. Shi, M. Styner, M. Sanchez, and M. Niethammer. Simple geodesic regression for image time-series. In B. Dawant, G. Christensen, J. Fitzpatrick, and D. Rueckert, editors, *Biomedical Image Registration*, number 7359 in Lecture Notes in Computer Science, pages 11–20. Springer Berlin Heidelberg, Jan. 2012. 3

[10] M. Kuklisova-Murgasova, P. Aljabar, L. Srinivasan, S. Counsell, V. Doria, A. Serag, I. Gousias, J. Boardman, M. Rutherford, A. Edwards, J. Hajnal, and D. Rueckert. A dynamic 4D probabilistic atlas of the developing brain. *NeuroImage*, 54(4):2750–2763, Feb. 2011. 1, 2

[11] M. Niethammer, Y. Huang, and F. Vialard. Geodesic regression for image time-series. *International Conference MICCAI 2011*, 14(Pt 2):655–662, 2011. 3

[12] D. Pugash, U. Nemec, P. Brugger, and D. Prayer. Fetal MRI of Normal Brain Development. In D. Prayer, editor, *Fetal MRI*, Medical Radiology, pages 147–175. Springer Berlin Heidelberg, Jan. 2011. 5

[13] L. Risser, F. Vialard, A. Serag, P. Ajabar, and D. Rueckert. Construction of diffeomorphic spatio-temporal atlases using Krcher means and LDDMM: Application to early cortical development. In *Workshop on Image Analysis of Human Brain Development (IAHBD), in International Conference MICCAI 2011*, Sept. 2011. 1, 2

[14] F. Rousseau, E. Oubel, J. Pontabry, M. Schweitzer, C. Studholme, M. Koob, and J. Dietemann. BTK: An Open-Source Toolkit for Fetal Brain MR Image Processing. *Computer methods and programs in biomedicine*, 109(1):65–73, Jan. 2013. 3

[15] T. Saul, R. Lewiss, and M. Rivera. Accuracy of emergency physician performed bedside ultrasound in determining gestational age in first trimester pregnancy. *Critical Ultrasound Journal*, 4(1):1–5, Dec. 2012. 2

[16] J. Scott, P. Habas, K. Kim, V. Rajagopalan, K. Hamzelou, J. Corbett-Detig, A. Barkovich, O. Glenn, and C. Studholme. Growth trajectories of the human fetal brain tissues estimated from 3D reconstructed in utero MRI. *International Journal of Developmental Neuroscience*, 29(5):529–536, Aug. 2011. 1

[17] A. Serag, P. Aljabar, G. Ball, S. Counsell, J. Boardman, M. Rutherford, A. Edwards, J. Hajnal, and D. Rueckert. Construction of a consistent high-definition spatio-temporal atlas of the developing brain using adaptive kernel regression. *NeuroImage*, 59(3):2255–2265, Feb. 2012. 1, 2

[18] C. Studholme. Mapping fetal brain development in utero using magnetic resonance imaging: the big bang of brain mapping. *Annual review of biomedical engineering*, 13:345–368, Aug. 2011. 1

[19] A. Toga and P. Thompson. 1 - an introduction to maps and atlases of the brain. In A.W. Toga and J.C. Mazziotta, editors, *Brain Mapping: The Systems*, pages 3–32. Academic Press, San Diego, 2000. 1

[20] J. Yuan, E. Bae, X. Tai, and Y. Boykov. A continuous max-flow approach to potts model. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Computer Vision ECCV 2010*, number 6316 in Lecture Notes in Computer Science, pages 379–392. Springer Berlin Heidelberg, Jan. 2010. 3, 4

[21] J. Zhan, I. Dinov, J. Li, Z. Zhang, S. Hobel, Y. Shi, X. Lin, A. Zamanyan, L. Feng, G. Teng, F. Fang, Y. Tang, F. Zang, A. Toga, and S. Liu. Spatialtemporal atlas of human fetal brain development during the early second trimester. *NeuroImage*, 82:115–126, Nov. 2013. 1, 2

21st *Computer Vision Winter Workshop*
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

Quantitative Comparison of Feature Matchers Implemented in OpenCV3

Zoltan Pusztai
Eötvös Loránd University
Budapest, Hungary
puzsaai@inf.elte.hu

Levente Hajder
MTA SZTAKI
Kende u. 13-17. Budapest, Hungary-1111
http://web.eee.sztaki.hu

Abstract. *The latest V3.0 version of the popular Open Computer Vision (OpenCV) framework has just been released in the middle of 2015. The aim of this paper is to compare the feature trackers implemented in the framework. OpenCV contains both feature detector, descriptor and matcher algorithms, all possible combinations of those are tried. For the comparison, a structured-light scanner with a turntable was used in order to generate very accurate ground truth (GT) tracking data. The tested algorithm on tracking data of four rotating objects are compared. The results is quantitatively evaluated as the matched coordinates can be compared to the GT values.*

1. INTRODUCTION

Developing a realistic 3D approach for feature tracker evaluation is very challenging since realistically moving 3D objects can simultaneously rotate and translate, moreover, occlusion can also appear in the images. It is not easy to implement a system that can generate ground truth (GT) data for real-world 3D objects. The Middlebury database[1] is considered as the state-of-the-art GT feature point generator. The database itself consists of several datasets that had been continuously developed since 2002. In the first period, they generated corresponding feature points of real-world objects [23]. The first Middlebury dataset can be used for the comparison of feature matchers. Later on, this stereo database was extended with novel datasets using structured-light [24] or conditional random fields [18]. Even subpixel accuracy can be achieved in this way as it is discussed in [22].

However, the stereo setup is too strict limitation for us, our goal is to obtain tracking data via multiple frames.

[1]http://vision.middlebury.edu/

The description of the optical flow datasets of Middlebury database was published in [3]. It was developed in order to make the optical flow methods comparable. The latest version contains four kinds of video sequences:

1. *Fluorescent images*: Nonrigid motion is taken by both color and UV-camera. Dense ground truth flow is obtained using hidden fluorescent texture painted on the scene. The scenes are moved slowly, at each point capturing separate test images in visible light, and ground truth images with trackable texture in UV light.

2. *Synthesized database*: Realistic images are generated by an image syntheses method. The tracked data can be computed by this system as every parameters of the cameras and the 3D scene are known.

3. *Imagery for Frame Interpolation.* GT data is computed by interpolating the frames. Therefore the data is computed by a prediction from the measured frames.

4. *Stereo Images of Rigid Scenes.* Structured light scanning is applied first to obtain stereo reconstruction. (Scharstein and Szeliski 2003). The optical flow is computed from ground truth stereo data.

The main limitation of the Middlebury optical flow database is that the objects move approximately linearly, there is no rotating object in the datasets. This is a very strict limitation as tracking is a challenging task mainly when the same texture is seen from different viewpoint.

It is interesting that the Middlebury multi-view database [25] contains ground truth 3D reconstruction of two objects, however, the ground truth tracking data were not generated for these sequences. Another limitation of the dataset is that only two low-textured objects are used.

It is obvious that tracking data can also be generated by a depth camera [26] such as Microsoft Kinect, but its accuracy is very limited. There are other interesting GT generators for planar objects such as the work proposed in [8], however, we would like to obtain the tracked feature points of real spatial objects.

Due to these limitations, we decided to build a special hardware in order to generate ground truth data. Our approach is based on a turntable, a camera, and a projector. They are not too costly, but the whole setup is very accurate as it is shown in our accepted paper [19].

2. Datasets

We have generated four GT datasets as it is published in our mentioned paper [19]. The feature points are always selected by the tested feature generator method in all frames and then these feature locations are matched between the frames. Then the matched point are filtered: the fundamental matrix [9] is robustly computed using 8-point method with RANSAC for every image pair and the outliers are removed from the results. The method implemented in the OpenCV framework is used for this robustification.

Examples for the moving GT feature points of the generated sets are visualized in Figures 1– 4. Point locations are visualized by light blue dots.

The feature matchers are tested in four data sequences:

- **Dinosaur**. A typical computer vision study deals with the reconstruction of a dinosaurs as it is shown in several scientific papers, e.g [6]. It has a simple diffuse surface that is easy to reconstruct in 3D, hence the feature matching is possible. For this reason, a dino is inserted to our testing dataset.

- **Flacon.** The plastic holder is another smooth and diffuse surface. A well-textured label is fixed on the surface.

- **Plush Dog.** The tracking of the feature point of a soft toy is a challenging task as it does not have a flat surface. A plush dog is included into the testing database that is a real challenge for feature trackers.

- **Poster.** The last sequence of our dataset is a rotating poster in a page of a motorcycle magazine. It is a relatively easy object for feature matchers since it is a well-textured plane. The pure efficiency of the trackers can be checked in this example due to two reasons: (i) there is no occlusion, and (ii) the GT feature tracking is equivalent to the determination of plane-plane homographies.

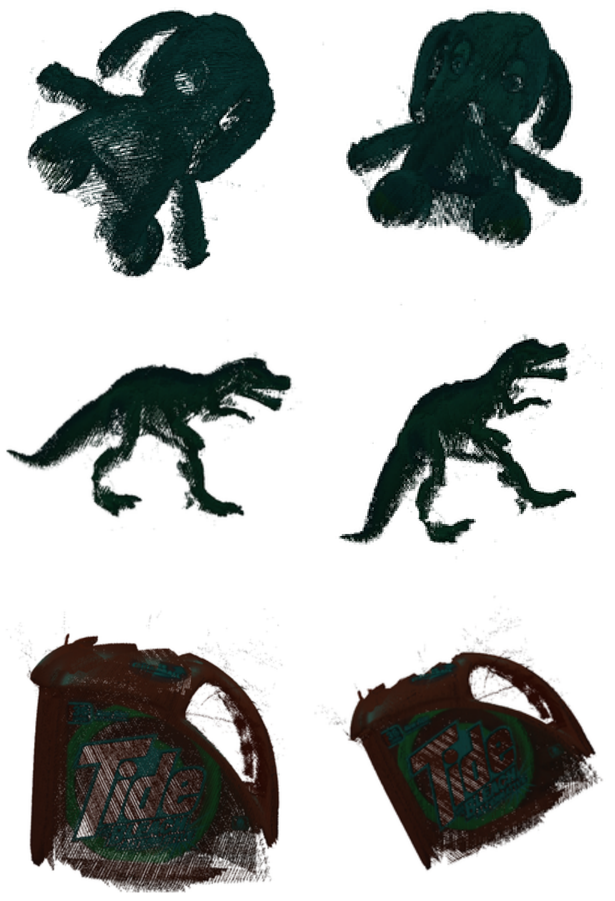


Figure 5. Reconstructed 3D model of testing objects. Top: Plush Dog. Center: Dinosaur. Bottom: Flacon.

2.1. GT Data Generation

Firstly, the possibilities is overviewed that OpenCV can give about feature tracking. These are the currently supported feature detectors in OpenCV AGAST [13], AKAZE [17], BRISK [10], FAST [20], GFTT [28] (Good Features To Track – also known as Shi-Tomasi corners), KAZE [2], MSER [14], ORB [21].

However, if you compile the contrib(nonfree) repository with the OpenCV, you can also get the

Figure 1. GT moving feature points of sequence 'Flacon'.

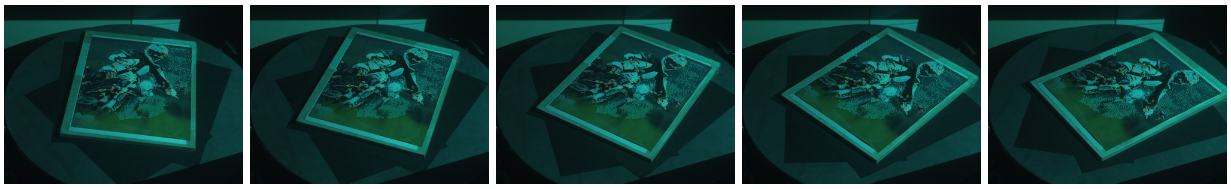


Figure 2. GT moving feature points of sequence 'Poster'.

following detectors: SIFT [12], STAR [1], and SURF [4].

We use our scanner to take 20 images about a rotating object. After each image taken, a structured light sequence is projected in order to make the reconstruction available for every position. (reconstructing only the points in the first image is not enough.)

Then we start searching for features in these images using all feature detectors. After the detection is completed, it is required to extract descriptors. Descriptors are needed for matching the feature points in different frames. The following descriptors are used (each can be found in OpenCV): AKAZE [17], BRISK [10], KAZE [2], ORB [21]. If one compiles the contrib repository, he/she can also get SIFT [12], SURF [4], BRIEF [5], FREAK [16], LATCH [11], DAISY [27] descriptors [2].

Another important issue is the parameterization of the feature trackers. It is obvious that the most accurate strategy is to find the best system parameters for the methods, nevertheless the optimal parameters can differ for each testing video. On the other hand, we think that the authors of the tested methods can set the parameters more accurately than us as they are interested in good performance. For this reason, the default parameter setting is used for each method, and we plan to make the dataset available for everyone and then the authors themselves can parameterize their methods.

After the detection and the extraction are done, the matching is started. Every image pair is taken into consideration, and match each feature point in the first image with one in the second image. This means that every feature point in the first image will have a pair in the second one. However, there can be some feature locations in the second image, which has more corresponding feature points in the first one, but it is also possible that there is no matching point.

The matching itself is done by calculating the minimum distances between the descriptor vectors. This distance is defined by the feature tracking method used. The following matchers are available in OpenCV:

- L_2 – *BruteForce*: a brute force minimization algorithm that computes each possible matches. The error is the L_2 norm of the difference between feature descriptors.
- L_1 – *BruteForce*: It is the same as L_2 – *BruteForce*, but L_1 norm is used instead of L_2 one.
- *Hamming* – *BruteForce*: For binary feature descriptor (BRISK, BRIEF, FREAK, LETCH,ORB,AKAZE), the Hamming distance is used.
- *Hamming2* – *BruteForce*: A variant of the hamming distance is used. The difference between Hamming and Hamming2 is that the former considers every bit as element of the vector, while Hamming2 use integer number, each bit pair forms a number from interval $0 \ldots 3$ [3].

[2]The BRIEF descriptor is not invariant to rotation, however, we hold it in the set of testing algorithms as it surprisingly served good results.

[3]OpenCV's documentation is not very informative about

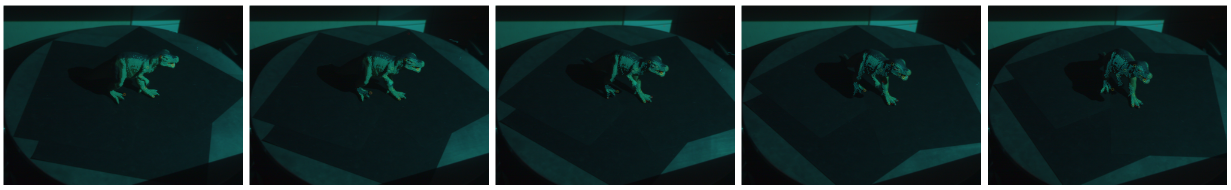


Figure 3. GT moving feature points of sequence 'Dinosaur'.

Figure 4. GT moving feature points of sequence 'Plush Dog'.

- *Flann-Based*: FLANN (Fast Library for Approximate Nearest Neighbors) is a set of algorithms optimized for fast nearest neighbor search in large datasets and for high dimensional features [15].

It is needed to point out that one can pair each feature detector with each feature descriptor but each feature matchers is not applicable for every descriptor. An exception is thrown by OpenCV if the selected algorithms cannot work together. But we try to evaluate every possible selection.

The comparison of the feature tracker predictions with the ground truth data is as follows: The feature points are reconstructed first in 3D using the images and the structured light. Then, because it is known that the turntable was rotated by 3 degrees per images, the projections of the points are calculated for all the remaining images. These projections were compared to the matched point locations of the feature trackers and the L_2 norm is used to calculate the distances.

3. Evaluation Methodology

The easiest and usual way for comparing the tracked feature points is to compute the summa and/or average and/or median of the 2D tracking errors in each image. This error is defined as the Euclidean distance of the tracked and GT locations. This methodology is visualized in Fig. 6.

Hamming2 distance. They suggest the usage of that for ORB features. However, it can be applied for other possible descriptors, all possible combinations are tried during our tests.

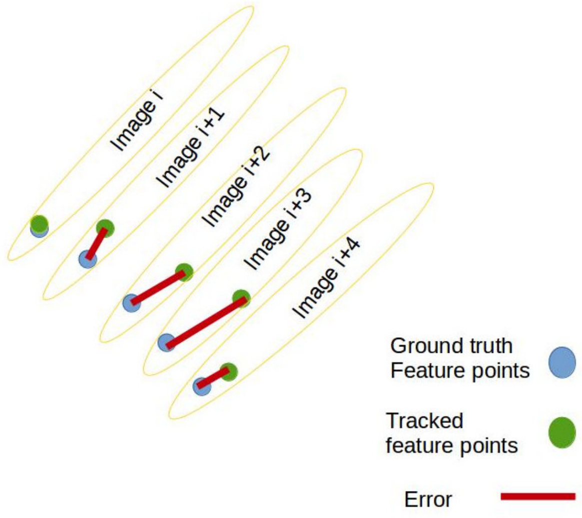


Figure 6. Error measurement based on simple Euclidean distances.

However, this comparison is not good enough because if a method fails to match correctly the feature points in an image pair, then the feature point moves to an incorrect location in the next image. Therefore, the tracker follows the incorrect location in the remaining frames and the new matching positions in those images will also be incorrect.

To avoid this effect, a new GT point is generated at the location of the matched point even if it is an incorrect matching. The GT location of that point can be determined in the remaining frames since that point can be reconstructed in 3D as well using the structured light scanning, and the novel positions of the new GT point can be determined using the calibration data of the test sequence.

Then the novel matching results are compared to all the previously determined GT points. The obtained error values are visualized in Fig. 7.

The error of a feature point for the i-th frame is the weighted average of all the errors calculated for that feature. For example, there is only one error value for the second frame as the matching error can only be compared to the GT location of the feature detected in the first image. For the third frame, there are two GT locations since GT error generated on both the first (original position) and second (position from first matching) image. For the i-th image, $i-1$ error values are obtained. the error is calculated as the weighted average of those. It can be formalized as

$$Error_{p_i} = \sum_{n=1}^{i-1} \frac{||p_i - p'_{i,n}||_2}{i-n} \tag{1}$$

where $Error_{p_i}$ is the error for the i-th frame, p_i the location of the tested feature detector, while $p'_{i,n}$ is the GT location of the feature points reconstructed from the n-th frame. The weights of the distances is $1/(i-n)$ that means that older GT points has less weights. Remark that the Euclidean (L_2) norm is chosen in order to measure the pixel distances.

If a feature point is only detected in one image and was not being followed in the next one (or was filtered out in the fundamental-matrix-based filtering step), then that point is discarded.

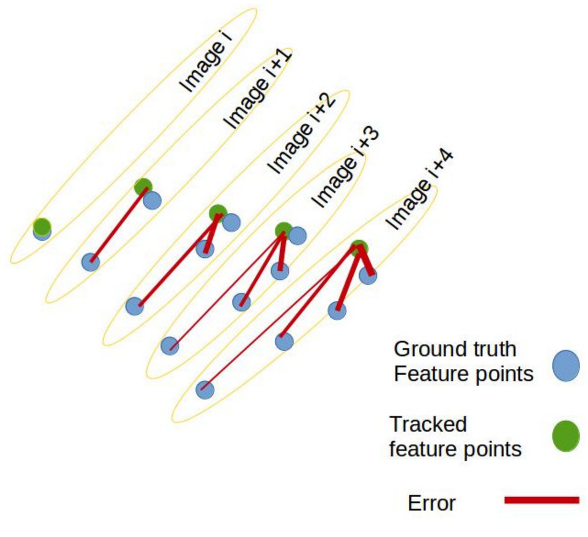


Figure 7. Applied error measurement.

After the pixel errors are valuated for each point in all possible images, the minimum, maximum, summa, average, and median error values of every feature points are calculated per image. The number of tracked feature points in the processed image is also counted. Furthermore, the average length of the feature tracks is calculated which shows that in how many images an average feature point is tracked through.

4. Comparison of the methods

The purpose of this section is to show the main issues occurred during the testing of the feature matchers. Unfortunately, we cannot show to the Reader all the charts due to the lack of space.

General remark. The charts in this section show different combinations of detectors, descriptors, and matchers. The method 'xxx:yyy:zzz' denotes in the charts that the current method uses the detector 'xxx', descriptor 'yyy', and matcher algorithm 'zzz'.

4.1. Feature Generation and Filtering using the Fundamental Matrix

The number of the detected feature points is examined first. It is an important property of the matcher algorithms since many good points are required for a typical computer vision application. For example, at least hundreds of points are required to compute 3D reconstruction of the observed scene. The matched and filtered values are calculated as the average of the numbers of generated features for all the frames as features can be independently generated in each image of the test sequences. Tables 1– 4 show the number of the generated features (left) and that of the filtered ones.

There are a few interesting behaviors within the data:

- The best images for feature tracking are obtained when the poster is rotated. The feature generators give significantly the most points in this case. It is a more challenging task to find goof feature points for the rotating dog and dinosaur. It is because the area of these objects in the images are smaller than that of the other two ones (flacon and poster).

- It is clearly seen that number of SURF feature points are the highest in all test cases after outlier removal. This fact suggests that they will be the more accurate features.

- The MSER method gives the most number of feature points, however, more than 90% of those are filtered. Unfortunately, the OpenCV3 library does not contain sophisticate matchers for

Table 1. Average of generated feature points and inliers of Sequence 'Plush Dog'.

Detector	#Features	#Inliers
BRISK	21.7	16.9
FAST	19.65	9.48
GFTT	1000	38.16
KAZE	68.6	40.76
MSER	**5321.1**	10.56
ORB	42.25	34.12
SIFT	67.7	42.8
STAR	7.15	5.97
SURF	514.05	**326.02**
AGAST	22.45	11.83
AKAZE	144	101.68

Table 2. Average of generated feature points and inliers of Sequence 'Poster'.

Detector	#Features	#Inliers
BRISK	233.55	188.79
FAST	224.75	139.22
GFTT	956.65	618.75
KAZE	573.45	469.18
MSER	**4863.6**	40.29
ORB	259.5	230.76
SIFT	413.35	343.08
STAR	41.25	35.22
SURF	1876.95	**1577.73**
AGAST	275.75	200.25
AKAZE	815	761.4

Table 3. Average of generated feature points and inliers of Sequence 'Flacon'.

Detector	#Features	#Inliers
BRISK	219.7	160.99
FAST	387.05	275.4
GFTT	1000	593.4
KAZE	484.1	387.93
MSER	**3664.1**	31.72
ORB	337.65	287.49
SIFT	348.15	260.91
STAR	69.1	54.86
SURF	952.95	**726.83**
AGAST	410.15	303.45
AKAZE	655	553.11

Table 4. Average of generated feature points and inliers of Sequence 'Dinosaur'.

Detector	#Features	#Inliers
BRISK	21.55	14.8
FAST	51.05	27.01
GFTT	1000	92
KAZE	58.55	33.92
MSER	**5144.4**	17.86
ORB	67.1	45.87
SIFT	52.8	34.96
STAR	3.45	3.45
SURF	276.95	**132.61**
AGAST	55	29.86
AKAZE	89.1	59.2

MSER such as [7], therefore its accuracy is relatively low.

- Remark that the GFTT algorithm usually gives 1000 points as the maximum number was set to thousand for this method. It is a parameter of OpenCV that may be changed, but we did not modify this value.

4.2. Matching accuracy

Two comparisons were carried out for the feature tracker methods. In the first test, every possible combination of the feature detectors and descriptors is examined, while the detectors are only combined with their own descriptor in the second test.

It is important to note that not only the errors of feature trackers should be compared, we must also pay attention to the number of features in the images and the length of the feature tracks. A method with less detected features usually obtains better results (lower error rate) than other methods with higher number of features. The mostly used chart is the AVG-MED, where the average and the median of the errors are shown.

Testing of all possible algorithms.

As it is seen in Fig 8 (sequence 'Plush Dog'), the SURF method dominates the chart. With the usage of SURF, DAISY, BRIEF, and BRISK descriptors more than 300 feature points remained and the median values of the errors are below 2.5 pixels, while the average is around 5 pixels. Moreover, the points are tracked through 4 images in average which yields pretty impressive statistics for the SURF detector.

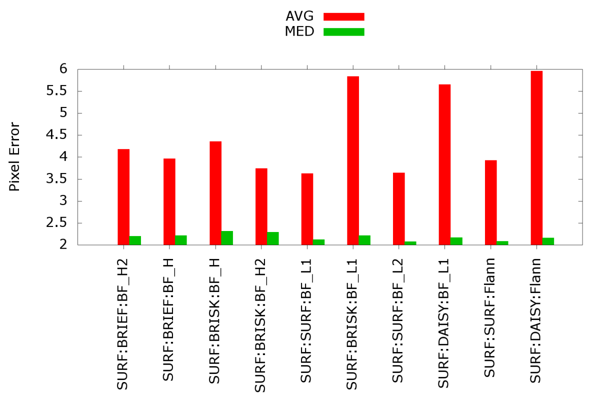


Figure 8. Average and median errors of top 10 methods for sequence 'Plush Dog'.

The next test object was the 'Poster'. The results are visualized in Fig 4.2. It is interesting to note that if the trackers are sorted by the number of the outliers and plot the top 10 methods, only the AKAZE detector remains where more than 90 percent of the feature points was considered as inlier. Besides the high number of points, average pixel error is between 3 and 5 pixels depending on the descriptor and matcher type.

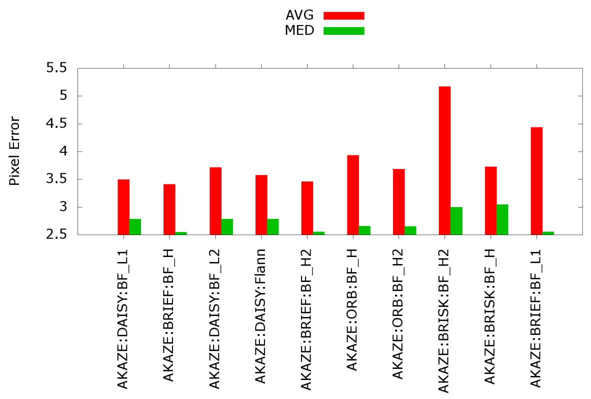
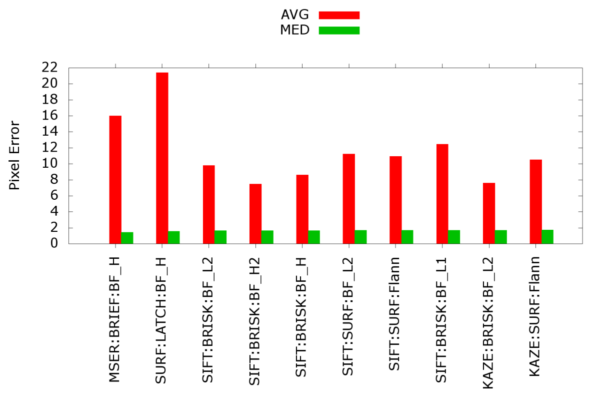


Figure 9. Average and median errors of top 10 methods for sequence 'Poster'.

In the test where the 'Flacon' object was used, we got similar results as in the case of 'Poster'. Both of the objects is rich in features, but the 'Flacon' is a spatial object. However, if we look at Fig. 10 where the methods with the lowest 10 median value were plotted, one can see that KAZE and SIFT had more feature points and can track these over more pictures than MSER or SURF after the fundamental filtering. Even though they had the lowest median values, the average errors of these methods were rather high.

However, if one takes a look at the methods with the lowest average error, then he/she can observe that AKAZE, KAZE and SURF present in the top 10. These methods can track more points then the previous ones and the median errors are just around 2.0 pixels.



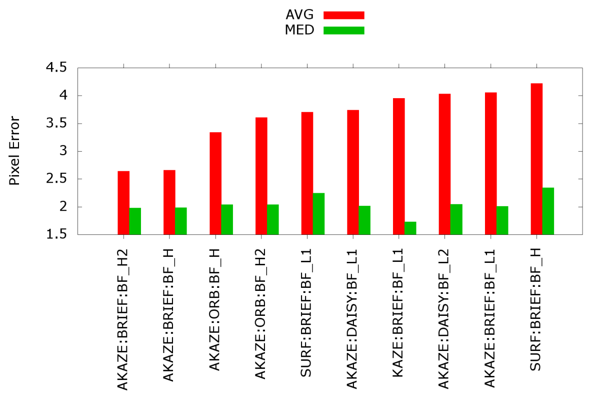


Figure 10. Top 10 method with the lowest median for sequence 'Flacon'. Chart are sorted by median (top) and average (bottom) values.

For the sequence 'Dinosaur' (Figure 11), the test object is very dark which makes feature detection hard. The number of available points is slightly more than 100. In this case, the overall winner of the methods is the SURF with both the lowest average and median errors. However, GFTT also present in the last chart too.

In the upper comparisons only the detectors were mentioned against each other. As one can see in the charts, most of the methods used either DAISY, BRIEF, BRISK or SURF descriptors. From the perspective of matchers, it does not really matter which type of the matcher is used for the same detector descriptor type. However, if the descriptor gives a binary vector, then obviously the hamming distance outperforms the L2 or L1. But there are just slightly differences between the L1-L2 and H1-H2 distances.

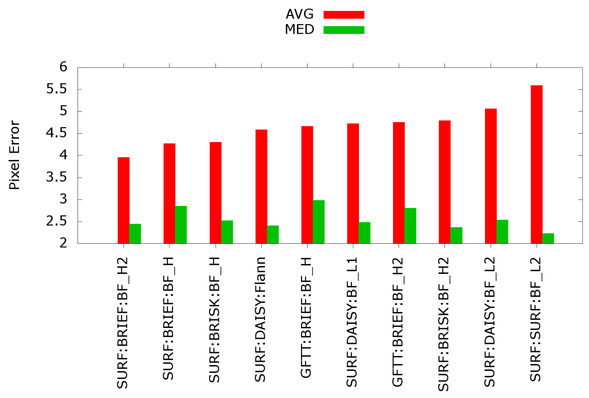


Figure 11. Top 10 methods (with lowest average error) on sequence 'Dinosaur'.

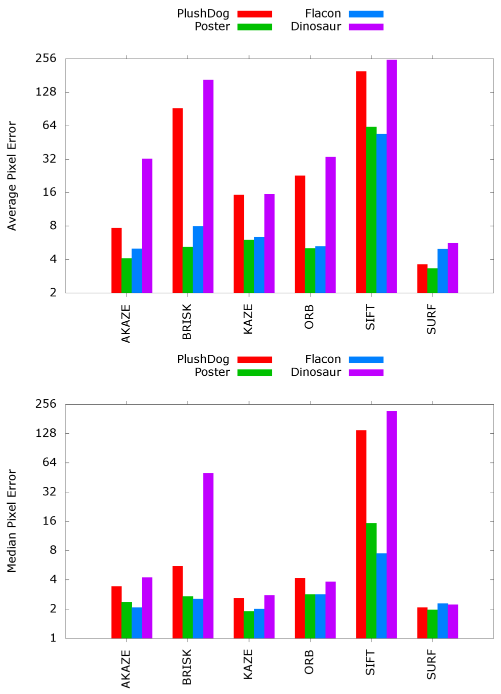


Figure 12. Overall average (top) and median (bottom) error values for all trackers and test sequences. The detectors and descriptors were the same.

Testing of algorithms with same detector and descriptor. In this comparison, only the detectors that have an own descriptor are tested. Always the best matchers is selected for which the error is minimal for the observed detector/descriptor.

As it can be seen in the log-scale charts in Fig. 12, the median error is almost the same for the AKAZE, KAZE, ORB and SURF trackers, but SURF is considered with the lowest average value. The tests 'Flacon' and 'Poster' result the lower pixel errors. On the other hand the rotation of the 'Dinosaur' was the hardest to track, it resulted much higher errors for all trackers comparing to the other tests.

5. Conclusions, Limitations, and Future Work

We quantitatively compared the well-known feature detectors, descriptors, and matchers implemented in OpenCV3 in this study. The GT datasets was generated by a structured-light scanner. The four testing objects were rotated by the turntable of our equipment. It seems to be clear that the most accurate feature for matching methods is the SURF [4] one proposed by Bay et al. It outperforms the other algorithms in all test cases. The other very accurate algorithms are KAZE [2]/AKAZE [17], they are the runner-up in our competition.

The most important conclusion for us is that such a comparison is a very hard task: for example, there are infinite number of possible error metrics; the quality is hardly influenced by the number of features, and so on. The main limitation here is that we can only test the methods in images of rotating objects. We are not sure that the same performance would be obtained if translating objects are observed. A possible solution to the extension of this paper is to compare the same methods on the Middlebury database and unify the obtained results for rotation and translation.

We hope that this paper is just the very first step of our research. We plan to generate more testing data, and more algorithms will also be involved into the tests. The GT dataset will be online, and an open-source testing system is also planned to be available soon [4].

References

[1] M. Agrawal and K. Konolige. Censure: Center surround extremas for realtime feature detection and matching. In *ECCV*, 2008. 3

[2] P. F. Alcantarilla, A. Bartoli, and A. J. Davison. Kaze features. In *ECCV (6)*, pages 214–227, 2012. 2, 3, 8

[4]See http://web.eee.sztaki.hu

[3] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, 2011. 1

[4] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, 2008. 3, 8

[5] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. In *Proceedings of the 11th European Conference on Computer Vision: Part IV*, pages 778–792, 2010. 3

[6] A. W. "Fitzgibbon, G. Cross, and A. Zisserman. "automatic 3D model construction for turn-table sequences". In *"3D Structure from Multiple Images of Large-Scale Environments, LNCS 1506"*, pages "155–170", "1998". 2

[7] P.-E. Forssn and D. G. Lowe. Shape descriptors for maximally stable extremal regions. In *ICCV*. IEEE, 2007. 6

[8] S. Gauglitz, T. Höllerer, and M. Turk. Evaluation of interest point detectors and feature descriptors for visual tracking. *International Journal of Computer Vision*, 94(3):335–360, 2011. 2

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

[10] S. Leutenegger, M. Chli, and R. Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 2548–2555, 2011. 2, 3

[11] G. Levi and T. Hassner. LATCH: learned arrangements of three patch codes. *CoRR*, 2015. 3

[12] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision*, ICCV '99, pages 1150–1157, 1999. 3

[13] E. Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger. Adaptive and generic corner detection based on the accelerated segment test. In *Proceedings of the 11th European Conference on Computer Vision: Part II*, pages 183–196, 2010. 2

[14] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proc. BMVC*, pages 36.1–36.10, 2002. 2

[15] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *In VISAPP International Conference on Computer Vision Theory and Applications*, pages 331–340, 2009. 4

[16] R. Ortiz. Freak: Fast retina keypoint. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 510–517, 2012. 3

[17] A. B. Pablo Alcantarilla (Georgia Institute of Technolog), Jesus Nuevo (TrueVision Solutions AU). Fast explicit diffusion for accelerated features in nonlinear scale spaces. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2013. 2, 3, 8

[18] C. J. Pal, J. J. Weinman, L. C. Tran, and D. Scharstein. On learning conditional random fields for stereo - exploring model structures and approximate inference. *International Journal of Computer Vision*, 99(3):319–337, 2012. 1

[19] Z. Pusztai and L. Hajder. A Turntable-based Approach for Ground Truth Tracking Data Generation . In *VISAPP 2016*, pages 498–509, 2016. 2

[20] E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. In *In Internation Conference on Computer Vision*, pages 1508–1515, 2005. 2

[21] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *International Conference on Computer Vision*, 2011. 2, 3

[22] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nesic, X. Wang, and P. Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *Pattern Recognition - 36th German Conference, GCPR 2014, Münster, Germany, September 2-5, 2014, Proceedings*, pages 31–42, 2014. 1

[23] D. Scharstein and R. Szeliski. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision*, 47:7–42, 2002. 1

[24] D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *CVPR (1)*, pages 195–202, 2003. 1

[25] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), 17-22 June 2006, New York, NY, USA*, pages 519–528, 2006. 1

[26] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. "a benchmark for the evaluation of rgb-d slam systems". In *"Proc. of the International Conference on Intelligent Robot Systems (IROS)"*, "2012". 2

[27] E. Tola, V. Lepetit, and P. Fua. Daisy: An efficient dense descriptor applied to wide baseline stereo. *IEEE TRANS. PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 32(5), 2010. 3

[28] Tomasi, C. and Shi, J. Good Features to Track. In *IEEE Conf. Computer Vision and Pattern Recognition*, pages 593–600, 1994. 2

21st Computer Vision Winter Workshop
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

Real-Time Eye Blink Detection using Facial Landmarks

Tereza Soukupová and Jan Čech
Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University in Prague
{soukuter,cechj}@cmp.felk.cvut.cz

Abstract. *A real-time algorithm to detect eye blinks in a video sequence from a standard camera is proposed. Recent landmark detectors, trained on in-the-wild datasets exhibit excellent robustness against a head orientation with respect to a camera, varying illumination and facial expressions. We show that the landmarks are detected precisely enough to reliably estimate the level of the eye opening. The proposed algorithm therefore estimates the landmark positions, extracts a single scalar quantity – eye aspect ratio (EAR) – characterizing the eye opening in each frame. Finally, an SVM classifier detects eye blinks as a pattern of EAR values in a short temporal window. The simple algorithm outperforms the state-of-the-art results on two standard datasets.*

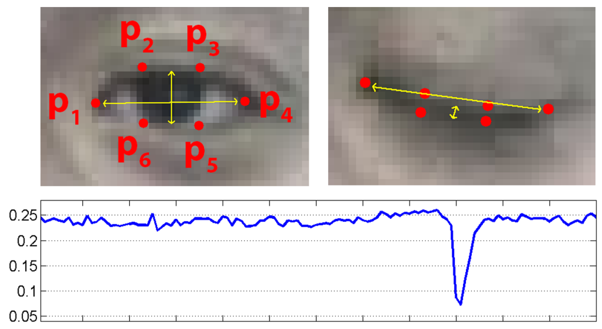


Figure 1: Open and closed eyes with landmarks p_i automatically detected by [1]. The eye aspect ratio EAR in Eq. (1) plotted for several frames of a video sequence. A single blink is present.

1. Introduction

Detecting eye blinks is important for instance in systems that monitor a human operator vigilance, e.g. driver drowsiness [5, 13], in systems that warn a computer user staring at the screen without blinking for a long time to prevent the dry eye and the computer vision syndromes [17, 7, 8], in human-computer interfaces that ease communication for disabled people [15], or for anti-spoofing protection in face recognition systems [11].

Existing methods are either active or passive. Active methods are reliable but use special hardware, often expensive and intrusive, e.g. infrared cameras and illuminators [2], wearable devices, glasses with a special close-up cameras observing the eyes [10]. While the passive systems rely on a standard remote camera only.

Many methods have been proposed to automatically detect eye blinks in a video sequence. Several methods are based on a *motion estimation* in the eye region. Typically, the face and eyes are detected by a Viola-Jones type detector. Next, motion in the eye area is estimated from optical flow, by sparse tracking [7, 8], or by frame-to-frame intensity differencing and adaptive thresholding. Finally, a decision is made whether the eyes are or are not covered by eyelids [9, 15]. A different approach is to infer the *state of the eye opening from a single image*, as e.g. by correlation matching with open and closed eye templates [4], a heuristic horizontal or vertical image intensity projection over the eye region [5, 6], a parametric model fitting to find the eyelids [18], or active shape models [14].

A major drawback of the previous approaches is that they usually implicitly impose too strong requirements on the setup, in the sense of a relative face-camera pose (head orientation), image resolution, illumination, motion dynamics, etc. Especially the heuristic methods that use raw image intensity are likely to be very sensitive despite their real-time performance.

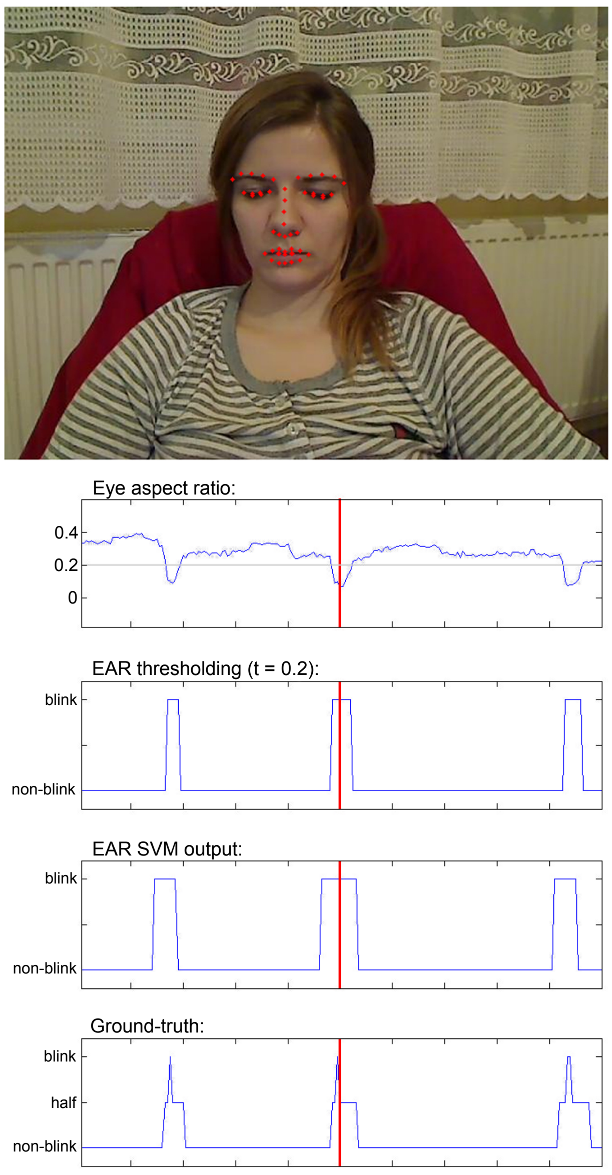


Figure 2: Example of detected blinks. The plots of the eye aspect ratio EAR in Eq. (1), results of the EAR thresholding (threshold set to 0.2), the blinks detected by EAR SVM and the ground-truth labels over the video sequence. Input image with detected landmarks (depicted frame is marked by a red line).

However nowadays, robust real-time *facial landmark detectors* that capture most of the characteristic points on a human face image, including eye corners and eyelids, are available, see Fig. 1. Most of the state-of-the-art landmark detectors formulate a regression problem, where a mapping from an image into landmark positions [16] or into other landmark parametrization [1] is learned. These modern landmark detectors are trained on "in-the-wild datasets" and they are thus robust to varying illumination, various facial expressions, and moderate non-frontal head rotations. An average error of the landmark localization of a state-of-the-art detector is usually below five percent of the inter-ocular distance. Recent methods run even significantly super real-time [12].

Therefore, we propose a simple but efficient algorithm to detect eye blinks by using a recent facial landmark detector. A single scalar quantity that reflects a level of the eye opening is derived from the landmarks. Finally, having a per-frame sequence of the eye opening estimates, the eye blinks are found by an SVM classifier that is trained on examples of blinking and non-blinking patterns.

Facial segmentation model presented in [14] is similar to the proposed method. However, their system is based on active shape models with reported processing time of about 5 seconds per frame for the segmentation, and the eye opening signal is normalized by statistics estimated by observing a longer sequence. The system is thus usable for offline processing only. The proposed algorithm runs real-time, since the extra costs of the eye opening from landmarks and the linear SVM are negligible.

The contributions of the paper are:

1. Ability of two state-of-the-art landmark detectors [1, 16] to reliably distinguish between the open and closed eye states is quantitatively demonstrated on a challenging in-the-wild dataset and for various face image resolutions.

2. A novel real-time eye blink detection algorithm which integrates a landmark detector and a classifier is proposed. The evaluation is done on two standard datasets [11, 8] achieving state-of-the-art results.

The rest of the paper is structured as follows: The algorithm is detailed in Sec. 2, experimental validation and evaluation is presented in Sec. 3. Finally, Sec. 4 concludes the paper.

2. Proposed method

The eye blink is a fast closing and reopening of a human eye. Each individual has a little bit different pattern of blinks. The pattern differs in the speed of closing and opening, a degree of squeezing the eye and in a blink duration. The eye blink lasts approximately 100-400 ms.

We propose to exploit state-of-the-art facial landmark detectors to localize the eyes and eyelid contours. From the landmarks detected in the image, we derive the eye aspect ratio (EAR) that is used as an estimate of the eye opening state. Since the per-frame EAR may not necessarily recognize the eye blinks correctly, a classifier that takes a larger temporal window of a frame into account is trained.

2.1. Description of features

For every video frame, the eye landmarks are detected. The eye aspect ratio (EAR) between height and width of the eye is computed.

$$\text{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}, \qquad (1)$$

where $p_1, \ldots, p_6$ are the 2D landmark locations, depicted in Fig. 1.

The EAR is mostly constant when an eye is open and is getting close to zero while closing an eye. It is partially person and head pose insensitive. Aspect ratio of the open eye has a small variance among individuals and it is fully invariant to a uniform scaling of the image and in-plane rotation of the face. Since eye blinking is performed by both eyes synchronously, the EAR of both eyes is averaged. An example of an EAR signal over the video sequence is shown in Fig. 1, 2, 7.

A similar feature to measure the eye opening was suggested in [9], but it was derived from the eye segmentation in a binary image.

2.2. Classification

It generally does not hold that low value of the EAR means that a person is blinking. A low value of the EAR may occur when a subject closes his/her eyes intentionally for a longer time or performs a facial expression, yawning, etc., or the EAR captures a short random fluctuation of the landmarks.

Therefore, we propose a classifier that takes a larger temporal window of a frame as an input. For the 30fps videos, we experimentally found that ± 6 frames can have a significant impact on a blink detection for a frame where an eye is the most closed when blinking. Thus, for each frame, a 13-dimensional feature is gathered by concatenating the EARs of its ± 6 neighboring frames.

This is implemented by a linear SVM classifier (called EAR SVM) trained from manually annotated sequences. Positive examples are collected as ground-truth blinks, while the negatives are those that are sampled from parts of the videos where no blink occurs, with 5 frames spacing and 7 frames margin from the ground-truth blinks. While testing, a classifier is executed in a scanning-window fashion. A 13-dimensional feature is computed and classified by EAR SVM for each frame except the beginning and ending of a video sequence.

3. Experiments

Two types of experiments were carried out: The experiments that measure accuracy of the landmark detectors, see Sec. 3.1, and the experiments that evaluate performance of the whole eye blink detection algorithm, see Sec 3.2.

3.1. Accuracy of landmark detectors

To evaluate accuracy of tested landmark detectors, we used the 300-VW dataset [19]. It is a dataset containing 50 videos where each frame has associated a precise annotation of facial landmarks. The videos are "in-the-wild", mostly recorded from a TV.

The purpose of the following tests is to demonstrate that recent landmark detectors are particularly robust and precise in detecting eyes, i.e. the eye-corners and contour of the eyelids. Therefore we prepared a dataset, a subset of the 300-VW, containing sample images with both open and closed eyes. More precisely, having the ground-truth landmark annotation, we sorted the frames for each subject by the eye aspect ratio (EAR in Eq. (1)) and took 10 frames of the highest ratio (eyes wide open), 10 frames of the lowest ratio (mostly eyes tightly shut) and 10 frames sampled randomly. Thus we collected 1500 images. Moreover, all the images were later subsampled (successively 10 times by factor 0.75) in order to evaluate accuracy of tested detectors on small face images.

Two state-of-the-art landmark detectors were tested: Chehra [1] and Intraface [16]. Both run in real-time[1]. Samples from the dataset are shown in Fig. 3. Notice that faces are not always frontal to the camera, the expression is not always neutral, people are often emotionally speaking or smiling, etc. Sometimes people wear glasses, hair may occasionally partially occlude one of the eyes. Both detectors perform generally well, but the Intraface is more robust to very small face images, sometimes at impressive extent as shown in Fig. 3.

[1]Intraface runs in 50 Hz on a standard laptop.

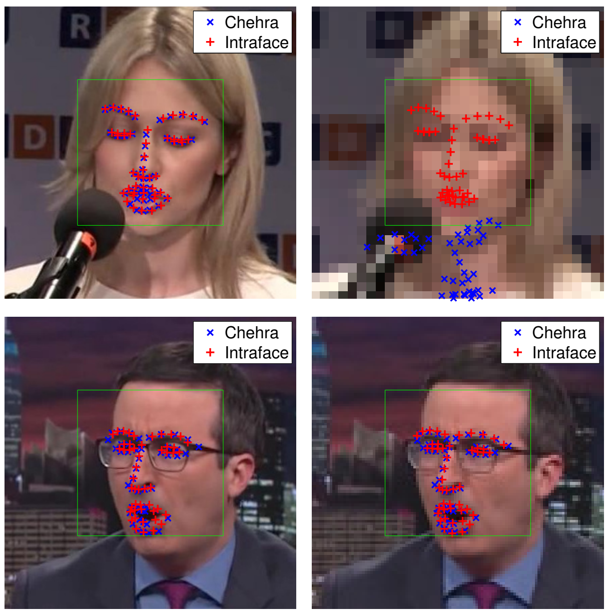


Figure 3: Example images from the 300-VW dataset with landmarks obtained by Chehra [1] and Intraface [16]. Original images (left) with inter-ocular distance (IOD) equal to 63 (top) and 53 (bottom) pixels. Images subsampled (right) to IOD equal to 6.3 (top) and 17 (bottom).

Quantitatively, the accuracy of the landmark detection for a face image is measured by the average relative landmark localization error, defined as usually

$$\epsilon = \frac{100}{\kappa N} \sum_{i=1}^{N} ||x_i - \hat{x}_i||_2, \tag{2}$$

where x_i is the ground-truth location of landmark i in the image, $\hat{x}_i$ is an estimated landmark location by a detector, N is a number of landmarks and normalization factor κ is the inter-ocular distance (IOD), i.e. Euclidean distance between eye centers in the image.

First, a standard cumulative histogram of the average relative landmark localization error ϵ was calculated, see Fig. 4, for a complete set of 49 landmarks and also for a subset of 12 landmarks of the eyes only, since these landmarks are used in the proposed eye blink detector. The results are calculated for all the original images that have average IOD around 80 px, and also for all "small" face images (including subsampled ones) having IOD $\leq$ 50 px. For all landmarks, Chehra has more occurrences of very small errors (up to 5 percent of the IOD), but Intraface is more robust having more occurrences of errors below 10 percent of the IOD. For eye landmarks only,

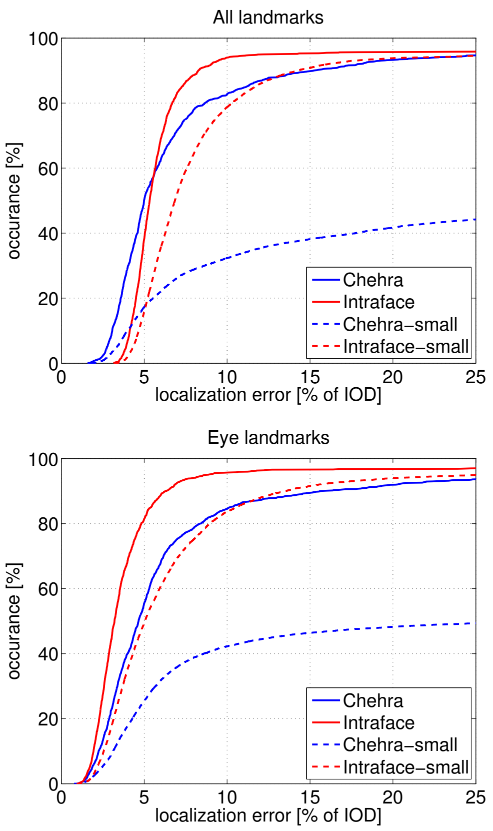


Figure 4: Cumulative histogram of average localization error of all 49 landmarks (top) and 12 landmarks of the eyes (bottom). The histograms are computed for original resolution images (solid lines) and a subset of small images (IOD $\leq$ 50 px).

the Intraface is always more precise than Chehra. As already mentioned, the Intraface is much more robust to small images than Chehra. This behaviour is further observed in the following experiment.

Taking a set of all 15k images, we measured a mean localization error μ as a function of a face image resolution determined by the IOD. More precisely, $\mu = \frac{1}{|S|} \sum_{j \in S} \epsilon_j$, i.e. average error over set of face images S having the IOD in a given range. Results are shown in Fig. 5. Plots have errorbars of standard deviation. It is seen that Chehra fails quickly for images with IOD $<$ 20 px. For larger faces, the mean error is comparable, although slightly better for Intraface for the eye landmarks.

The last test is directly related to the eye blink detector. We measured accuracy of EAR as a function of the IOD. Mean EAR error is defined as a mean absolute difference between the true and the estimated EAR. The plots are computed for two subsets: closed/closing (average true ratio 0.05 ± 0.05)

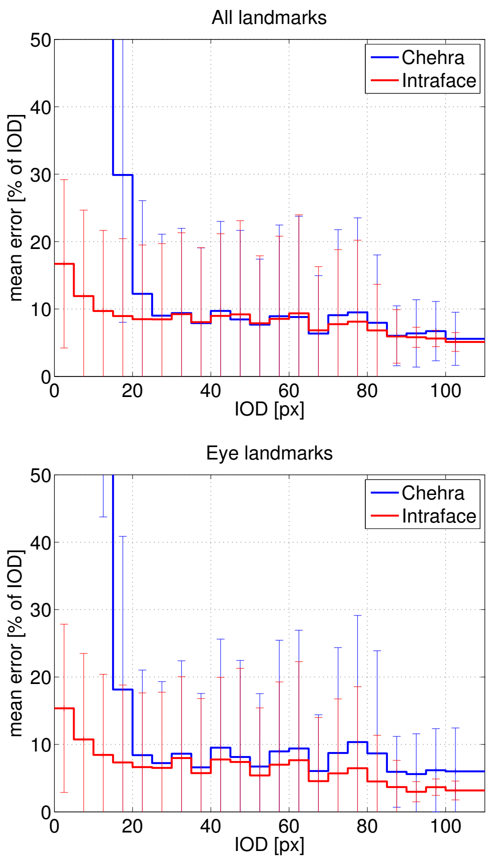


Figure 5: Landmark localization accuracy as a function of the face image resolution computed for all landmarks and eye landmarks only.

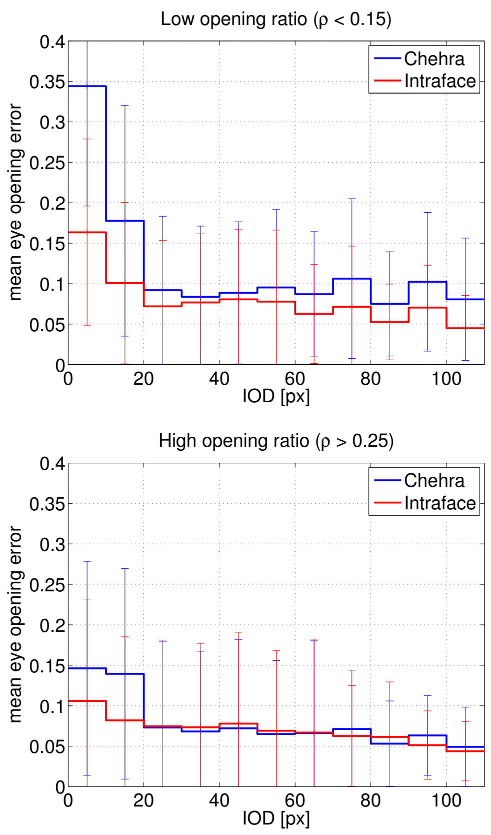


Figure 6: Accuracy of the eye-opening ratio as a function of the face image resolution. Top: for images with small true ratio (mostly closing/closed eyes), and bottom: images with higher ratio (open eyes).

and open eyes (average true ratio 0.4 ± 0.1). The error is higher for closed eyes. The reason is probably that both detectors are more likely to output open eyes in case of a failure. It is seen that ratio error for IOD < 20 px causes a major confusion between open/close eye states for Chehra, nevertheless for larger faces the ratio is estimated precisely enough to ensure a reliable eye blink detection.

3.2. Eye blink detector evaluation

We evaluate on two standard databases with ground-truth annotations of blinks. The first one is ZJU [11] consisting of 80 short videos of 20 subjects. Each subject has 4 videos: 2 with and 2 without glasses, 3 videos are frontal and 1 is an upward view. The 30fps videos are of size 320×240 px. An average video length is 136 frames and contains about 3.6 blinks in average. An average IOD is 57.4 pixels. In this database, subjects do not perform any noticeable facial expressions. They look straight into the camera at close distance, almost do not move, do not either smile nor speak. A ground-truth blink is defined by its beginning frame, peak frame and ending frame. The second database Eyeblink8 [8] is more challenging. It consists of 8 long videos of 4 subjects that are smiling, rotating head naturally, covering face with hands, yawning, drinking and looking down probably on a keyboard. These videos have length from 5k to 11k frames, also 30fps, with a resolution 640×480 pixels and an average IOD 62.9 pixels. They contain about 50 blinks on average per video. Each frame belonging to a blink is annotated by half-open or close state of the eyes. We consider half blinks, which do not achieve the close state, as full blinks to be consistent with the ZJU.

Besides testing the proposed EAR SVM methods, that are trained to detect the specific blink pattern, we compare with a simple baseline method, which only thresholds the EAR in Eq. (1) values. The EAR SVM classifiers are tested with both landmark detectors Chehra [1] and Intraface [16].

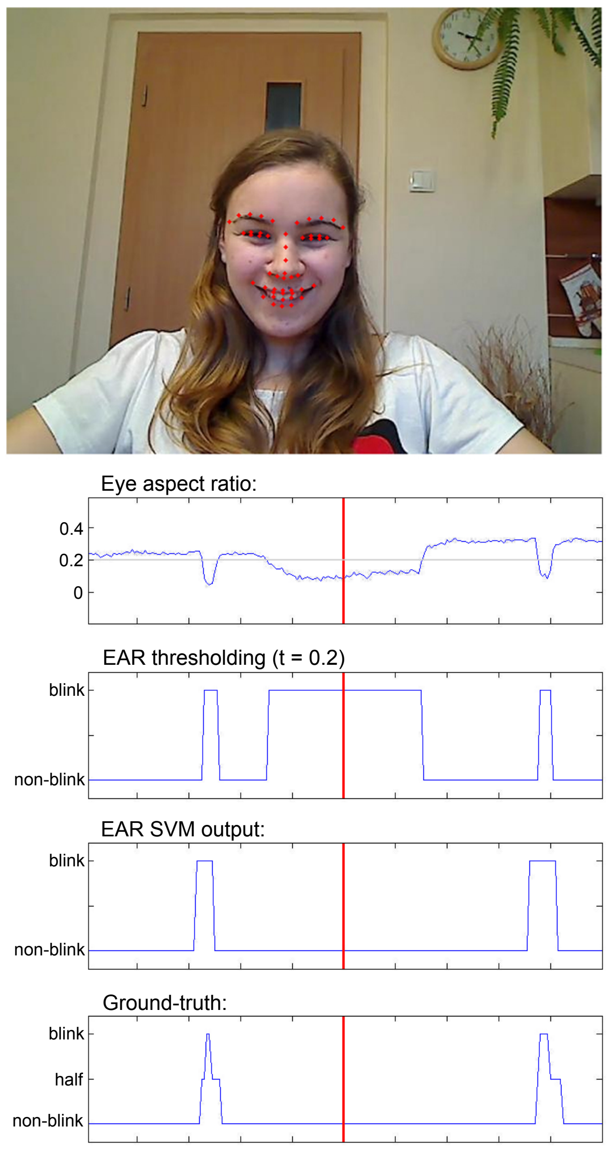


Figure 7: Example of detected blinks where the EAR thresholding fails while EAR SVM succeeds. The plots of the eye aspect ratio EAR in Eq. (1), results of the EAR thresholding (threshold set to 0.2), the blinks detected by EAR SVM and the ground-truth labels over the video sequence. Input image with detected landmarks (depicted frame is marked by a red line).

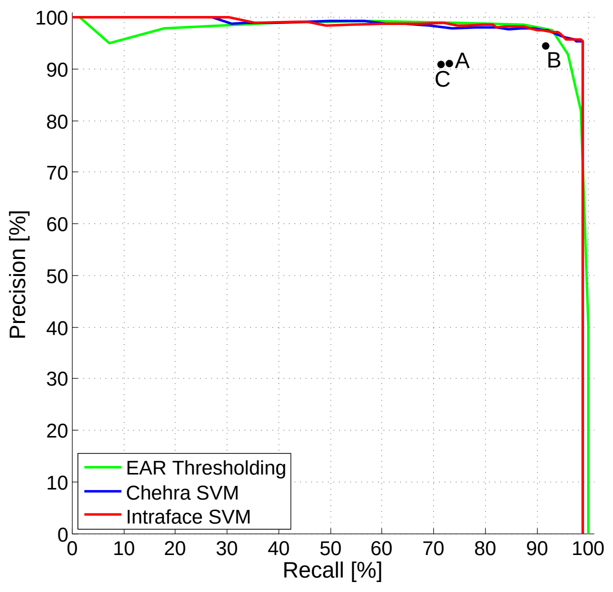
The experiment with EAR SVM is done in a cross-dataset fashion. It means that the SVM classifier is trained on the Eyeblink8 and tested on the ZJU and vice versa.

To evaluate detector accuracy, predicted blinks are compared with the ground-truth blinks. The number of true positives is determined as a number of the ground-truth blinks which have a non-empty intersection with detected blinks. The number of false negatives is counted as a number of the ground-truth blinks which do not intersect detected blinks. The number of false positives is equal to the number of detected blinks minus the number of true positives plus a penalty for detecting too long blinks. The penalty is counted only for detecting blinks twice longer then an average blink of length A. Every long blink of length L is counted $\frac{L}{A}$ times as a false positive. The number of all possibly detectable blinks is computed as number of frames of a video sequence divided by subject average blink length following Drutarovsky and Fogelton [8].

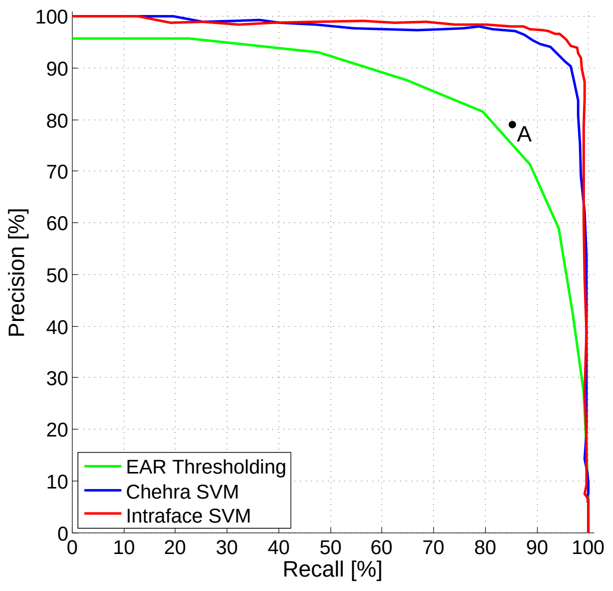
The ZJU database appears relatively easy. It mostly holds that every eye closing is an eye blink. Consequently, the precision-recall curves shown in Fig. 8a of the EAR thresholding and both EAR SVM classifiers are almost identical. These curves were calculated by spanning a threshold of the EAR and SVM output score respectively. All our methods outperform other detectors [9, 8, 5]. The published methods presented the precision and the recall for a single operation point only, not the precision-recall curve. See Fig. 8a for comparison.

The precision-recall curves in Fig. 8b shows evaluation on the Eyeblink8 database. We observe that in this challenging database the EAR thresholding lags behind both EAR SVM classifiers. The thresholding fails when a subject smiles (has narrowed eyes - see an example in Fig. 7), has a side view or when the subject closes his/her eyes for a time longer than a blink duration. Both SVM detectors performs much better, the Intraface detector based SVM is even a little better than the Chehra SVM. Both EAR SVM detectors outperform the method by Drutarovsky and Fogelton [8] by a significant margin.

Finally, we measured a dependence of the whole blink detector accuracy on the average IOD over the dataset. Every frame of the ZJU database was subsampled to 90%, 80%, ..., 10% of its original resolution. Both Chehra-SVM and Intraface-SVM were used for evaluation. For each resolution, the area under the precision-recall curve (AUC) was computed. The result is shown in Fig. 9. We can see that with Chehra landmarks the accuracy remains very high until average IOD is about 30 px. The detector fails on images with the IOD $<$ 20 px. Intraface landmarks are much better in low resolutions. This confirms our previous study on the accuracy of landmarks in Sec. 3.1.



(a) ZJU



(b) Eyeblink8

Figure 8: Precision-recall curves of the EAR thresholding and EAR SVM classifiers measured on (a) the ZJU and (b) the Eyeblink8 databases. Published results of methods A - Drutarovsky and Fogelton [8], B - Lee et al. [9], C - Danisman et al. [5] are depicted.

4. Conclusion

A real-time eye blink detection algorithm was presented. We quantitatively demonstrated that regression-based facial landmark detectors are precise enough to reliably estimate a level of eye openness. While they are robust to low image quality (low image resolution in a large extent) and in-the-wild phenomena as non-frontality, bad illumination, facial expressions, etc.

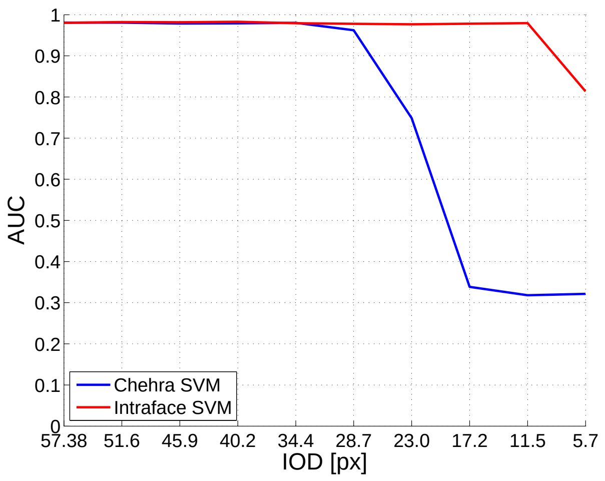


Figure 9: Accuracy of the eye blink detector (measured by AUC) as a function of the image resolution (average IOD) when subsampling the ZJU dataset.

State-of-the-art on two standard datasets was achieved using the robust landmark detector followed by a simple eye blink detection based on the SVM. The algorithm runs in real-time, since the additional computational costs for the eye blink detection are negligible besides the real-time landmark detectors.

The proposed SVM method that uses a temporal window of the eye aspect ratio (EAR), outperforms the EAR thresholding. On the other hand, the thresholding is usable as a single image classifier to detect the eye state, in case that a longer sequence is not available.

We see a limitation that a fixed blink duration for all subjects was assumed, although everyone's blink lasts differently. The results could be improved by an adaptive approach. Another limitation is in the eye opening estimate. While EAR is estimated from a 2D image, it is fairly insensitive to a head orientation, but may lose discriminability for out of plane rotations. A solution might be to define the EAR in 3D. There are landmark detectors that estimate a 3D pose (position and orientation) of a 3D model of landmarks, e.g. [1, 3].

Acknowledgment

The research was supported by CTU student grant SGS15/155/OHK3/2T/13.

References

[1] A. Asthana, S. Zafeoriou, S. Cheng, and M. Pantic. Incremental face alignment in the wild. In *Conference on Computer Vision and Pattern Recognition*, 2014. 1, 2, 3, 4, 5, 7

[2] L. M. Bergasa, J. Nuevo, M. A. Sotelo, and M. Vazquez. Real-time system for monitoring driver vigilance. In *IEEE Intelligent Vehicles Symposium*, 2004. 1

[3] J. Cech, V. Franc, and J. Matas. A 3D approach to facial landmarks: Detection, refinement, and tracking. In *Proc. International Conference on Pattern Recognition*, 2014. 7

[4] M. Chau and M. Betke. Real time eye tracking and blink detection with USB cameras. Technical Report 2005-12, Boston University Computer Science, May 2005. 1

[5] T. Danisman, I. Bilasco, C. Djeraba, and N. Ihaddadene. Drowsy driver detection system using eye blink patterns. In *Machine and Web Intelligence (ICMWI)*, Oct 2010. 1, 6, 7

[6] H. Dinh, E. Jovanov, and R. Adhami. Eye blink detection using intensity vertical projection. In *International Multi-Conference on Engineering and Technological Innovation, IMETI 2012*. 1

[7] M. Divjak and H. Bischof. Eye blink based fatigue detection for prevention of computer vision syndrome. In *IAPR Conference on Machine Vision Applications*, 2009. 1

[8] T. Drutarovsky and A. Fogelton. Eye blink detection using variance of motion vectors. In *Computer Vision - ECCV Workshops*. 2014. 1, 2, 5, 6, 7

[9] W. H. Lee, E. C. Lee, and K. E. Park. Blink detection robust to various facial poses. *Journal of Neuroscience Methods*, Nov. 2010. 1, 3, 6, 7

[10] Medicton group. The system I4Control. `http://www.i4tracking.cz/`. 1

[11] G. Pan, L. Sun, Z. Wu, and S. Lao. Eyeblink-based anti-spoofing in face recognition from a generic webcamera. In *ICCV*, 2007. 1, 2, 5

[12] S. Ren, X. Cao, Y. Wei, and J. Sun. Face alignment at 3000 fps via regressing local binary features. In *Proc. CVPR*, 2014. 2

[13] A. Sahayadhas, K. Sundaraj, and M. Murugappan. Detecting driver drowsiness based on sensors: A review. *MDPI open access: sensors*, 2012. 1

[14] F. M. Sukno, S.-K. Pavani, C. Butakoff, and A. F. Frangi. Automatic assessment of eye blinking patterns through statistical shape models. In *ICVS*, 2009. 1, 2

[15] D. Torricelli, M. Goffredo, S. Conforto, and M. Schmid. An adaptive blink detector to initialize and update a view-basedremote eye gaze tracking system in a natural scenario. *Pattern Recogn. Lett.*, 30(12):1144–1150, Sept. 2009. 1

[16] X. Xiong and F. De la Torre. Supervised descent methods and its applications to face alignment. In *Proc. CVPR*, 2013. 2, 3, 4, 5

[17] Z. Yan, L. Hu, H. Chen, and F. Lu. Computer vision syndrome: A widely spreading but largely unknown epidemic among computer users. *Computers in Human Behaviour*, (24):2026–2042, 2008. 1

[18] F. Yang, X. Yu, J. Huang, P. Yang, and D. Metaxas. Robust eyelid tracking for fatigue detection. In *ICIP*, 2012. 1

[19] S. Zafeiriou, G. Tzimiropoulos, and M. Pantic. The 300 videos in the wild (300-VW) facial landmark tracking in-the-wild challenge. In *ICCV Workshop*, 2015. `http://ibug.doc.ic.ac.uk/resources/300-VW/`. 3

21^{st} *Computer Vision Winter Workshop*
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

Solving Dense Image Matching in Real-Time using Discrete-Continuous Optimization

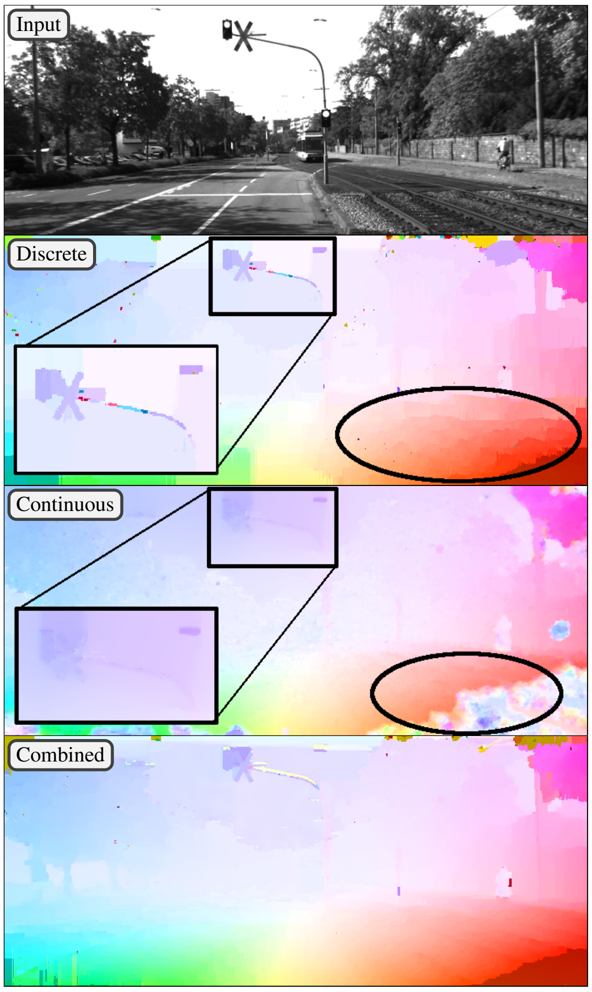
Alexander Shekhovtsov, Christian Reinbacher, Gottfried Graber and Thomas Pock
Institute for Computer Graphics and Vision, Graz University of Technology
`{shekhovtsov,reinbacher,graber,pock}@icg.tugraz.at`

Abstract. *Dense image matching is a fundamental low-level problem in Computer Vision, which has received tremendous attention from both discrete and continuous optimization communities. The goal of this paper is to combine the advantages of discrete and continuous optimization in a coherent framework. We devise a model based on energy minimization, to be optimized by both discrete and continuous algorithms in a consistent way. In the discrete setting, we propose a novel optimization algorithm that can be massively parallelized. In the continuous setting we tackle the problem of non-convex regularizers by a formulation based on differences of convex functions. The resulting hybrid discrete-continuous algorithm can be efficiently accelerated by modern GPUs and we demonstrate its real-time performance for the applications of dense stereo matching and optical flow.*

1. Introduction

The *dense image matching* problem is one of the most basic problems in computer vision: The goal is to find matching pixels in two (or more) images. The applications include stereo, optical flow, medical image registration, face recognition [1], *etc.* Since the matching problem is inherently ill-posed, typically *optimization* is involved in solving it. We can distinguish two fundamentally different approaches: discrete and continuous optimization. Whereas discrete approaches (see [14] for a recent comparison) assign a distinct label to each output pixel, continuous approaches try to solve for a *function* using the calculus of variations [6, 8, 21]. Both approaches have received enormous attention, and there exist state-of-the-art algorithms in both camps: continuous [23, 24, 28] and discrete [18, 30]. Due to the specific mathematical tools available to solve the problems (discrete combinatorial optimization vs. continuous calculus of variations), both approaches have distinct advantages and disadvantages.

In this paper, we argue that on a fundamental level the advantages and disadvantages of discrete and continuous optimization for dense matching problems are *complementary* as summarized in Figure 1. The previous work combining discrete and continuous optimization primarily used discrete optimization to fuse (find the optimal



	data term	Large motion	Parallelization
Discrete	Arbitrary (sampled)	Easy	Difficult
Continuous	Convex (linearized)	Difficult	Easy

Figure 1: Optical flow problem solved by a purely discrete method, a purely continuous method and the combined method. All methods are as described in this paper, they use the same data term and are run until convergence here. In the discrete solution we can see small scale details and sharp motion boundaries but also discretization artifacts. The continuous solution exhibits sub-pixel accuracy (smoothness), but lacks small details and has difficulties with large motions. The combined solution delivers smooth flow fields while retaining many small scale details.

crossover) of candidate continuous proposals, *e.g.* [36, 30] (stereo) and [25] (flow). The latter additionally performs local continuous optimization of the so-found solution. Many works also alternate between continuous and discrete optimizations, addressing a Mumford-Shah-like model, *e.g.*, [5]. Similarly to [25] we introduce a continuous energy which is optimized using a combined method. However, we work with a full (non-local) discretization of this model and propose new parallel optimization methods.

The basic difference in discrete and continuous approaches lies in the handling of the data term. The data term is a measure how well the solution (*i.e.* value of a pixel) fits the underlying measurement (*i.e.* input images). In the discrete setting, the solution takes discrete labels, and hence the number of labels is finite. Typically the data cost is precomputed for all possible labels. The discrete optimization then uses the data cost to find the optimal label for each pixel according to a suitable model in an energy minimization framework. We point out that due to the *sampling* in both label space and spatial domain, the discrete algorithm has access to the full information at every step. *I.e.* it deals with a *global optimization* model and in some lucky cases can find a globally optimal solution to it or provide an approximation ratio or partial optimality guarantees [27].

In the continuous setting, the solution is a continuous function. This means it is not possible to precompute the data cost; an infinite number of solutions would require infinite amount of memory. More importantly, the data cost is a non-convex function stemming from the similarity measure between the images. In order to make the optimization problem tractable, a popular approach is the linearization of the data cost. However, this introduces a range of new problems, namely the inability to deal with large motions due to the fact that the linearization is valid only in a small neighborhood around the linearization point. Most continuous methods relying on linearization therefore use a coarse-to-fine framework in an attempt to overcome this problem [4]. One exception is a recent work [16], which can handle piece-wise linear data terms and truncated TV regularization.

Our goal in this paper is to combine the advantages of both approaches, as well as real-time performance, which imposes tough constraints on both methods resulting in a number of challenges:

Challenges The discrete optimization method needs to be highly parallel and able to *couple* the noisy / ambiguous data over large areas. The continuous energy should be a refinement of the discrete energy so that we can evaluate the two-phase optimization in terms of a single energy function. The continuous method needs to handle robust (truncated) regularization terms.

Contribution Towards the posed challenges, we propose: i) a new method for the discrete problem, working in the dual (*i.e.* making equivalent changes of the data cost volume), in parallel on multiple chains; ii) a continuous optimization method, reducing non-convex regularizers to a primal-dual method with non-linear operators [31]; iii) an efficient implementation of both methods on GPU and proof of concept experiments showing advantages of the combined approach.

2. Method

In this section we will describe our two-step approach to the dense image matching problem. To combine the previously discussed advantages of discrete and continuous optimization methods it is essential to minimize the same energy in both optimization methods. Starting from a continuous energy formulation in § 2.1, we first show how to discretize the energy in § 2.2 and subsequently minimize it using a novel discrete parallel block coordinate descent, described in § 2.3. The output of this algorithm will be the input to a refinement method which is posed as a continuous optimization problem, solved by a non-linear primal-dual algorithm described in § 2.4.

2.1. Model

Let us formally define the dense image matching problem to be addressed by the discrete-continuous optimization approach. In both formulations we consider that the image domain is a discrete *set of pixels* $\mathcal{V}$. The continuous formulation has continuous ranged variables $u = (u_i^k \in \mathbb{R} \mid k = 1, \dots d,\ i \in \mathcal{V})$, where $d = 1, 2$ for stereo / flow, respectively. The matching problem is formulated as

$$\min_{u \in U} \Big[E(u) = D(u) + R(Au) \Big], \tag{1}$$

where $U = \mathbb{R}^{d \times \mathcal{V}}$; D is the *data term* and $R(Au)$ is a *regularizer* (A is a linear operator explained below). The discrete formulation will quantize variable ranges.

Data Term We assume $D(u) = \sum_{i \in \mathcal{V}} D_i(u_i)$, where $D_i \colon \mathbb{R}^d \to \mathbb{R}$ encodes the deviation of u_i from some underlying measurement. A usual choice for dense image matching are robust filters like Census Transform or Normalized Cross Correlation, computed on a small window around a pixel. This data term is non-convex in u and piecewise linear. In the discrete setting, the data term is sampled at discrete locations, in the continuous setting, the data term is convexified by linearizing or approximating it around the current solution. The details will be described in the respective sections.

Regularization Term The regularizer encodes properties of the solution of the energy minimization like local smoothness or preservation of sharp edges. The choice of this term is crucial in practice, since the data term may be unreliable or uninformative in large areas of dense matching problems. We assume

$$R(Au) = \sum_{ij \in \mathcal{E}} \omega_{ij} \sum_{k=1}^{d} r((Au^k)_{ij}), \tag{2}$$

where $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is the set of *edges*, *i.e.*, pairs of neighboring pixels; linear operator $A \colon \mathbb{R}^{\mathcal{V}} \to \mathbb{R}^{\mathcal{E}} \colon u^k \mapsto$

$(u_i^k - u_j^k \in \mathbb{R} \mid \forall ij \in \mathcal{E})$ essentially computes gradients along the edges in $\mathcal{E}$ for the solution dimension k; the gradients are penalized by the *penalty function* $r\colon \mathbb{R} \to \mathbb{R}$ and ω_{ij} are image dependent per-edge strength weights, reducing the penalty around sharp edges. Our particular choice for the penalty function r is depicted in Fig. 2. We chose to use a truncated norm which has shown to be robust against noise that one typically encounters in dense matching problems. It generalizes truncated Total Variation in the continuous setting. In the discrete setting it generalizes the $P1$-$P2$ penalty model [11], Potts model and the truncated linear model.

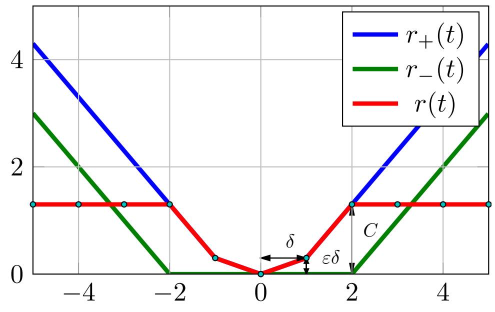


Figure 2: Regularizer function r. In our continuous optimization method it is decomposed into a difference of convex functions $r_+ - r_-$. For the discrete optimization it is sampled at label locations depicted as dots.

2.2. Discrete Formulation

In the discrete representation we will use the following formalism. To a continuous variable u_i we associate a discrete variable $x_i \in \mathcal{L}$. The discrete label space $\mathcal{L}$ can be chosen to our convenience as long as it has the desired number of elements, denoted K. We let $\mathcal{L}$ to be vectors in $\{0,1\}^K$ with exactly one component equal 1 (the 1-hot encoding of natural numbers from 1 to K). For $f_i \in \mathbb{R}^K$ we denote $f_i(x_i) = \langle f_i, x_i\rangle = f_i^{\mathsf{T}} x_i$ and for $f_{ij} \in \mathbb{R}^{K\times K}$ we denote $f_{ij}(x_i, x_j) = x_i^{\mathsf{T}} f_{ij} x_j$. Let $f = (f_w \mid w \in \mathcal{V} \cup \mathcal{E})$ denote the energy *cost vector*. The *energy function* corresponding to the cost vector f is given by

$$f(x) = \sum_{i\in\mathcal{V}} f_i(x_i) + \sum_{ij\in\mathcal{E}} f_{ij}(x_i, x_j). \tag{3}$$

Whenever we need to refer to f as a function and not as the cost vector, we will always use the argument notation, *e.g.* $f(x) \geq g(x)$ is different from $f \geq g$.

Energy function f that can be written as $\sum_i f_i(x_i) = \langle f, x\rangle$ is called *modular*, *separable* or *linear*. Formally, all components f_{ij} of f are identically zero. If f_{ij} is nonzero only for a subgraph of $(\mathcal{V}, \mathcal{E})$ which is a set of chains, we say that f is a *chain*.

The *discrete energy minimization* problem is defined as

$$\min_{x\in\mathcal{L}^{\mathcal{V}}} f(x). \tag{4}$$

Stereo We discretize a range of disparities and let $u(x) \in \mathbb{R}^{\mathcal{V}}$ denote the continuous solution corresponding to the *labeling* x. We set $f_i(x_i) = D_i(u(x_i))$ and $f_{ij}(x_i, x_j) = \omega_{ij} r((Au(x))_{ij})$.

Flow Discretization of the flow is somewhat more challenging. Since u_i is a 2D vector, assuming large displacements, discretizing all combinations is not tractable. Instead, components u_i^1 and u_i^2 can be represented as separate discrete variables x_{i^1}, x_{i^2} , where (i^1, i^2) is a pair of nodes duplicating i, leading to the *decomposed* formulation [26]. To retain the pairwise energy form (3), this approach assigns the data terms $D_i(u_i)$ to a pairwise cost $f_{i^1 i^2}(x_{i^1}, x_{i^2})$ and the regularization is imposed on each layer of variables $(x_{i^1} \mid i \in \mathcal{V})$ and $(x_{i^2} \mid i \in \mathcal{V})$ separately. To this end, we tested a yet simpler representation, in which we assign optimistic data costs, given by

$$f_{i^1}(x_{i^1}) = \min\nolimits_{x_{i^2}} D_i(x_{i^1}, x_{i^2}), \tag{5a}$$

$$f_{i^2}(x_{i^2}) = \min\nolimits_{x_{i^1}} D_i(x_{i^1}, x_{i^2}), \tag{5b}$$

where $D_i(x_{i^1}, x_{i^2})$ is the discretized data cost, and regularize in each layer individually. This makes the two layers fully decouple into, essentially, a two independent stereo-like problems. At the same time, the coupled scheme [26], passing messages between the two layers, differs merely in recomputing (5) for a reparametrized data costs in a loop. Our simplification then is not a principled limitation but an intermediate step.

2.3. Discrete Optimization

In this section we give an overview of a new method under development addressing problem (4) through its LP-relaxation dual. In real-time applications like stereo and flow there seem to be a demand in methods performing fast approximate discrete optimization, preferably well-parallelizable. It has motivated a significant research. The challenge may sound as "*best solution in a limited time budget*".

Well-performing methods, from local to global, range from cost volume filtering [12], semi-global matching (SGM) [11] (has been implemented in GPU and FPGA [2]), dynamic programming on spanning trees adjusting the cost volume [3] and more-global matching (MGM) [10] to the sequential dual block coordinates methods, such as TRW-S [15]. Despite being called sequential, TRW-S exhibits a fair amount of parallelism in its computation dependency graph, which is exploited in the parallel GPU/FPGA implementations [7, 13]. At the same time SGM has been interpreted [9] as a single step of parallel TRW algorithm [32] developed for solving the dual. MGM goes further in this direction, resembling even more the structure of a dual solver: it combines together more messages but in a heuristic fashion and introducing more computation dependencies, in fact similar to TRW-S. It appears that all these approaches go somehow in the direction of a fast processing of the dual.

We propose a new dual update scheme, which: i) is a monotonous block-coordinate ascent; ii) performs as good as TRW-S for an equal number of iterations while having a comparable iteration cost; and iii) offers more parallelism, better mapping to current massively parallel compute architectures. Thus it bridges the gap between highly parallel heuristics and the best "sequential" dual methods without compromising on the speed and performance.

On a higher level, the method is most easily presented in the dual decomposition framework. For clarity, let us consider a decomposition into two subproblems only (horizontal and vertical chains). Consider minimizing the energy function $E(x)$ that separates as

$$E(x) = f(x) + g(x), \tag{6}$$

where $f,\ g\colon \mathcal{L}^{\mathcal{V}} \to \mathbb{R}$ are chains.

Primal Majorize-Minimize Even before introducing the dual, we can propose applying the majorize-minimize method (a well-known optimization technique) to the primal problem in the form (6). It is instructive for the subsequent presentation of the dual method and has an intriguing connection to it, which we do not yet fully understand.

Definition 2.1. A modular function $\bar{f}$ is a *majorant* (upper bound) of f if $(\forall x)\quad \bar{f}(x) \geq f(x)$, symbolically $\bar{f} \succeq f$. A modular minorant $\underline{f}$ of f is defined similarly.[1]

Noting that minimizing a chain function plus a modular function is easy, one could straightforwardly propose Algorithm 1, which alternates between majorizing one of f or g by a modular function and minimizing the resulting chain problem $\bar{f} + g$ (resp. $f + \bar{g}$). We are not aware of this approach being evaluated before. Somewhat novel, the sum of two chain functions is employed rather than, say, difference of submodular [19], but the principle is the same. To ensure monotonicity of the algorithm we need to pick a majorant $\bar{f}$ of f which is exact in the current primal solution x^k as in Line 1. Then $f(x^{k+1}) + g(x^{k+1}) \leq \bar{f}(x^{k+1}) + g(x^{k+1}) \leq \bar{f}(x^k) + g(x^k) = f(x^k) + g(x^k)$. Steps 3-4 are completely similar. Algorithm 1 has the following properties:

- primal monotonous;
- parallel, since, *e.g.*, $\min_x(\bar{f} + g)(x)$ decouples over all vertical chains;
- uses more information about subproblem f than just the optimal solution (as in most primal block-coordinate schemes: ICM, alternating lines, *etc.*).

The performance of this method highly depends on the strategy of choosing majorants. This will be also the main question to address in the dual setting.

Dual Decomposition Minimization of (6) can be written as

$$\min_{x^1 = x^2} f(x^1) + g(x^2). \tag{7}$$

Introducing a vector of Lagrange multipliers $\lambda \in \mathbb{R}^{\mathcal{L}\times\mathcal{V}}$ for the constraint $x^1 = x^2$, we get the Lagrange dual problem:

[1] $\underline{f}$ reads "f underbar".

Algorithm 1: `Primal_MM`

```
Input: Initial primal point $x^k$;
Output: New primal point $x^{k+2}$;
1 $\bar{f} \succeq f$, $\bar{f}(x^k) = f(x^k)$;            /* Majorize */
2 $x^{k+1} \in \underset{x}{\mathrm{argmin}}(\bar{f} + g)(x)$;   /* Minimize */
3 $\bar{g} \succeq g$, $\bar{g}(x^{k+1}) = g(x^{k+1})$;      /* Majorize */
4 $x^{k+2} \in \underset{x}{\mathrm{argmin}}(f + \bar{g})(x)$;   /* Minimize */
```

$$\max_{\lambda}\Big[\underbrace{\min_x \big(f(x) + \langle\lambda, x\rangle\big)}_{D^1(\lambda)} + \underbrace{\min_x \big(g(x) - \langle\lambda, x\rangle\big)}_{D^2(\lambda)}\Big]. \tag{8}$$

The so-called *slave* problems $D^1(\lambda)$ and $D^2(\lambda)$ have the form of minimizing an energy function with a data cost modified by λ. The goal of the *master problem* (8) is to balance the data cost between the slave problems such that their solutions agree. The slave problems are minima of finitely many functions linear in λ, the objective of the master problem (8) $D(\lambda) = D^1(\lambda) + D^2(\lambda)$ is thus a concave piece-wise linear function. Problem (8) is a concave maximization. However, since x was taking values in a discrete space, there is only a weak duality: (7) $\geq$ (8). It is known that (8) can be written as a linear program (LP), which is as difficult in terms of computation complexity as a general LP [22].

Dual Minorize-Maximize In the dual, which is a maximization problem, we will speak of a minorize-maximize method. The setting is similar to the primal. We can *efficiently* maximize D^1, D^2 but not $D^1 + D^2$. Suppose we have an initial dual point λ^0 and let $x^0 \in \mathrm{argmin}_x(f + \lambda^0)(x)$ be a solution to the slave subproblem D^1, that is, $D^1(\lambda^0) = f(x^0) + \lambda^0(x^0)$.

Proposition 2.2. Let $\underline{f}$ be a modular minorant of f exact in x^0 and such that $\underline{f} + \lambda^0 \geq D^1(\lambda^0)$ (component-wise). Then the function $\underline{D}^1(\lambda) = \min_x(\underline{f} + \lambda)(x)$ is a minorant of $D^1(\lambda)$ exact at $\lambda = \lambda^0$.

Proof. Since $\underline{f}(x) \leq f(x)$ for all x it follows that $\min_x(\underline{f} + \lambda)(x) \leq \min_x(f + \lambda)(x)$ for all λ and therefore $\underline{D}^1$ is a minorant of D^1. Next, on one hand we have $\underline{D}^1(\lambda^0) \leq D^1(\lambda^0)$ and on the other, $D^1(\lambda^0) \leq (\underline{f} + \lambda^0)(x)$ for all x and thus $D^1(\lambda^0) \leq \underline{D}^1(\lambda^0)$. $\square$

We have constructed a minorant of D^1 which is itself a (simple) piece-wise linear concave function. The maximization step of the minorize-maximize is to solve

$$\max_{\lambda}(\underline{D}^1(\lambda) + D^2(\lambda)). \tag{9}$$

Proposition 2.3. $\lambda^* = -\underline{f}$ is a solution to (9).

Proof. Substituting λ^* into the objective (9) we obtain $\underline{D}^1(\lambda^*) + D^2(\lambda^*) = \min_x(\underline{f} - \underline{f})(x) + D^2(-\underline{f}) = \min_x(\underline{f} + g)(x)$. This value is the maximum because

```
Algorithm 2: Dual_MM
  Input: Initial dual point $\underline{g}^k$;
  Output: New dual point $\underline{g}^{k+2}$;
1 $x^k \in \mathrm{argmin}_x(f + \underline{g}^k)(x)$;        /* Minimize */
  /* Minorize                                                   */
2 $\underline{f}^{k+1} \preceq f$, $\underline{f}^{k+1}(x^k) = f(x^k)$,
  $\underline{f}^{k+1} + \underline{g}^k \geq f(x^k) + \underline{g}^k(x^k)$;
3 $x^{k+1} \in \mathrm{argmin}_x(\underline{f}^{k+1} + g)(x)$;  /* Minimize */
  /* Minorize                                                   */
4 $\underline{g}^{k+2} \preceq g$, $\underline{g}^{k+2}(x^{k+1}) = g(x^{k+1})$,
  $\underline{f}^{k+1} + \underline{g}^{k+2} \geq \underline{f}^{k+1}(x^{k+1}) + g(x^{k+1})$;
```

$\underline{D}^1(\lambda) + D^2(\lambda) = \min_x(\underline{f} + \lambda)(x) + \min_x(g - \lambda)(x) \leq \min_x(\underline{f} + \lambda + g - \lambda)(x) = \min_x(\underline{f} + g)(x)$. $\square$

Note, for the dual point $\lambda = -\underline{f}$, in order to construct a minorant of D^2 (similarly to Proposition 2.2) we need to find a solution to the second slave problem,

$$x^1 \in \mathrm{argmin}(g - \lambda)(x) = \mathrm{argmin}(\underline{f} + g)(x). \tag{10}$$

We obtain Algorithm 2 with the following properties:

- It builds the sequence of dual points given by $\lambda^{2t} = \underline{g}^{2t}$, $\lambda^{2t+1} = -\underline{f}^{2t+1}$ and the dual objective does not decrease on each step;
- The minimization subproblems and minorants are decoupled (can be solved in parallel) for all horizontal (resp. vertical) chains;
- When provided good minorants (see below) the algorithm has same fixed points as TRW-S [15];
- Updating only a single component λ_i for a pixel i is a monotonous step as well, therefore the algorithm is a *parallel block-coordinate ascent*.

Notice also that Dual_MM and Primal_MM are very similar, nearly up to replacing minorants with majorants. The sequence $\{E(x^k)\}_k$ is monotonous in Algorithm 1 but not in Algorithm 2.

Good and Fast Minorants The choice of the minorant in Dual_MM is non-trivial as there are many, which makes it sort of a secrete ingredient. Figure 3 illustrates two of the possible choices. The *naive* minorant for a chain problem $f + \lambda$ is constructed by calculating its min-marginals and dividing by chain length to ensure that the simultaneous step is monotonous (*c.f.* tree block update algorithm of Sontag and Jaakkola [29, Fig. 1]). The *uniform* minorant is found through the optimization procedure that tries to build the tightest modular lower bound, by increasing uniformly all components that are not yet tight. The details are given in §A. In practice, we build fast minorants, which try to approximate the uniform one using fast message passing operations. Parallelization of decoupled chains allowed us to achieve an implementation which, while having the same number of memory accesses as TRW-S (including messages / dual variables), saturates the GPU memory bandwidth, $\sim$ 230GB/s.[2] This allows to perform 5 iterations of Algorithm 2 for an image 512×512 and 64 labels at the rate of about 30 fps.

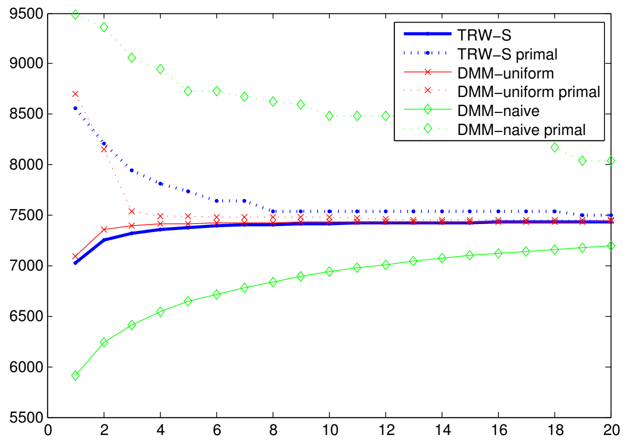


Figure 3: Lower bounds and best primal solutions by TRW-S and by Dual_MM with a naive and a uniform minorants. The problem is a small crop from stereo of size 40×40, 16 labels, truncated linear regularization. On the x-axis one iteration is a forward-backward pass of TRW-S *vs.* iteration of Dual_MM (equal number of updates per pixel). With a good choice of minorant, Dual_MM can perform even better than the sequential baseline in terms of iterations. Parallelizing it can be expected to give a direct speedup.

2.4. Continuous Refinement

In this section we describe the continuous refinement method, which is based on variational energy minimization. The goal of this step is to refine the output of the optimization method described in § 2.3 which is discrete in label-space.

To that end, it is important to minimize the same energy in both formulations. Considering the optimization problem in (1), we are seeking to minimize a non-convex, truncated norm together with a non-convex data term. For clarity, let us write down the problem again:

$$\min_{u \in U} D(u) + R(Au). \tag{11}$$

Non-Convex Primal-Dual Efficient algorithms exist to solve (11) in case both $D(u)$ and $R(Au)$ are convex (but possibly non-smooth), *e.g.* the primal-dual solver of Chambolle and Pock [6]. Kolmogorov et al. [16] solves (11) for a truncated total variation regularizer using a splitting into horizontal and vertical 1D problems and applying [6] to the Lagrangian function. Here we will use a recently proposed extension to [6] by Valkonen [31]. He considers problems of the form $\min_x \mathcal{G}(x) + \mathcal{F}(\mathcal{A}(x))$, *i.e.* of the same structure as (11), where $\mathcal{G}$ and $\mathcal{F}$ are convex, $\mathcal{G}$ is differentiable and $\mathcal{A}(u)$ is a twice differentiable but possibly non-linear operator. In the primal-dual formulation, the problem is written as

$$\min_x \max_y \left[\mathcal{G}(x) + \langle \mathcal{A}(x), y \rangle - \mathcal{F}^*(y)\right], \tag{12}$$

[2]This is about 10 times faster than reported for FPGA implementation [7] of TRW-S.

where * is the convex conjugate. Valkonen proposes the following modified primal-dual hybrid gradient method:

$$x^{k+1} = (I + \tau\partial\mathcal{G})^{-1}(x^k - \tau\left[\nabla\mathcal{A}(x^k)\right]^\mathsf{T} y^k) \tag{13a}$$
$$y^{k+1} = (I + \sigma\partial\mathcal{F}^*)^{-1}(y^k + \sigma\mathcal{A}(2x^{k+1} - x^k)). \tag{13b}$$

Reformulation In order to apply method [31], we will reformulate the non-convex problem (11) to the form (12). We start by formulating the regularizer $R(Au)$ as a *difference of convex functions*: $R(Au) = R_+(Au) - R_-(Au)$, where R_+ and R_- are convex. The primal-dual formulation of (11) then reads

$$\min_u \Big[\max_p(\langle Au, p\rangle - R_+^*(p)) + \max_q(\langle Au, q\rangle - R_-^*(q)) + D(u)\Big]. \tag{14}$$

Because $\min_x -f(x) = -\max_x f(x)$, (14) equals

$$\min_u \Big[\max_p(\langle Au, p\rangle - R_+^*(p)) + + \min_q(-\langle Au, q\rangle + R_-^*(q)) + D(u)\Big]. \tag{15}$$

Grouping terms we arrive at

$$\min_{u,q} \max_p \Big[\langle Au, p-q\rangle - R_+^*(p) + R_-^*(q) + D(u)\Big]. \tag{16}$$

The problem now arises in minimizing the bilinear term $\langle Au, q\rangle$ in (16) in both u and q. We thus move this term into the nonlinear operator $\mathcal{A}(x)$ and rewrite (16) as

$$\underbrace{\min_{u,q}}_{x} \underbrace{\max_{p,d=1}}_{y} \left\langle \underbrace{\begin{bmatrix} Au \\ -\langle Au, q\rangle \end{bmatrix}}_{\mathcal{A}(x)}, \begin{bmatrix} p \\ d \end{bmatrix} \right\rangle + \underbrace{R_-^*(q) + D(u)}_{\mathcal{G}(x)} - \underbrace{R_+^*(p)}_{\mathcal{F}^*(y)} \tag{17}$$

by introducing a dummy variable $d = 1$.

Implementation Details The gradient of $\mathcal{A}$ needed by iterates (13) is given by

$$\nabla\mathcal{A}(x) = \begin{bmatrix} A & 0 \\ -A^\mathsf{T} q & -Au \end{bmatrix}. \tag{18}$$

The regularization function r is represented as a difference of two convex functions (see Figure 2):

$$r(t) = r_{\varepsilon,\delta}(t) - r_{0,(C+\delta-\varepsilon\delta)}(t), \tag{19}$$

where

$$r_{\alpha,\beta}(t) = \begin{cases} \alpha|t| & \text{if } |t| \le \beta \\ |t| - \beta(1-\alpha) & \text{else} \end{cases} \tag{20}$$

is convex for $\alpha \le 1$. Convex functions $R_+(Au)$ and $R_-(Au)$ are defined by decomposition (19) and (2).

To compute the proximal map $(I + \sigma\partial\mathcal{F}^*)^{-1}(\hat{y})$ we first need the convex conjugate of $\omega_{ij} r_{\alpha,\beta}(t)$. It is given by $(\omega_{ij} r_{\alpha,\beta})^*(t^*) =$

$$\begin{cases} \max(0, \beta|t^*| - \omega_{ij}\alpha\beta) & \text{if } \alpha < |t^*| < \omega_{ij} \\ \infty & \text{else} \end{cases}. \tag{21}$$

The proximal map for $(\omega_{ij} r_{\alpha,\beta})^*$ at $t^* \in \mathbb{R}$ is given by $\bar{t} = \operatorname{clamp}(\pm\omega_{ij}, t')$, where $\operatorname{clamp}(\pm\omega_{ij}, \cdot)$ denotes a clamping to the interval $[-\omega_{ij}, \omega_{ij}]$ and

$$t' = \begin{cases} t^* & \text{if } |t^*| \le \alpha\omega_{ij} \\ \max(\alpha\omega_{ij}, |t^*| - \beta\sigma)\operatorname{sign}(t^*) & \text{else.} \end{cases} \tag{22}$$

Proximal map $(I + \sigma\partial\mathcal{F}^*)^{-1}(\hat{y})$ is calculated by applying expression (22) component-wise to $\hat{y}$. The proximal map $(I + \tau\partial\mathcal{G})^{-1}$ depends on the choice of the data term $D(u)$ and will thus be defined in § 3.

3. Applications

3.1. Stereo Reconstruction

For the problem of estimating depth from two images, we look at a setup of two calibrated and synchronized cameras. We assume that the input images to our method have been rectified according to the calibration parameters of the cameras. We aim to minimize the energy (1) where u encodes the disparity in x-direction. The data term measures the data fidelity between images I_1 and I_2, warped by the disparity field u. As a data term we use the Census Transform [37] computed on a small local patch in each image. The cost is given by the pixel-wise Hamming distance on the transformed images.$D(u)$ is non-convex in the argument u which makes the optimization problem in (1) intractable in general.

We start by minimizing (1) using the discrete method (§2.3) in order to obtain an initial solution $\mathring{u}$. We approximate the data term around the current point $\mathring{u}$ by a piecewise linear convex function $\tilde{D}(u) =$

$$D(\mathring{u}) + \delta_{[\mathring{u}-h,\mathring{u}+h]}(u) + \begin{cases} s_1(u - \mathring{u}) & \text{if } u \le \mathring{u} \\ s_2(u - \mathring{u}) & \text{otherwise} \end{cases} \tag{23}$$

with $s_1 = \frac{D(\mathring{u}+h) - D(\mathring{u})}{h}$ and $s_2 = \frac{D(\mathring{u}) - D(\mathring{u}+h)}{h}$ for a small h. To ensure convexity, we set $s_1 = s_2 = \frac{s_1+s_2}{2}$ if $s_2 < s_1$. The indicator function δ is added to ensure that the solution stays within $\mathring{u} \pm h$ where the approximation is valid. We then apply the continuous method (§2.4). The proximal map $\bar{u} = (I + \tau\partial\mathcal{G})^{-1}(\hat{u})$ needed by the algorithm (13) for the approximated data term expresses as the pointwise soft-thresholding

$$\bar{u}_i = \operatorname{clamp}\left(\mathring{u}_i \pm h, \hat{u}_i - \begin{cases} \tau s_{1,i} & \text{if } \hat{u}_i > \mathring{u}_i + \tau s_{1,i} \\ \tau s_{2,i} & \text{if } \hat{u}_i < \mathring{u}_i + \tau s_{2,i} \\ 0 & \text{otherwise} \end{cases}\right)$$

In practice, the minimization has to be embedded in a *warping framework*: after optimizing for n iterations, the data term is approximated anew at the current solution u.

3.2. Optical Flow

The optical flow problem for two images I_1, I_2 is posed again as model (1). In contrast to stereo estimation, we now have $u_i \in \mathbb{R}^2$ encoding the flow vector. For the discrete optimization step (§2.3) the flow problem is decoupled into two independent stereo-like problems as discussed in §2.2.

For the continuous refinement step, the main problem is again the non-convexity of the data term. Instead of a convex approximation with two linear slopes we build a quadratic approximation, now in 2D, following [34]. The approximated data term reads $\tilde{D}_i(u_i) = \delta_{[\mathring{u}_i-h,\mathring{u}_i+h]}(u_i)+$

$$D_i(\mathring{u}_i) + L_i^\mathsf{T}(u_i - \mathring{u}_i) + \frac{1}{2}(u_i - \mathring{u}_i)^\mathsf{T} Q_i (u_i - \mathring{u}_i), \quad (24)$$

where $L_i \in \mathbb{R}^2$ and $Q_i \in \mathbb{R}^{2\times2}$ are finite difference approximations of the gradient and the Hessian with stepsize h. Convexity of (24) is ensured by retaining only positive-semidefinite part of Q_i as in [34]. The proximal map $\bar{u} = (I + \tau\partial\mathcal{G})^{-1}(\hat{u})$ for data term (24) is given point-wise by

$$\bar{u}_i^k = \text{clamp}\left(\mathring{u}_i^k \pm h, \frac{\hat{u}_i^k + \tau(Q_i\mathring{u}_i - L_i)^k}{1 + \tau L_i^k}\right). \quad (25)$$

Optimizing (1) is then performed as proposed in §2.4.

4. Experiments

4.1. Stereo Reconstruction

We evaluate our proposed real-time stereo method on datasets where Ground-Truth data is available as well as on images captured using a commercially available stereo camera.

4.1.1 Influence of Truncated Regularizer

We begin by comparing the proposed method to a simplified version that does not use a truncated norm as regularizer but a standard Total Variation. We show the effect of this change in Fig. 4, where one can observe much sharper edges, when using a robust norm in the regularization term. On the downside it is more sensitive to outliers, which however can be removed in a post-processing step like a two-side consistency check.

4.1.2 Live Dense Reconstruction

To show the performance of our stereo matching method in a real live setting, we look at the task of creating a live dense reconstruction from a set of depth images. To that end, we are using a reimplementation of *KinectFusion* proposed by Newcombe et al. [20] together with the output of our method. This method was originally designed to be used with the RGBD output of a Microsoft Kinect and tracks the 6 DOF position of the camera in real-time.

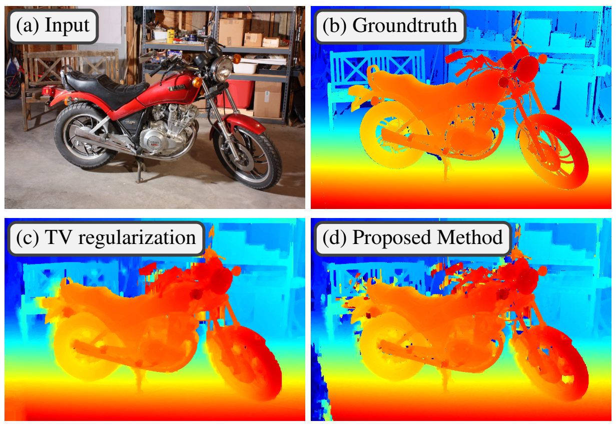


Figure 4: Influence of the robust regularizer in the continuous refinement on stereo reconstruction quality.

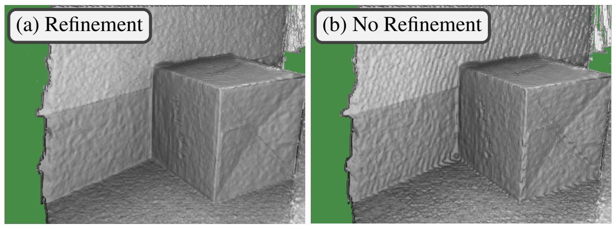


Figure 5: Influence of continuous refinement on the reconstruction quality of KinectFusion.

For the purpose of this experiment we replace the Kinect with a Point Grey Bumblebee2 stereo camera. *KinectFusion* can only handle relatively small camera movements between images, so a high framerate is essential. We set the parameters to our method to achieve a compromise between highest quality and a framerate of $\approx 4-5$ fps: camera resolution 640×480, 128 disparities, 4 iterations of `Dual_MM`, 5 warps and 40 iterations per warp of the continuous refinement.

Influence of Continuous Refinement The first stage of our reconstruction method, `Dual_MM`, already delivers high quality disparity images that include details on fine structures and depth discontinuities that are nicely aligned with edges in the image. In this experiment we want to show the influence of the second stage, the continuous refinement, on the reconstruction quality of KinectFusion. To that end we mount the camera on a tripod and collect 300 depthmaps live from our full method and 300 frames with the continuous refinement switched off. By switching off the camera tracking, the final reconstruction will show us the artifacts produced by the stereo method. Figure 5 depicts the result of this comparison. One can easily see that the output of the discrete method contains fine details, but suffers from staircasing artifacts on slanted surfaces due to the integer solution. The increase in quality due to the refinement stage can be especially seen on far away objects, where a disparity step of 1 pixel is not enough to capture smooth surfaces.

Timing To show the influence of the individual steps in our stereo method on runtime, we break down the total

Cost Vol.	**Discrete**	**Cont. Ref.**	**Total**
27 ms	73 ms	39 ms	**139 ms**

Table 1: Runtime analysis of the individual components of our stereo matching method. Details regarding computing hardware and parameters are in the text. In case of the full left-right check procedure the total computation time doubles.

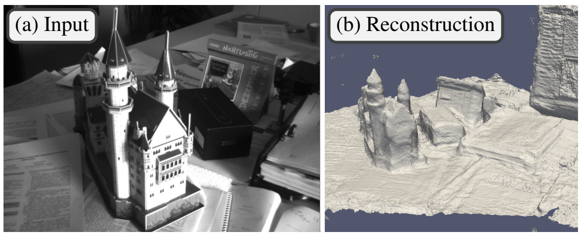


Figure 6: Qualitative result of reconstructing a desktop scene using KinectFusion[3].

time of ≈ 140 ms per frame in Table 1. Those timings have been achieved using a PC with 32 GB RAM with a NVidia 980GTX, running Linux.

Qualitative Results To give an impression about the quality of the generated depthmaps and the speed of our method, we run our full algorithm and aim to reconstruct a desktop scene with a size of $1 \times 1 \times 1$ meters and show some renderings in Fig. 6. To better visualize the quality of the geometry, the model is rendered without texture[3].

4.2. Optical Flow

In this section we show preliminary results of our algorithm applied to optical flow. A further improvement in quality can be expected by exploiting the coupled scheme [26] in the discrete optimization, as discussed in § 2.2. As depicted in Figure 7, our method is able to deliver reasonable results on a variety of input images. We deliberately chose scenes that contain large motion as well as small scale objects, to highlight the strengths of the discrete-continuous approach. For comparison, we use a state-of-the-art purely continuous variational optical flow algorithm [33]. The runtime of our method is 2s for an image of size 640×480.

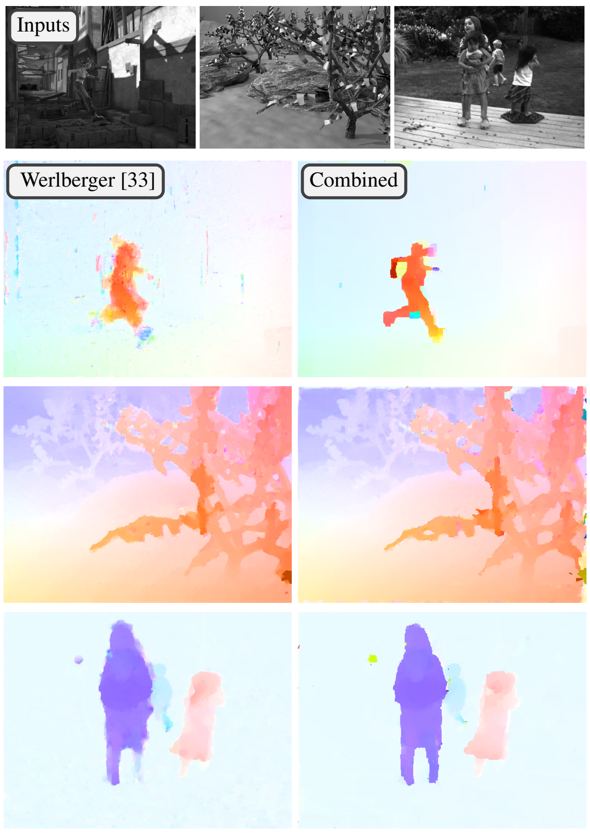


Figure 7: Subjective comparison of variational approach [33] (left) with our combined method (right). Top row show input images, one from a pair. Both methods use the same data term. Parameters of both algorithms have been tuned by hand to deliver good results. Note that for [33] it is often impossible to get sharp motion boundaries as well as small scale details, despite a very strong data term (*e.g.* artifacts in left image, first row).

5. Conclusion

The current results demonstrate that it is feasible to solve dense image matching problems using global optimization methods with a good quality in real time. We have proposed a highly parallel discrete method, which even when executed sequentially, is competitive with the best sequential methods. As a dual method, we believe, it has a potential to smoothly handle more complex models in the dual decomposition framework and is in theory applicable to general graphical models. When the solution is sufficiently localized, continuous representation increases the accuracy of the model as well as optimization speed. In the continuous optimization, we experimented with non-convex models and showed a reduction allowing to handle them with the help of a recent non-linear primal-dual method. This in turn allowed to speak of a global model to be solved by a discrete-continuous optimization.

Ideally, we would like to achieve a method, which, when given enough time, produces an accurate solution, and in the real time setting gives a robust result. We plan further to improve on the model. A vast literature on the topic suggest that modeling occlusions and using planar hypothesis can be very helpful. At the same time, we are interested in a tighter coupling of discrete and continuous optimization towards a globally optimal solution.

Acknowledgements

This work was supported by the research initiative Mobile Vision with funding from the AIT and the Austrian Federal Ministry of Science, Research and Economy HRSM programme (BGBl. II Nr. 292/2012).

[3]We point the interested reader to a video that shows the reconstruction pipeline in real-time: `http://gpu4vision.icg.tugraz.at/videos/cvww16.mp4`

References

[1] Arashloo, S. R. and Kittler, J. (2014). Fast pose invariant face recognition using super coupled multiresolution Markov random fields on a GPU. *Pattern Recognition Letters*, 48.

[2] Banz, C., Hesselbarth, S., Flatt, H., Blume, H., and Pirsch, P. (2010). Real-time stereo vision system using semi-global matching disparity estimation: Architecture and FPGA-implementation. In *ICSAMOS*.

[3] Bleyer, M. and Gelautz, M. (2008). Simple but effective tree structures for dynamic programming-based stereo matching. In *VISAPP*.

[4] Brox, T., Bruhn, A., Papenberg, N., and Weickert, J. (2004). High accuracy optical flow estimation based on a theory for warping. In *ECCV*.

[5] Brox, T., Bruhn, A., and Weickert, J. (2006). Variational motion segmentation with level sets. In *ECCV*, volume 3951.

[6] Chambolle, A. and Pock, T. (2011). A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1).

[7] Choi, J. and Rutenbar, R. A. (2012). Hardware implementation of MRF MAP inference on an FPGA platform. In *Field Programmable Logic*.

[8] Combettes, P. L. and Pesquet, J.-C. (2011). Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*.

[9] Drory, A., Haubold, C., Avidan, S., and Hamprecht, F. (2014). Semi-global matching: A principled derivation in terms of message passing. In *Pattern Recognition*, volume 8753.

[10] Facciolo, G., de Franchis, C., and Meinhardt, E. (2015). MGM: A significantly more global matching for stereovision. In *BMVC*.

[11] Hirschmuller, H. (2011). Semi-global matching-motivation, developments and applications.

[12] Hosni, A., Rhemann, C., Bleyer, M., Rother, C., and Gelautz, M. (2013). Fast cost-volume filtering for visual correspondence and beyond. *PAMI*, 35(2).

[13] Hurkat, S., Choi, J., Nurvitadhi, E., Martınez, J. F., and Rutenbar, R. A. (2012). Fast hierarchical implementation of sequential tree-reweighted belief propagation for probabilistic inference. In *Field Programmable Logic*.

[14] Kappes, J. H., Andres, B., Hamprecht, F. A., Schnörr, C., Nowozin, S., Batra, D., Kim, S., Kausler, B. X., Lellmann, J., Komodakis, N., and Rother, C. (2013). A comparative study of modern inference techniques for discrete energy minimization problem. In *CVPR*.

[15] Kolmogorov, V. (2006). Convergent tree-reweighted message passing for energy minimization. *PAMI*, 28(10).

[16] Kolmogorov, V., Pock, T., and Rolinek, M. (2015). Total variation on a tree. *CoRR*, abs/1502.07770.

[17] Lawler, E. (1966). Optimal cycles in doubly weighted directed linear graphs. In *Intl Symp. Theory of Graphs*.

[18] Menze, M., Heipke, C., and Geiger, A. (2015). Discrete optimization for optical flow. In *GCPR*.

[19] Narasimhan, M. and Bilmes, J. (2005). A supermodular-submodular procedure with applications to discriminative structure learning. In *Uncertainty in Artificial Intelligence*.

[20] Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohli, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011). Kinectfusion: Real-time dense surface mapping and tracking. In *ISMAR*.

[21] Ochs, P., Chen, Y., Brox, T., and Pock, T. (2014). ipiano: Inertial proximal algorithm for non-convex optimization. *SIAM JIS*, 7(2).

[22] Prusa, D. and Werner, T. (2015). Universality of the local marginal polytope. *PAMI*, 37(4).

[23] Ranftl, R., Bredies, K., and Pock, T. (2014). Non-local total generalized variation for optical flow estimation. In *ECCV*.

[24] Ranftl, R., Gehrig, S., Pock, T., and Bischof, H. (2012). Pushing the limits of stereo using variational stereo estimation. In *Intelligent Vehicles Symposium*.

[25] Roth, S., Lempitsky, V., and Rother, C. (2009). Discrete-continuous optimization for optical flow estimation. In *Statistical and Geometrical Approaches to Visual Motion Analysis*, volume 5604.

[26] Shekhovtsov, A., Kovtun, I., and Hlaváč, V. (2008). Efficient MRF deformation model for non-rigid image matching. *CVIU*, 112.

[27] Shekhovtsov, A., Swoboda, P., and Savchynskyy, B. (2015). Maximum persistency via iterative relaxed inference with graphical models. In *CVPR*.

[28] Sinha, S. N., Scharstein, D., and Szeliski, R. (2014). Efficient high-resolution stereo matching using local plane sweeps. In *CVPR*.

[29] Sontag, D. and Jaakkola, T. S. (2009). Tree block coordinate descent for MAP in graphical models. In *AISTATS*.

[30] Taniai, T., Matsushita, Y., and Naemura, T. (2014). Graph cut based continuous stereo matching using locally shared labels. In *CVPR*.

[31] Valkonen, T. (2014). A primal-dual hybrid gradient method for nonlinear operators with applications to MRI. *Inverse Problems*, 30(5).

[32] Wainwright, M., Jaakkola, T., and Willsky, A. (2005). MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Transactions on Information Theory*, 51(11).

[33] Werlberger, M. (2012). *Convex Approaches for High Performance Video Processing*. PhD thesis, Institute for Computer Graphics and Vision, Graz University of Technology, Graz, Austria.

[34] Werlberger, M., Pock, T., and Bischof, H. (2010). Motion estimation with non-local total variation regularization. In *CVPR*.

[35] Werner, T. (2007). A linear programming approach to max-sum problem: A review. *PAMI*, 29(7).

[36] Woodford, O., Torr, P., Reid, I., and Fitzgibbon, A. (2009). Global stereo reconstruction under second-order smoothness priors. *PAMI*, 31(12).

[37] Zabih, R. and Woodfill, J. (1994). Non-parametric local transforms for computing visual correspondence. In *ECCV*, volume 801.

Appendix A. Details of Dual MM

In this section we specify details regarding computation of minorants in `Dual_MM`. The minorants are computed using message passing and we'll also need the notion of min-marginals.

A.1. Min-Marginals and Message Passing

Definition A.1. For cost vector f its *min-marginal* at node i is the function $m^f : \mathcal{L} \to \mathbb{R}$ given by

$$m^f(x_i) = \min_{x_{\mathcal{V}\backslash i}} f(x). \tag{26}$$

Function $m^f(x_i)$ is a min projection of $f(x)$ onto x_i only. Given the choice of x_i, it returns the cost of the best labeling in f that passes through x_i. For a chain problem it can be computed using dynamic programming. Let us assume that the nodes $\mathcal{V}$ are enumerated in the order of the chain and $\mathcal{E} = \{(i, i+1) \mid i = 1 \dots |\mathcal{V}| - 1\}$. We then need to compute: *left min-marginals*: $\varphi_{i-1,i}(x_i) :=$

$$\min_{x_{1,\dots i-1}} \sum_{i'<i} f_{i'}(x_{i'}) + \sum_{i'j' \in \mathcal{E} \mid i'<i} f_{i'j'}(x_{i'}, x_{j'}); \tag{27}$$

and *right min-marginals*: $\varphi_{i+1,i}(x_i) :=$

$$\min_{x_{i+1,\dots|V|}} \sum_{i'>i} f_{i'}(x_{i'}) + \sum_{i'j' \in \mathcal{E} \mid i' \geq i} f_{i'j'}(x_{i'}, x_{j'}). \tag{28}$$

These values for all $ij \in \mathcal{E}$, $x_i, x_j \in \mathcal{L}$ can be computed dynamically (recursively). After that, the min-marginal $m^f(x_i)$ expresses as

$$m^f(x_i) = f_i(x_i) + \varphi_{i-1,i}(x_i) + \varphi_{i+1,i}(x_i). \tag{29}$$

TRW-S method [15] can be derived as selecting one node i at a time and maximizing (8) with respect to λ_i only. For the two slave problems in (8) TRW-S needs to compute min-marginals $m^{f+\lambda}(x_i)$ and $m^{g-\lambda}(x_i)$. A (non-unique) optimal choice for λ_i would be to ensure that

$$m^{f+\lambda}(x_i) = m^{g-\lambda}(x_i) \ \forall x_i \in \mathcal{L} \tag{30}$$

by setting

$$\lambda_i := \lambda_i + (m^{g-\lambda}(x_i) - m^{f+\lambda}(x_i))/2. \tag{31}$$

If i and j are two nodes in a chain $f+\lambda$ then performing the update of λ_i changes the min-marginal at j and vice-versa. The updates must be implemented sequentially or otherwise one gets a non-monotonous behavior and the method may fail to converge (see [15]).

TRW-S gains its efficiency in that after the update (31), the min-marginal at a neighboring node can be recomputed by a single step of dynamic programming. Let the neighboring node be $j = i + 1$. The expression for the right min-marginal at j remains correct and the expression for left min-marginal is updated using its recurrent expression $\varphi_{ij}(x_j) :=$

$$\min_{x_i} \big[\varphi_{i-1,i}(x_i) + f_i(x_i) + f_{ij}(x_i, x_j)\big], \tag{32}$$

also known as *message passing*. Then min-marginal at j becomes available through (29).

It is possible to perform update (31) in parallel by scaling down the step size by the number of variables (or the length of the chain). This is equivalent to decomposing a chain f into n copies with costs f/n so that they contribute one for each node i with a min-marginal $m^f(x_i)/n$. Confer to the parallel tree block update algorithm of Sontag and Jaakkola [29, Fig. 1]). However, the gain from the palatalization does not pay off the decrease in the step size.

A.2. Slacks

In the following we will also use the term slack. Shortly, it is explained as follows. The dual problem (8) can be written as a linear program, see *e.g.*, [35]. Dual inequality constraints in that program can satisfied as equalities, in which case they are tight, or they can be satisfied as strict inequalities in which case there is a *slack*. Equivalent reparametrization of the problem (change of the dual variables) can propagate a slack from one constraint (corresponding to a label-node pair) to another one. If all constraints in a group becomes non-tight, their minimum slack can be subtracted and increments the lower bound. Since for a chain problem the LP relaxation is tight, the maximum slack that can be concentrated in a label-node equals the corresponding min-marginal.

A.3. Good Minoratns

Definition A.2. A modular minorant λ of f is *maximal* if there is no other modular minorant $\lambda' \geq \lambda$ such that $\lambda'(x) > \lambda(x)$ for some x.

Lemma A.3. For a maximal minorant λ of f all min-marginals of $f - \lambda$ are identically zero.

Proof. Since λ is a minorant, min-marginals $m_i(x_i) = \min_{x_{\mathcal{V}\backslash i}}[f(x) - \lambda(x)]$ are non-negative. Assume for contradiction that $\exists i$, $\exists x_i$ such that $m_i(x_i) > 0$. Clearly, $\lambda'(x) := \lambda(x) + m_i(x_i)$ is also a minorant and $\lambda' > \lambda$. □

Even using maximal minorants, the Algorithm 2 can get stuck in fixed points which do not satisfy weak tree agreement [15], *e.g.* suboptimal even in the class of message passing algorithms. Consider the following example of a minorant leading to a poor fixed point.

Example A.4. Consider a model in Figure 8 with two labels and strong Ising interactions ensuring that the optimal labeling is uniform. If we select minorants that just takes the unary term, without redistributing it along horizontal or vertical chains, the lower bound will not increase. For example, for the horizontal chain (v_1, v_2), the minorant $(1,\ 0)$ (displayed values correspond to $\lambda_v(1) - \lambda_v(2)$). This minorant is maximal, but it does not propagate the information available in v_1 to v_2 for the exchange with the vertical chain (v_2, v_4).

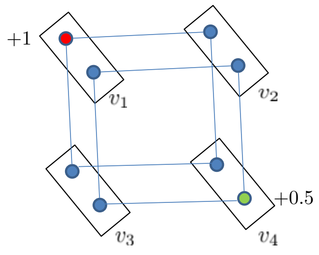


Figure 8: Example minorize-minimize stucks with a minorant that does not redistribute slack.

A.3.1 Uniform Minorants

Dual algorithms, by dividing the slacks between subproblems ensure that there is always a non-zero fraction of it (depending on the choice of weights in the scheme) propagated along each chain. We need a minorant, which will expose in every variable what is the preferable solution for the subproblem. We can even try to treat all variables uniformly. The practical strategy proposed below is motivated by the following.

Proposition A.5. Let $f^* = \min_x f(x)$ and let O_u be the support set of all optimal solutions x^*_u in $u \in \mathcal{V}$. Consider the minorant λ given by $\lambda_u(x_u) = \varepsilon(1 - O_u)$ and maximizing ε:

$$\max\{\varepsilon \mid (\forall x)\ \varepsilon\langle 1 - O, x\rangle \le f(x)\}. \tag{33}$$

The above minorant assigns cost ε to all labels but those in the set of optimal solutions. If the optimal solution x^* is unique, it takes the form $\lambda = \varepsilon(1 - x^*)$. This minorant corresponds to the direction of the subgradient method and ε determines the step size which ensures monotonicity. However it is not maximal. In $f - \lambda$ there still remains a lot of slack that can be useful when exchanging to the other problem. It is possible to consider $f - \lambda$ again. If we have solved (33), it will necessarily have a larger set of optimal solutions. We can search for a maximal ε_1 that can be subtracted from all non-optimal label-nodes in $f - \lambda$ and so on. The algorithm is specified as Algorithm 3.

Algorithm 3: Maximal Uniform Minorant

```
Input: Chain subproblem $f$;
Output: Minorant $\lambda$;
1 $\lambda := 0$;
2 while true
3     Compute min-marginals $m$ of $f - \lambda$;
4     if $m = 0$ then return $\lambda$;
5     Let $O := [\![m = 0]\!]$, the support set of optimal
      solutions of $m - \lambda$;
6     Find $\max\{\varepsilon \mid (\forall x)\ \varepsilon\langle 1 - O, x\rangle \le (f - \lambda)(x)\}$;
7     Let $\lambda := \lambda + \varepsilon(1 - O)$;
```

The optimization problem in Line 6 can be solved using the minimum ratio cycle algorithm of Lawler [17]. We search for a path with a minimum ratio of the cost given by $(f - \lambda)(x)$ to the number of selected labels with non-zero min-marginals given by $\langle 1 - O, x\rangle$. This algorithm is rather efficient, however Algorithm 3 it is still too costly and not well-suited for a parallel implementation. We will not use this method in practice directly, rather it establishes a sound baseline that can be compared to.

The resulting minorant λ is maximal and uniform in the following sense.

Lemma A.6. Let m be the vector of min-marginals of f. The uniform minorant λ found by Algorithm 3 satisfies

$$\lambda \ge m/n, \tag{34}$$

where n is the length of the longest chain in f.

Proof. This is ensured by Algorithm 3 as in each step the increment ε results from dividing the min-marginal by $\langle 1 - O, x\rangle$ which is at most the length of the chain. □

In fact, when the chain is strongly correlated, the minorant will approach m/n and we cannot do better than that. However, if the correlation is not as strong the minorant becomes tighter, and in the limit of zero pairwise interactions there holds $\lambda = m$. In a sense the minorant computes "decorrellated" min-marginals.

The next example illustrates uniform minorants and steps of the algorithm.

Example A.7. Consider a chain model with the following data unary cost entries (3 labels, 6 nodes):

0	0	1	0	0	8
9	7	0	3	2	8
7	3	6	9	1	0

The regularization is a Potts model with cost $f_{uv}(x_u, x_v) = 1[\![x_u \neq x_v]\!]$. Min-marginals of the problem and iteration of Algorithm 3 are ilustrated in Figure 9. At the first iteration the constructed minorant is

0	0	0	0	0	1
1	1	1	1	1	1
1	1	1	1	1	0

And the final minorant is:

0	0	0	0	0	7
8	7	1	2	2	7
6	4	6	7	1	0

The minorant follows min-marginals (first plot in Figure 9), because the interaction strength is relatively weak and min-marginals are nearly independent. If we increase interaction strength to 5, we find the following min-marginals and minorant, respectively:

0	0	0	0	0	3
14	15	8	8	7	8
12	13	15	10	1	0

0	0	0	0	0	3
5.5	5.5	3	3	3	3
4.75	4.75	4.75	4.75	1	0

It is seen that in this case min-marginals are correlated and only a fraction can be drained in parallel. The uniform approach automatically divides the cost equally between strongly correlated labels.

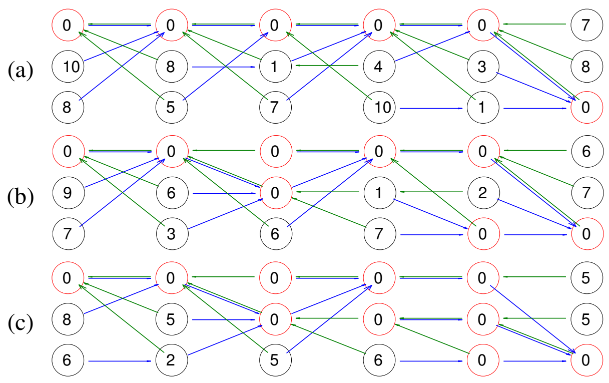


Figure 9: (a) Min-marginals (normalized by subtracting the value of the minimum) at vertices and arrows allowing to back-track the optimal solution passing through a given vertex. (b), (c) min-marginals of $f-\lambda$ after one (resp. two) iterations of Algorithm 3 ($\varepsilon_1 = 1$ and $\varepsilon_2 = 1$). With each iteration the number of vertices having zero min-marginal strictly increases.

A basic performance test of Dual_MM with uniform minorants versus TRW-S is shown in Figure 3. It demonstrates that the Dual_MM can be faster, when provided good minorants. The only problem is that determining the uniform minorant involves repeatedly solving minimum ratio path problems, plus there is a numerical instability in determining the support set of optimal solutions O.

A.3.2 Iterative Minorants

A simpler way to construct a maximal minorant would be to iteratively subtract from f a portion of its min-marginals and accumulate them in the minorant, until all min-marginals of the reminder become zero. Algorithm 4 implements this idea. The portion of min-marginals drained from the reminder $f-\lambda$ to the minorant λ in each iteration is controlled by $\gamma_s \in (0,1]$. Reversing the chain

Algorithm 4: Iterative Minorant

```
Input: Chain subproblem $f$;
Output: Minorant $\lambda$;
1 $\lambda := 0$;
2 for $s = 1 \ldots$ max_pass do
3     for $i = 1 \ldots |V|$ do
4         Compute min-marginal $m_i$ of $f - \lambda$ at $i$
          dynamically, equations (32) and (29);
5         $\lambda_i \mathrel{+}= \gamma_s m_i$;
6     Reverse the chain;
```

efficiently alternates between the forward and the backward passes. For the last pass coefficient γ_s is set to 1 to ensure that the output minorant is maximal. Figure 10 illustrates that this idea can perform well in practice.

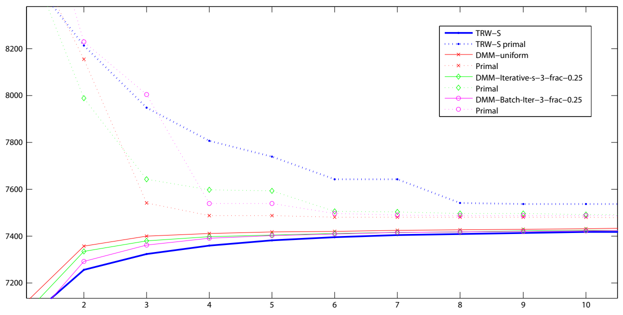


Figure 10: Same setting as in Figure 3. The new plots show that Iterative minorants are not as good as uniform but still perform very well. Parameter max_pass $= 3$ and $\gamma_s = 0.25$ were used. The Batch Iterative method (Batch-Iter) runs forward-backward iterations in a smaller range, which is more cache-efficient and is also performing relatively well in this example.

A.3.3 Hierarchical Minorants

The idea of hierarchical minorants is as follows. Let f be a one horizontal chain. We can break it into two sub-chains of approximately the same size, sharing a variable x_i in the middle. By introducing a Lagrange multiplier over this variable, we can decouple the two chains. The value of the Lagrange multiplier can be chosen such that both subchains have exactly the same min-marginals in x_i. This makes the split uniform in a certain sense. Proceeding so we increase the amount of parallelism and hierarchically break the chain down to two-variable pieces, for which the minorant is computed more or less straightforwardly. This is the method used to obtain all visual experiments in the paper. Its more detailed benchmarking is left for future work. We detail now the simplest case when the chain has length two, *i.e.*, the energy is given by $f_1(x_1) + f_{12}(x_1,x_2) + f_2(x_2)$. The procedure to compute the minorant is as follows:

- Compute $m^f_1(x_1)$ and let $\lambda_1 := m^f_1(x_1)/2$. *I.e.*, we subtract a half of the min-marginal in the first node.
- Recompute the new min-marginal at node 2: update

Algorithm 5: `Handshake`

```
Input: Energy terms $f_i$, $f_j$, $f_{ij}$, messages $\varphi_{i-1,i}(x_i)$ and $\varphi_{j,j+1}(x_j)$ ;
Output: Messages for decorrellated chains: $\varphi_{ji}(x_i)$ and $\varphi_{ij}(x_j)$ ;
   /* Message from $j$ to $i$ */
1  $\varphi_{ji}(x_i) := \texttt{Msg}_{ji}(f_j + \varphi_{j,j+1})$;
   /* Total min-marginal at $i$ */
2  $m_i(x_i) := \varphi_{i-1,i}(x_i) + f_i(x_i) + \varphi_{ji}(x_i)$;
   /* Share a half to the right */
3  $\varphi_{ij}(x_j) := \texttt{Msg}_{ij}(m_i/2 - \varphi_{ji})$;
   /* Bounce back what cannot be shared */
4  $\varphi_{ji}(x_i) := \texttt{Msg}_{ji}(-\varphi_{ij})$;
5  Procedure $\texttt{Msg}_{ij}(a)$
       Input: Unary cost $a \in \mathbb{R}^K$;
       Output: Message from $i$ to $j$;
6      return $\varphi(x_j) := \min_{x_i \in \mathcal{L}} \big[a(x_i) + f_{ij}(x_i, x_j)\big]$;
```

the message $\varphi_{12}(x_2) := \texttt{Msg}_{12}(f_1 - \lambda_1)$; Reassemble $m_2^{f-\lambda}(x_2) = \varphi_{12}(x_2) + f_2(x_2)$.

- Take this whole remaining min-marginal to the minorant: let $\lambda_2 := m_2^{f-\lambda}(x_2)$.
- Recompute the new min-marginal at node 1: update the message $\varphi_{21}(x_1) := \texttt{Msg}_{21}(f_2 - \lambda_2)$; It still may be non-zero. For example, if the pairwise term of f is zero we recover the remaining half of the initial min-marginal at node 1. Let $\lambda_1 \mathrel{+}= m_1^{f-\lambda}(x_1)$.

Importantly, the computation has been expressed in terms of message passing, and therefore can be implemented as efficiently. The procedure fro the two-node case is straightforwardly generalized to longer chains. Let ij be an edge in the middle of the chain. We compute left min-marginal at i, right min-marginal at j and then apply the `Handshake` procedure over the edge ij, defined in Algorithm 5. The procedure divides the slack between nodes i and j similarly to how it is described above for the pair. The result of this redistribution is encoded directly in the messages. The two subchains $1, \dots i$ and $j, \dots |\mathcal{V}|$ are "decorrellated" by the `Handshake` and will not talk to each other further during the construction of the minorant. The left min-marginal for subchain $j, \dots |\mathcal{V}|$ at node $j+1$ is computed using update (32) and so on until the middle of the subchain where a new `Handshake` is invoked. The minorant is computed at the lowest level of hierarchy when the length of the subchain becomes two. The structure of the processing is illustrated in Figure 11. It is seen that each level after the top one requires to send messages only for a half of nodes in total. Moreover, there is only a logarithmic number of level. It turns out that this procedure is not much more computationally costly than just computing min-marginals. For example, to restore left min-marginal for the subchain $j, \dots |V|$, in node $i+1$ we

We conjecture that while iterative minorants may transfer only a geometric fraction of min-marginals in some cases, the hierarchical minorant is only by a constant factor inferior to the uniform one.

```
[>>>>>>>>>>>>>>><<<<<<<<<<<<<<<]
[.......<<<<<<<][>>>>>>>.......]
[...<<<][>>>...][...<<<][>>>...]
[.<][>.][.<][>.][.<][>.][.<][>.]
[][][][][][][][][][][][][][][][]
```

Figure 11: Messages passed in the construction of the hierarchical minorant for a chain of length 32. From top to bottom: level of hierarchical processing. Symbols > and < denote message passing in the respective direction. Brackets `[]` mark the limits of the decorrellated sub-chains at the current level. Dots denote places where the previously computed messages in the needed direction remain valid and need not be recomputed. Places where the two opposite messages meet correspond to the execution of the `Handshake` procedure. The lowest level consists of 16 decorrellated chains of length 2 each.

A.4. Iteration Complexity

The bottleneck in a fast implementation of dual algorithms are the memory access operations. This is simply because there is a big cost data volume that needs to be scanned in each iteration plus messages have to be red and written in TRW-S as well as in out Algorithm 2 (dual variables λ). We therefore will assess complexity in terms of memory access operations and ignore the slightly higher arithmetic complexity of our minorants.

For TRW-S the accesses per pixel are:

- read all incoming messages (4 access);
- read data term (1 access);
- write out messages in the pass direction (2 accesses).

The cache can potentially amortize writing messages and reading them back in the next scan line, in which case the complexity could be counted as 5 accesses per pixel. However, currently only CPU cache is big enough for this, while multiprocessors in GPU have relatively small cache divided between many parallel threads.

For the iterative minorant we have 3 forward-backward passes reading the data cost, the reverse message and writing the forward message (3*2*3 accesses), the last iteration writes λ and not the message. Some saving is possible with a small cache set at a cost of more computations. Computing the hierarchical minorant as described in Figure 11 for a chain of length 2048, assuming that chunks of size 8 already fit in the fast memory (registers + shared memory) has the following complexity. Reading data costs and writing messages until length 8 totals to $2 + \log_2(2048/8)/2 = 6$ accesses. Reading messages is only required at `Handshake` points and needs to be counted only until reaching length 8. Writing λ adds one more access. These estimates are summarized in Table 2.

TRW-S	Iterative	Naive BCD	Hierarchical
7(5)	18(8)	5(4)	7

Table 2: Memory accesses per pixel in TRW-S and `Dual_MM` with variants of minorants. Naive BCD here means just computing min-marginals.

21st *Computer Vision Winter Workshop*
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

Touching without vision: terrain perception in sensory deprived environments

Vojtěch Šalanský*, Vladimír Kubelka*†, Karel Zimmermann*, Michal Reinstein*, Tomáš Svoboda*†

Abstract. *In this paper we demonstrate a combined hardware and software solution that enhances sensor suite and perception capabilities of a mobile robot intended for real Urban Search & Rescue missions. A common fail-case, when exploring unknown environment of a disaster site, is the outage or deterioration of exteroceptive sensory measurements that the robot heavily relies on—especially for localization and navigation purposes. Deprivation of visual and laser modalities caused by dense smoke motivated us to develop a novel solution comprised of force sensor arrays embedded into tracks of our platform. Furthermore, we also exploit a robotic arm for active perception in cases when the prediction based on force sensors is too uncertain. Beside the integration of hardware, we also propose a framework exploiting Gaussian processes followed by Gibb's sampling to process raw sensor measurements and provide probabilistic interpretation of the underlying terrain profile. In the final, the profile is perceived by proprioceptive means only and successfully substitutes for the lack of exteroceptive measurements in the close vicinity of the robot, when traversing unknown and unseen obstacles. We evaluated our solution on real world terrains.*

1. Introduction

Advances in robotic technology allow mobile robots to be deployed in gradually more and more challenging environments. However, real-world conditions often complicate or even prohibit adoption of classical approaches to localization, mapping, navigation, or teleoperation. When rescuers operate a UGV during joint experiments in the TRADR project[1], which develops novel software and technology for human-robot teams in disaster response efforts [1], we have to deal with such problems.

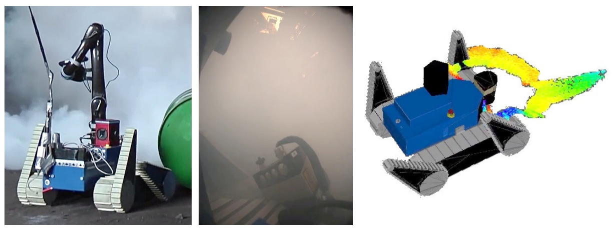


Figure 1. From left: UGV robot approaches smoke area; Example of visual information that the operator sees inside a cloud of smoke: a crop out from the omnidirectional camera (middle) and output of the laser rangefinder (rainbow-colored point cloud in the right half of the image). Laser beams are randomly reflected by smoke particles. The resulting 3D point cloud is just noise close to the robot.

One of the crucial fail-cases is the presence of dense smoke that blocks camera view and spoils laser measurements, creating false obstacles in front of the robot (Fig. 1). Without exteroceptive measurements, classical approaches to robot SLAM cannot be used. Localization can only be in the dead-reckoning sense and the operator of the robot has to rely solely on the maps created up to the point of the sensor outage. In an industrial environment consisting of many hazardous areas, driving blind can lead to damage or loss of the robot.

Therefore, we propose a combined hardware and software solution to predict the profile of terrain underneath and in front of the tracked robot. The algorithm exploits a prototype of a force sensor array installed inside a track of the robot, a robotic arm attached to the robot, proprioceptive measurements from joints and an inertial measurement unit (IMU), and information learned from a dataset of traversed terrains. The prototype of the force sensor (Fig. 2, 3) is suitable for tracked robots and is installed between rubber track and its support, allowing it to serve as

*Authors are with the Faculty of Electrical Engineering, Czech Technical University in Prague, {salanvoj, kubelvla, reinstein.michal, zimmerk, svobodat}@fel.cvut.cz

†Authors are with the Czech Institute of Informatics, Robotics and Cybernetics, Czech Technical University in Prague

[1] http://www.tradr-project.eu

a tactile sensor. The arm is used to measure height of terrain outside the reach of the force sensor as contact between the arm end-effector and the terrain. The height of terrain that cannot be measured directly is estimated by sampling from a joint probability distribution of terrain heights, conditioned by proprioceptive measurements (geometric configuration of the robot, torques in joints and attitude of the robot) and learned from a training dataset consisting of real-world examples of traversed terrains.

The estimates of terrain profile are used as a partial substitute for missing laser range-finder data that would reveal obstacles or serve as an input for adaptive traversability algorithm.

Our contribution is twofold: we designed a new force sensor suitable for tracked robots as well as an algorithm that uses proprioceptive and tactile measurements to estimate terrain shape in conditions that prohibit usage of cameras and laser range-finders. We extended this solution with robotic arm to deal with special cases when the predictions have too high uncertainty.

The rest of the paper is structured as follows: Section II concludes briefly the related work, Section III describes the hardware solution and Section IV the actual software. In Section V we present both qualitative and quantitative experimental evaluation and we conclude our achievements in Section VI.

2. Related work

The problem of terrain characterization primarily using proprioceptive sensors, but also by sonar/infrared range-finders and by a microphone is discussed in [2]. The authors exploit neural networks trained for each sensor and demonstrate that they are able to recognize different categories: gravel, grass, sand, pavement and dirt surface. More recent results come from legged robotics, in [3], Pitman-Yor process mixture of Gaussians is used to learn terrain types both in supervised and unsupervised manner based on force and torque features sensed in legs. In our work, we focus more on the actual terrain profile prediction, necessary for successful traversal.

Lack of sufficient visual information related to danger of collision with obstacles is addressed in [4]: decision whether it is safe to navigate through vegetation is based on wide-band radar measurements since it is impossible to detect solid obstacle behind vegetation from laser range-finder or visual data. Artificial whiskers offer an alternative solution; they mimic facial whiskers of animals and using them as a tactile sensor is a promising way to explore areas, which are prohibitive to standard exteroceptive sensors. Work of [5] presents a way to use array of actively actuated whiskers to discriminate various surface textures. In [6], similar sensor is used for a SLAM task. Two sensing modalities—the whisker sensor array and the wheel odometry are used to build a 2D occupancy map. Robot localization is then performed using particle filter with particles representing one second long "whisk periods". During these periods, the sensor actively builds local model of the obstacle it touches. Unfortunately, design of our platform does not allow using such whiskers due to rotating laser range-finder.

Relation between shape of terrain that we are interested in and configuration of the flippers is investigated in [7]. The authors exploit the knowledge about robot configuration and torques in joints to define a set of rules for climbing and descending obstacles not observed by exteroceptive sensors. We investigated this problem in [8] by introducing the adaptive traversability algorithm based on machine learning. We collected features from both proprioceptive and exteroceptive sensors to learn a policy that ensures safe traversal over obstacles by adjusting robot morphology. An idea of adding pressure sensors mimicking properties of human skin to feet of bipedal robots is presented in [9, 10]. These sensors can be used for measuring force distribution between the robotic foot and ground, or for terrain type classification. In tracked robots, caterpillar tracks can be further used to explore terrain, authors of [11] propose a novel distributed sensor that detects deflection of the track in contact points with terrain. Their sensor is especially suitable for chained tracks with rubber shoes. The prototype we present is more suitable for thin rubber tracks.

On contrary to the approaches exploiting only simple contact sensors, we extend our sensory suite with a robotic arm for further active perception for cases if necessary. Related to the active perception, relevant ideas and techniques come from the field of haptics. The work of [12] proposes to create models of objects in order to be able to grasp them. The idea is to complement visual measurements by tactile ones by strategically touching the object in areas with high shape uncertainty. For this purpose they use Gaussian processes (GP, [13]) to express the shape of the object. We take a similar approach: we choose parts of terrain to be explored by the robotic arm based on uncertainty of the estimate resulting from the sampling process (Sec. 4.3). Probabilistic approach to express uncertainty in touched points is also described in [14], where only tactile sensors of a robotic hand are used to reconstruct the shape of an unknown object. Active tactile terrain exploration can also lead to terrain type classification, as works of [15, 16] demonstrate.

3. Sensors

3.1. Sensors of the TRADR UGV

The *TRADR* UGV platform is equipped with both proprioceptive and exteroceptive sensors. Inertial measurement unit *Xsens MTi-G* (IMU) provides basic attitude measurements; all joints have angle encoders installed to reveal current configuration of the robot like flipper angles, and velocity of the caterpillar tracks. Electric currents to all motors are measured and translated into torque values. Visual information about the environment is acquired by an omni-directional *Point Grey Ladybug 3* camera accompanied by a rotating *SICK LMS-151* laser range finder that provides depth information. The laser range-finder is used to collect data that are processed to serve as ground truth for the terrain reconstruction purposes.

3.2. Prototype of force sensor

To obtain well-defined contact points with the ground, we decided to take advantage of the flippers that can reach in front of the robot and are designed to operate on dirty surfaces or sharp edges. The original mechatronics of the robot allows to measure torque in flipper servos and thus detect physical contact between flippers and the environment. To be able to locate the contact point on the flipper exactly, we designed a thin force sensor between the rubber track and its plastic support (see Fig. 2, 3). Since it is a first prototype, we use it only in one flipper and consider only symmetrical obstacles or steps. The sensor construction is a sandwich of two thin stripes of steel with *FSR 402* sensing elements between them which allows the rubber track to slide over it while measuring forces applied onto the track. There are six force sensing elements; the protecting sheet of steel distributes the force among them, the sensor is thus sensitive along its whole length.

The *FSR 402* sensing elements are passive sensors that exhibit decrease in resistance with increasing force; the force sensitivity range is $0.1 - 10\,\mathrm{N}$. To measure the resistance, we connect them in series with a fixed reference resistor forming a voltage divider. We apply 5 V to this divider and measure voltage on the reference resistor. We use an analog-to-digital converter expansion board for the Raspberry Pi computer to read the six voltages. We calibrate the voltage values for initial bias caused by the sandwich construction.

Figure 4 shows three examples of the sensor readings. The first case consists of a flipper touching flat floor. Although one would expect to see more or less equal distribution of the contact force along the flipper track, the torque generated by the flipper actually lifts the robot slightly and thus, most of the force con-

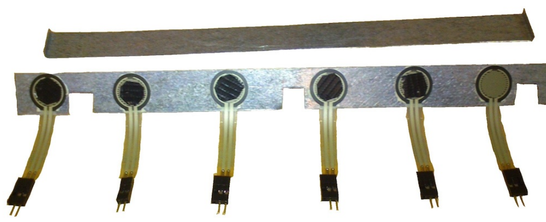
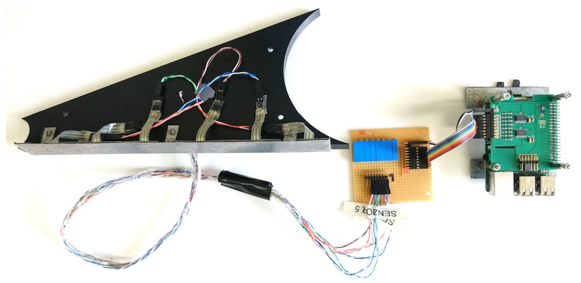


Figure 2. Prototype of the flipper force sensor: array of six sensing elements (FSR 402) is covered by a stripe of steel, forming a thin sensor that fits between the rubber track and the plastic track support. The stripe of steel protects the sensors from the moving rubber track and distributes measured force amongst them.



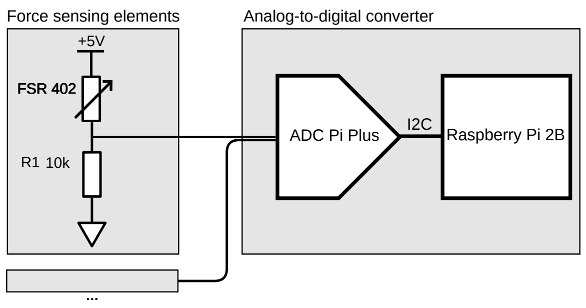


Figure 3. The sensor mounted to the plastic track support (top). The sensing elements are passive sensors that exhibit decrease in resistance with applied force. For each sensing element, we use a reference resistor to form a voltage divider; we obtain voltage inversely proportional to the resistance of the FSR 402 elements (bottom).

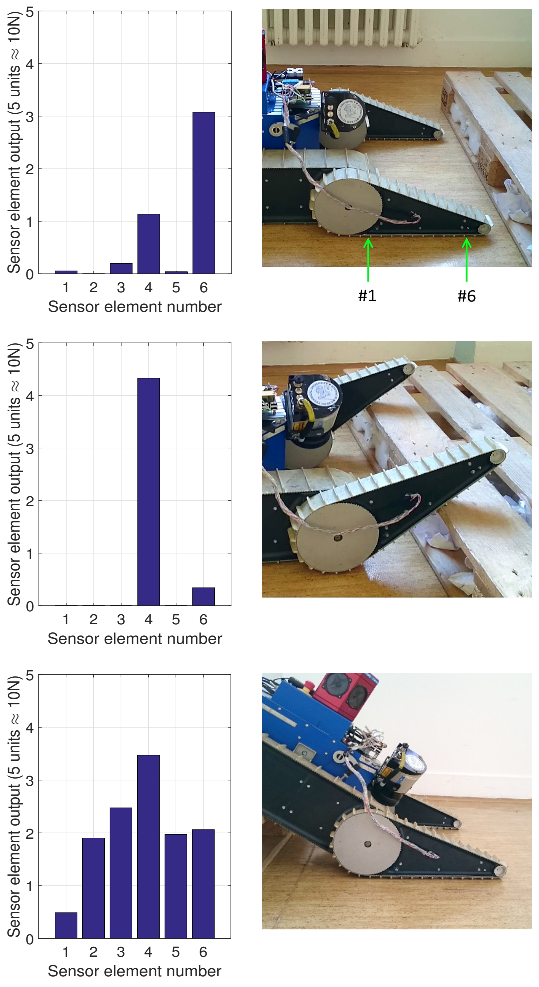


Figure 4. Examples of the force sensor readings. The plots on the left side show raw readings of each sensing element, only corrected for bias. The photos on the right side document the moments of the readings acquisition. See section 3 for discussion over the three example cases.

centrates at its tip (element n. 6). Compare this case with the third one (bottom), where the pose of the robot prohibits the lifting effect, and we therefore see the expected result. The second case (middle) shows an example of a touch in one isolated point.

3.3. Robotic arm

The UGV is equipped with a *Kinova Jaco* robotic arm[1], see Fig. 1 left. It is a 6-DOF manipulator (with one extra DOF in each finger) capable of lifting 1.5 kg. For our approach, it is used for tactile exploration of surroundings up to cca. 50 cm around the robot. For the terrain sensing, robotic arm holds a tool with a wooden stick—this setup protects its fingers from being broken when pushing against ground. It also allows the robot to measure the height of terrain in a chosen point by gradually lowering the arm until upsurge of actuator currents indicates contact with ground (there are currently no touch sensors) [17]. Accuracy of the measurement is 3 cm (standard deviation). However, the process of unfolding the arm, planning and execution of the desired motion and finally folding back to home position can easily take 45 s. Therefore, it is practical to use the arm for this purpose only in situations when the gain from the additional information overweights the cost of time spent to get it. In Section 4.4, we describe criterion for decision to use the arm.

[1]http://www.kinovarobotics.com/service-robotics/products/robot-arms

4. Terrain shape reconstruction

When robot is teleoperated operator's awareness is based on camera images and the 3D laser map. In the presence of smoke, both of these modalities are useless, see output of the operator console in the presence of smoke shown in Figure 1. We propose active tactile exploration mode (ATEM), in which flippers and robotic arm autonomously explores the terrain shape in close vicinity of the robot. Estimated terrain shape and expected reconstruction accuracy are eventually displayed to the operator.

If ATEM is requested by the operator, robot first adjusts flippers to press against the terrain and capture proprioceptive measurements. Then the initial probabilistic reconstruction of the underlying terrain shape is estimated from the captured data. If the reconstruction is ambiguous, the robotic arm explores the terrain height in the most inaccurate place. Eventually, the probabilistic reconstruction is repeated. As a result, reconstructed terrain shape with estimated variances is provided. The ATEM procedure is summarized in Algorithm 1. The rest of this section provides detailed description of particular steps.

4.1. Flipper exploration mode

As soon as the ATEM is requested, the robot halts driving and adjusts angles of front flippers towards ground until they reach an obstacle or the ground. They keep pressing against it by defined torque while vector of proprioceptive measurements $\mathbf{s}$ is captured. We measure: i) pitch of the robot (estimated from IMU sensor), ii) angles of flippers, iii) currents in flipper engines, and iv) 6-dimensional output of the force sensor.

```
Variables: $\mathbf{h}$ - vector of terrain bin heights,
           $\mathbf{v}$ - vector of height variances,
           $\mathbf{s}$ - vector of proprioceptive measurements.
while ATEM is requested do
    stop_robot;
    // Invoke flipper exploration mode
    // Section 4.1
    while torque_in_front_flippers < threshold do
        push_flippers_down;
    end
    $\mathbf{s}$ = capture_proprioceptive_measurements();
    // Perform kinematic reconstruction
    // Section 4.2
    $[\mathbf{h}, \mathbf{v}]$ = kinematic_reconstruction($\mathbf{s}$);
    // Perform probabilistic reconstr.
    // Section 4.3
    $[\mathbf{h}, \mathbf{v}]$ = probabilistic_reconstruction($\mathbf{h}, \mathbf{v}, \mathbf{s}$);
    // Invoke arm exploration
    // Section 4.4
    if any($\mathbf{v}$ > threshold) then
        $[\mathbf{h}, \mathbf{v}]$ = arm_exploration($\mathbf{h}, \mathbf{v}$);
        $[\mathbf{h}, \mathbf{v}]$ = probabilistic_reconstruction($\mathbf{h}, \mathbf{v}, \mathbf{s}$);
    end
    move_forward;
end
```

Algorithm 1: Active tactile exploration mode for terrain shape reconstruction.

4.2. Kinematic reconstruction

The terrain shape is modeled by Digital Elevation Map (DEM), which consists of eleven 0.1 m-wide bins. If there is only one isolated contact point sensed by the force sensor and the force surpasses experimentally identified threshold (see Fig. 4, second case), the height $\mathbf{h}_i$ of the terrain in the corresponding bin i is estimated by a geometric construction from known robot kinematics, using the attitude of the robot, configuration of joints and the position of the contact point on the flipper. Variance $\mathbf{v}_i$ for the corresponding force sensor is set to an experimentally estimated value. The remaining $\mathbf{h}_i$ and $\mathbf{v}_i$ values are set to non-numbers.

4.3. Probabilistic reconstruction

In the probabilistic reconstruction procedure, the vector of heights $\mathbf{h}$ and the vector of variances $\mathbf{v}$ are estimated by the Gibbs sampling [18]. Let us denote the set of all bins J and the set of all bins in which the reconstruction is needed by I (i.e. those which height was not estimated in the kinematic reconstruction procedure or measured by the robotic arm). We use the Gibbs sampling to obtain height samples $\mathbf{h}_I^k$, $k = 1 \dots K$ from the joint probability distributions $p(\mathbf{h}_I|\mathbf{h}_{J\setminus I}, s)$ of all missing heights $\mathbf{h}_I$. Missing heights $\mathbf{h}_I$ are reconstructed as the mean of generated samples, variances $\mathbf{v}_I$ are estimated as the variance of samples.

In the beginning, the missing heights $\mathbf{h}_I$ are randomly initialized. The k-th sample $\mathbf{h}_I^k$ is obtained by iterating over all unknown bins $i \in I$ and generating their heights $\mathbf{h}_i^k$ from conditional probabilities $p(\mathbf{h}_i|\mathbf{h}_{J\setminus i}, \mathbf{s})$. The conditional probability is modeled by Gaussian process [19, 13, 20] with a squared exponential kernel.

To train the conditional probabilities, we collected real-world trajectories with i) sensor measurements $\mathbf{s}^u$ and ii) corresponding terrain shapes $\mathbf{h}^u$ estimated from the 3D laser map for $u = 1 \dots U$. The i-th conditional probability $p(\mathbf{h}_i|\mathbf{h}_{J\setminus i}, \mathbf{s})$ is modeled by one Gaussian process learned from the training set $\{[(\mathbf{h}^1_{J\setminus i}, \mathbf{s}^1)^\top, \mathbf{h}^1_i], \dots, [(\mathbf{h}^U_{J\setminus i}, \mathbf{s}^U)^\top, \mathbf{h}^U_i]\}$.

Modeling the bin height probabilities as normal distributions is a requirement laid by the Gaussian process. However, it allows samples of the bin height that collide with the body of robot, which is of course physically impossible. We propose to use Gaussian distribution truncated by known kinematic constraints, in which are samples constrained by the maximal height that does not collide with the body of the robot. We discuss impact of this modification in the Section 5.

4.4. Active arm exploration

We use the robotic arm to measure the height of the terrain in bins the flippers cannot reach. The measurement taken by the robotic arm is reasonably accurate and precise but in its current state it takes about 45s to complete [17]. If the probabilistic reconstruction contains bins with variance $\mathbf{v}$ higher than a user-defined threshold, the robotic arm is used to measure the height in the most uncertain bin, i.e. the bin $j = \arg\max_i \mathbf{v}_i$. The height sensed in the given bin is then fixed and the probabilistic reconstruction process is repeated.

5. Experimental evaluation

In qualitative experiments, we focus on typical cases of terrain profile shapes and discuss performance of different settings of our algorithm. In quantitative experiments, we present performance statistics over the whole testing dataset.

The *training dataset* consists of 28 runs containing driving on flat terrain, approaching obstacles of

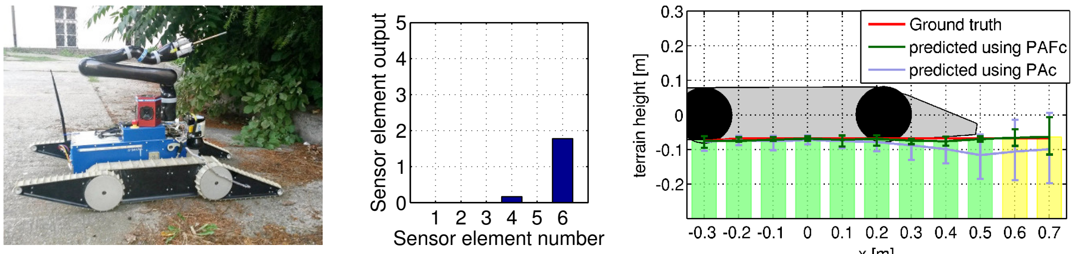


Figure 5. From left: photo of the robot on a concrete ground; measured forces; terrain reconstruction, the gray polygon indicates position of the robot and its flippers, thin red line is the ground truth—flat ground in this instance.

two different heights, traversing them and descending from them back to flat ground. Shape of obstacles selected for the dataset reflects the industrial accident scenario of the TRADR project - the environment mostly consists of right-angle-shaped concrete and steel objects. From the recorded runs, we have extracted approximately 1400 individual terrain profile measurements for training. The whole training dataset was recorded indoors on flat hard surfaces. The *testing dataset* was recorded outdoors and combines uneven grass, stone and rough concrete surfaces. It contains more complex obstacles with various heights (different from those seen in the training dataset). The testing dataset consists of more than 300 terrain profiles with the corresponding sensory data. Ground truth necessary for training and testing was created manually by sampling scans from the laser range-finder recorded during the experiments.

We compare four different algorithms for terrain profile prediction. The baseline approach [8] uses only the IMU sensor and angles of flippers, we call it PA (pitch + angle of flippers) for short. The second setup uses the same data and adds the probability of terrain height being adapted in the way described in Section 4.3. If the sampled height collides with the robot, the sample is set to the maximal possible height that is not in collision. The approach is called PAc (pitch + angle of flippers; constrained). The third approach adds the flipper force sensor; measured data are used in two ways. If the force measured by a sensor element exceeds a threshold (experimentally set on 2 units), then the height of the bin is computed from kinematics of the robot (pitch and flipper angles and position of the sensor element) and the bin is fixed and excluded from the Gibbs sampling step. It should be noted however, that the measured forces are used even if they are not bigger than the threshold – they are part of the proprioceptive data $\mathbf{s}$. The approach is called as PAFc (pitch + angle of flippers + flipper force sensor; constrained). The fourth approach adds direct terrain measurement: we simulate use of the robotic arm for measurements the terrain height in bins with high uncertainty [17]. The simulation means revealing the value of the bin captured in the ground truth, variance of the bin is then equal to the variance of the arm measurements. In the experiments shown in this paper we set the standard deviation threshold of Gibbs samples that leads to arm exploration to $0.06\,\mathrm{m}$. The fourth approach is called as PAFAc (pitch + angle of flippers + flipper force sensor + robotic arm; constrained).

5.1. Qualitative Evaluation

In the figures 5, 6 and 7, we present typical terrain profiles and robot actions: flat ground, two steps with different height, climbing up a step and stepping down of a step. We compare performance of two algorithms: i) PAc uses the kinematic constraints when sampling but does not use the force sensors (light blue line in the plots) ii) PAFc algorithm which uses the force sensors (green line and bars). The last two bars marked yellow in order to emphasize the predictions are learnt from training dataset and we do not have enough information to correct the predictions from the sensing by flippers.

We use mean of the (Gibbs) samples as the predicted value (connected by lines) and 0.1 and 0.9 quantiles for displaying dispersion of samples (errorbars). The point $(0, 0)$ coincides with the location of the IMU sensor inside the robot body. The depicted sketch of the robot: the pitch is estimated by IMU, flipper angle is directly measured. When the robot lies on a flat ground, Fig. 5, contact point is sensed by the sixth element. The force measurement reduces uncertainty mainly in positions $0.3 - 0.7\,\mathrm{m}$.

Climbing up a step cases are depicted in Fig. 6. The higher $0.28\,\mathrm{m}$ step obstacle is on top. The fifth sensor element measures the force that is bigger than threshold and the height in the bin 0.4 is fixed and not sampled. Note that algorithm PAc which does

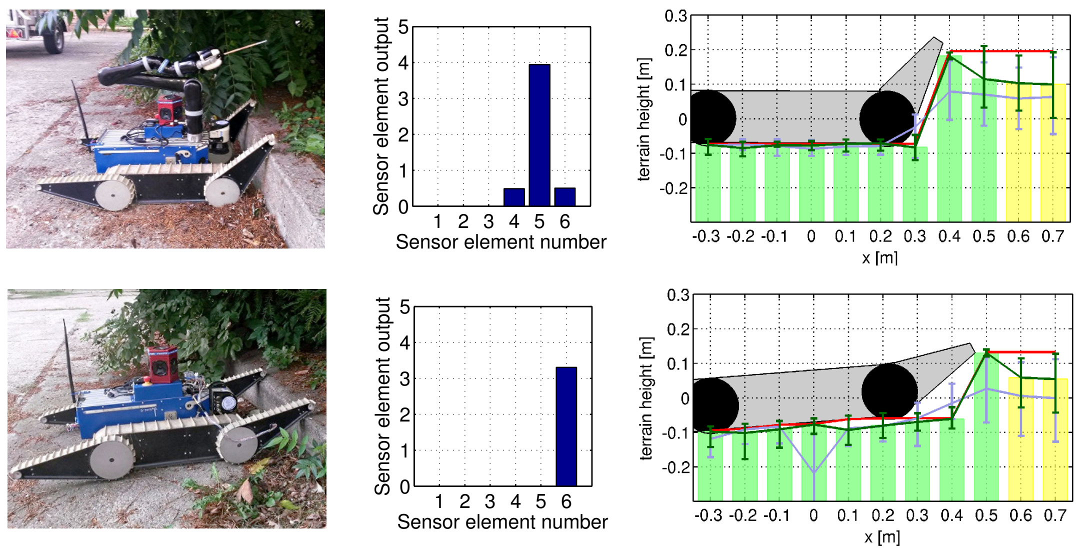


Figure 6. Top: 0.28 m step, bottom: 0.2 cm step. Note the reduced uncertainty for the PAFc – green line and errorbars. The top photo of the robot is flipped in order to preserve left-to-right orientation which should ease the visual comparison.

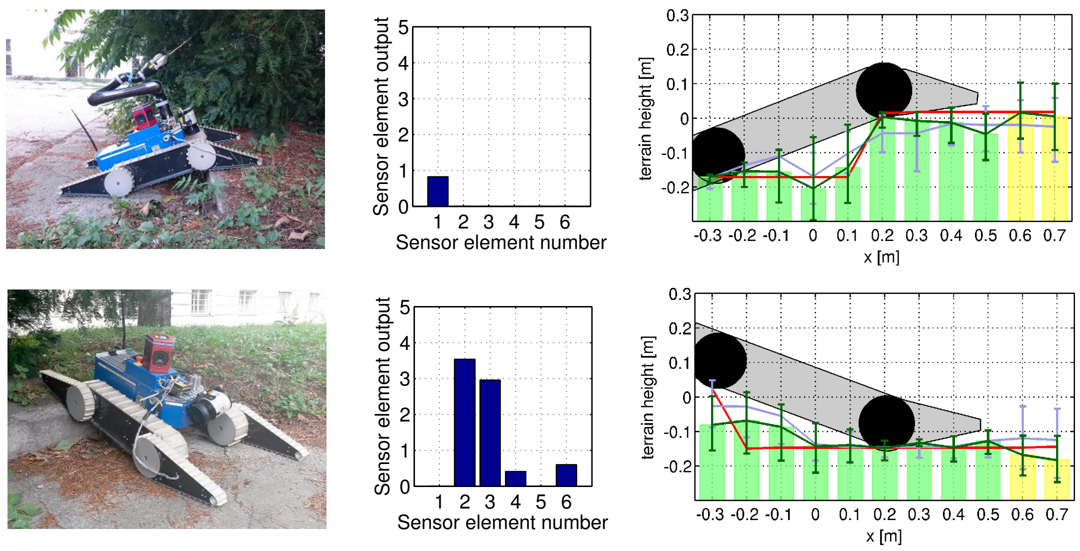


Figure 7. Top: climbing up a step; Bottom: stepping down of a step. When stepping down, the robot "hangs" on the rear flippers, not the main flippers.

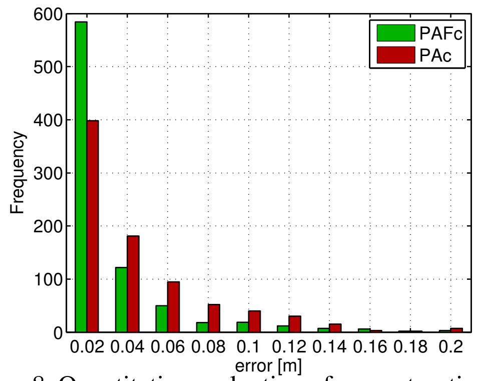


Figure 8. Quantitative evaluation of reconstruction quality in the places/bins that are under the flipper.

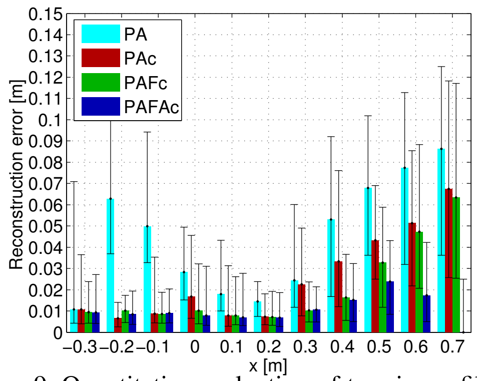


Figure 9. Quantitative evaluation of terrain profile reconstruction – for all the DEM bins. Median, 1st quartile and 3rd quartile of errors are shown

not use force sensor cannot predict the exact edge location. This fact is indicated by big dispersion of samples in bins 0.3 and 0.4. The second situation shown in Fig. 6 is the lower step. The height of the lower step $0.2\,\mathrm{m}$ was correctly measured by the sixth element of the force sensor.

Climbing up and stepping down cases are displayed in Fig. 7. Variances in the bins that are underneath the robot are high because we do not have enough information to estimate the correct heights. Still, the means are correct due to models learnt from the training data.

5.2. Quantitative Evaluation

As our metric of performance is the absolute error of estimated bin heights which is non-negative, we prefer to describe its statistical properties by quantile characteristics rather than by the means and standard deviations. The statistics are computed from the whole testing dataset - i.e. from more than 300 outdoor terrain profiles.

First, we measure the direct effect of the force measurement on the accuracy of the height estimates. The graph on Fig 8 shows the height error frequency of the DEM-bins that are underneath the front flipper. Note that the attribute “underneath the front flipper” is not fixed, it depends on the flipper angle. The force sensor indeed improves the accuracy over the using the flipper angle only.

The second experiment studies the statistics for all the DEM-bins individually, see Fig 9. Adding the kinematic contraint C naturally improves the estimates of the bins underneath the robot body $(-0.3 \ldots 0.2)$. Using the force sensors (PAFc) improves height estimates of the DEM-bins underneath the front flipper $(0.3 \ldots 0.5)$. The bins in front of the flippers, i.e. $(0.6$ and $0.7)$ are directly measurable only by the arm exploration. It is thus obvious that including the measurement by arm (PAFAc) has the dominant effect.

6. Conclusions

In this paper the aim was to demonstrate a combined hardware and software solution that enhances sensor suite and perception capabilities of our mobile robot intended for real Urban Search & Rescue missions. We focused our efforts on enabling proprioceptive terrain shape prediction for cases when vision and laser measurements are unavailable or deteriorated (such as in presence of a dense smoke). To evaluate our proposed solution experimentally, we designed and compared four algorithms—four possible approaches for proprioceptive terrain shape reconstruction: simple kinematics based approach, constrained kinematics, constrained kinematics with force sensors, and constrained kinematics with both force sensors and robotic arm—intended for special cases, where terrain prediction reaches very high uncertainty. From the presented qualitative and quantitative experimental evaluation we can clearly see that enhancing the sensor suite with force sensor array proves to be superior. The proposed algorithm, which combines Gaussian processes followed by Gibb’s sampling, was successfully implemented on-board the robot to process the raw force measurements and perform the actual terrain shape prediction in a probabilistic manner. We certainly do not claim this is the only and best way to perform such terrain prediction, but, it definitely serves as sufficiently robust and accurate proof of concept for intended deployment. As part of this concept, the integration of robotic arm for active perception in cases when the prediction based on force sensors is too uncertain proved to be important. For future work, we aim to embed additional force sensor arrays on all the four robot flippers and extend the terrain prediction algorithm accordingly.

ACKNOWLEDGMENT

The authors were supported by the Czech Science Foundation GA14-13876S, the EU-FP7-ICT-609763 TRADR project and the Czech Technical University grant SGS15/081/OHK3/1T/13.

References

[1] I. Kruijff-Korbayová, F. Colas, M. Gianni, F. Pirri, J. de Greeff, K. V. Hindriks, M. A. Neerincx, P. Ögren, T. Svoboda, and R. Worst, "TRADR project: Long-term human-robot teaming for robot assisted disaster response," *KI*, vol. 29, no. 2, pp. 193–201, 2015. 1

[2] L. Ojeda, J. Borenstein, G. Witus, and R. Karlsen, "Terrain characterization and classification with a mobile robot," *Journal of Field Robotics*, vol. 23, no. 2, pp. 103–122, 2006. 2

[3] P. Dallaire, K. Walas, P. Giguere, and B. Chaib-draa, "Learning terrain types with the pitman-yor process mixtures of Gaussians for a legged robot," in *Intelligent Robots and Systems (IROS)*, 2015. 2

[4] J. Ahtiainen, T. Peynot, J. Saarinen, and S. Scheding, "Augmenting traversability maps with ultra-wideband radar to enhance obstacle detection in vegetated environments," in *Intelligent Robots and Systems (IROS)*, 2013. 2

[5] J. Sullivan, B. Mitchinson, M. Pearson, M. Evans, N. Lepora, C. Fox, C. Melhuish, and T. Prescott, "Tactile discrimination using active whisker sensors," *IEEE Sensors Journal*, vol. 12, no. 2, pp. 350–362, 2012. 2

[6] M. Pearson, C. Fox, J. Sullivan, T. Prescott, T. Pipe, and B. Mitchinson, "Simultaneous localisation and mapping on a multi-degree of freedom biomimetic whiskered robot," in *Robotics and Automation (ICRA)*, 2013. 2

[7] K. Ohno, S. Morimura, S. Tadokoro, E. Koyanagi, and T. Yoshida, "Semi-autonomous control system of rescue crawler robot having flippers for getting over unknown-steps," in *Intelligent Robots and Systems (IROS)*, 2007. 2

[8] K. Zimmermann, P. Zuzanek, M. Reinstein, T. Petricek, and V. Hlavac, "Adaptive traversability of partially occluded obstacles," in *Robotics and Automation (ICRA)*, 2015. 2, 6

[9] H. Lee, "Development of the robotic touch foot sensor for 2d walking robot, for studying rough terrain locomotion," Master's thesis, University of Kansas, June 2012, mechanical Engineering. 2

[10] J. Shill, E. Collins, E. Coyle, and J. Clark, "Terrain identification on a one-legged hopping robot using high-resolution pressure images," in *Robotics and Automation (ICRA)*, 2014. 2

[11] D. Inoue, M. Konyo, K. Ohno, and S. Tadokoro, "Contact points detection for tracked mobile robots using inclination of track chains," in *International Conference on Advanced Intelligent Mechatronics*, 2008, pp. 194–199. 2

[12] M. Bjorkman, Y. Bekiroglu, V. Hogman, and D. Kragic, "Enhancing visual perception of shape through tactile glances," in *Intelligent Robots and Systems (IROS)*, 2013. 2

[13] C. K. Williams and C. E. Rasmussen, "Gaussian processes for regression," in *Advances in Neural Information Processing Systems 8*, D. Touretzky, M. Mozer, and M. Hasselmo, Eds. The MIT Press, 1996, pp. 514–520. 3, 5

[14] M. Meier, M. Schöpfer, R. Haschke, and H. Ritter, "A probabilistic approach to tactile shape reconstruction," *Robotics, IEEE Transactions on*, vol. 27, no. 3, pp. 630–635, 2011. 3

[15] J. Romano and K. Kuchenbecker, "Methods for robotic tool-mediated haptic surface recognition," in *Haptics Symposium (HAPTICS), 2014 IEEE*, 2014, pp. 49–56. 3

[16] D. Xu, G. Loeb, and J. Fishel, "Tactile identification of objects using Bayesian exploration," in *Robotics and Automation (ICRA)*, 2013. 3

[17] V. Šalanský, "Contact terrain exploration for mobile robot," Master's thesis, Czech Technical University in Prague, 2015, in Czech. 4, 5, 6

[18] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 6, pp. 721–741, 1984. 5

[19] A. O'Hagan and J. Kingman, "Curve fitting and optimal design for prediction," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 1–42, 1978. 5

[20] C. E. Rasmussen and H. Nickisch. Gpml matlab code. 5

21st *Computer Vision Winter Workshop*
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

Hessian Interest Points on GPU

Jaroslav Sloup, Michal Perd'och, Štěpán Obdržálek, Jiří Matas
Center for Machine Perception
Czech Technical University Prague
`sloup|perdom1|xobdrzal|matas @fel.cvut.cz`

Abstract.

This paper is about interest point detection and GPU programming. We take a popular GPGPU implementation of SIFT – the de-facto standard in fast interest point detectors – SiftGPU and implement modifications that according to recent research result in better performance in terms of repeatability of the detected points. The interest points found at local extrema of the Difference of Gaussians (DoG) function in the original SIFT are replaced by the local extrema of determinant of Hessian matrix of the intensity function.

Experimentally we show that the GPU implementation of Hessian-based detector (i) surpasses in repeatability the original DoG-based implementation, (ii) gives result very close to those of a reference CPU implementation, and (iii) is significantly faster than the CPU implementation. We show what speedup is achieved for different image sizes and provide analysis of computational cost of individual steps of the algorithm.

The source code is publicly available.

1. Introduction

A viewpoint-independent representation of objects in images is one of the fundamental problems in computer vision. A popular approach is to extract a set of local measurements, known as descriptors, at a sparse set of image locations. These locations are called interest points and their purpose is (as opposed to dense image sampling) to reduce the spatial domain of further computation, hence reducing the cost to obtain, and memory requirements to store, the image representation.

It follows that for an interest point extraction process to be practical it needs to repeatedly identify the same points on object surface when the viewpoint or environment (*e.g.* illumination) change. Establishing correspondences between interest points representing an object in multiple images is a building step for a multitude of computer vision tasks, including stereo or multi-view reconstruction, object recognition, and image search and retrieval.

These are the desirable qualities for which interest point detectors are evaluated:

- **Transformation Covariance**. The detected points should correspondingly 'follow' the object as it is depicted from different viewpoints. This paper concerns similarity-covariant detectors which follow 2D image locations (objects at different positions in the image), scales (objects at different distances) and 2D orientations (in-plane rotation of the objects). Affine detectors, which additionally follow out-of-plane 3D rotations, are not considered here.

- **Repeatability** of detected interest points. The percentage of the points detected at corresponding image locations when the viewpoint changes.

- **Accuracy** with which the interest points are located and their scales and orientations are estimated.

- **Coverage** of various visually different classes of objects.

- **Robustness** under image degradation – noise, motion blur, compression, out of focus images, etc.

- **Detection Speed**, the computational cost of the interest points detection.

One of the most popular interest point detection algorithms is still the Scale-Invariant Feature Transform (SIFT) proposed by David Lowe [8] in 2004.

It consistently ranks high on benchmarks in quality of detected points, but is computationally expensive, therefore unsuitable *e.g.* for real-time video processing. Many speedier approximations and alternatives were proposed, *e.g.* SURF [2], FAST [11] and ORB [12], or CenSurE [1] and SUSurE [3], which can detect interest points significantly faster than SIFT. But often at the expense of repeatability and accuracy.

The only widely used detector that in most tests scores higher in repeatability than SIFT is the so-called Hessian detector. In SIFT, points are identified at local minima or maxima of the Difference of Gaussians function, thence in presence of blob-like local image structures. In the Hessian detector, the points are located where the determinant of the Hessian matrix (a matrix of second-order partial derivatives) attains local extrema. Which occur either for blob-like (local maxima) or for saddle-like structures (local minima). Experiments show that the extrema of the determinant of Hessian are more repeatable and accurate than the extrema of the Difference of Gaussians, and, thanks to the additional detection of saddle points, the object coverage is generally also improved. The detection speed of Hessian is similar to that of the SIFT.

Taking advantage of the recent widespread availability of programmable graphic cards, execution time of many computer vision algorithms benefits if reimplemented for GPUs. Interest point detectors are no exception, a GPGPU (general-purpose GPU) SIFT implementation is available from [15, 14, 4]. The SIFTs are detected in real-time for moderately sized videos or images on a consumer-grade GPU, therefore there is now a large group of applications for which it is no longer necessary to sacrifice detection quality for execution speed.

We build upon the available GPU SIFT implementation [15] and extend it with several contributions. The Difference of Gaussians is replaced with the determinant of the Hessian matrix as the function of which extrema indicate presence of interest points. This improves repeatability, and coverage, of the detected points, as is experimentally demonstrated below. Selection of best K points (when ordered by magnitude of the determinant) is implemented in an early stage of the algorithm. If only a specific number of points is requested, it is faster to decide which these are early, on the GPU, before orientations are determined and descriptors computed. Additionally, the feature type (saddle, dark or white blob) is now part of the GPU code output. This is useful in follow-up matching – features of different types should not be considered for a correspondence.

Some of the functionality that was available in the original CPU SIFT implementation and omitted in the GPU version was reintroduced. We add the optional capability to compute orientations and descriptors only in $\langle 0, \pi \rangle$ range instead of $\langle 0, 2\pi \rangle$ by disregarding sign of the gradients involved, which is beneficial when matching images taken under significantly different illumination (day and night). The restriction that at each image location only at most two interest point orientations are detected was lifted. And the maximal number of iterations used for subpixel localization of a detected point is now configurable, the original Sift-GPU code allowed only a single iteration.

In the rest of the paper we quickly describe the SIFT detector and explain the relations and differences between the Laplacian operator, the Difference of Gaussians and the determinant of the Hessian matrix (Section 2). In Section 3 we sketch the GPU implementation and analyze the computational cost of individual components. Experiments in Section 4 show that the Hessian indeed achieves better performance than original SIFT and that the GPU and CPU implementations of Hessian give very similar results.

2. Laplacian of Gaussian, Difference of Gaussians and Determinant of Hessian Matrix

Let us consider a grayscale image to be a discretized form of an underlying real-valued continuous function $f(x, y) : \mathbb{R}^2 \to \mathbb{R}$. Its *Gaussian scale-space* representation $L(x, y, t) : \mathbb{R}^3 \to \mathbb{R}$ is then defined as

$$L(x, y; t) \ = g(x, y, t) * f(x, y)$$

where

$$g(x, y, t) = \frac{1}{2\pi t} e^{-\frac{x^2+y^2}{2t}}$$

is a rotationally symmetric 2D Gaussian kernel parametrized by variance $t = \sigma^2$, and where $*$ denotes convolution. Partial *Gaussian derivatives* of the image at a given scale t are then written as

$$\begin{aligned} L_{x^\alpha y^\beta}(\cdot, \cdot, t) &= \partial_{x^\alpha y^\beta} L(\cdot, \cdot, t) \\ &= (\partial_{x^\alpha y^\beta} g(\cdot, \cdot, t)) * f(\cdot, \cdot). \end{aligned}$$

The Hessian matrix for a given t is a square matrix of second-order partial derivatives

$$\mathbf{H} = \begin{pmatrix} \dfrac{\partial^2(f*g)}{\partial x^2} & \dfrac{\partial^2(f*g)}{\partial x\,\partial y} \\ \dfrac{\partial^2(f*g)}{\partial x\,\partial y} & \dfrac{\partial^2(f*g)}{\partial y^2} \end{pmatrix} = \begin{pmatrix} L_{xx} & L_{xy} \\ L_{xy} & L_{yy} \end{pmatrix}.$$

Let λ_1 and λ_2 denote the eigenvalues of the Hessian matrix. Laplacian (or the Laplace operator, the sum of second partial derivatives) of the Gaussian is then

$$\nabla^2 L = L_{xx} + L_{yy} = \lambda_1 + \lambda_2.$$

The Laplacian of Gaussian, appropriately normalized for different scales [7], is a basis for one of the first and also most common detector of blob-like interest points. Local scale-space extrema are detected that are maxima/minima of $\nabla^2 L$ simultaneously with respect to both space (x, y) and scale t [5]. In discrete domain, interest points are detected if the value of $\nabla^2 L$ at this point is greater/smaller than all values in its 26-neighbourhood. Locations of such points are covariant with translations, rotations and rescaling in the image domain. If a scale-space maximum is found at a point $(x_0, y_0; t_0)$ then after a rescaling of the image by a scale factor s there will be a corresponding scale-space maximum at $(sx_0, sy_0; s^2 t_0)$ [6].

The Laplacian of the Gaussian operator $\nabla^2 L(x, y, t)$ can be approximated [7] with a difference between two Gaussian-smoothed images at different scales t and $t + \Delta t$

$$\nabla^2 L(x, y; t) \approx \frac{t}{\Delta t} \left(L(x, y; t + \Delta t) - L(x, y; t) \right).$$

This approach is referred to as the Difference of Gaussians (DoG). In fashion similar to the Laplacian detector, interest points are detected as extrema in the 3D scale-space. The Difference of Gaussian is used in the SIFT algorithm [8].

Another differential interest point detector is derived from the determinant of the Hessian matrix $\mathbf{H}$

$$\det \mathbf{H} L(x, y; t) = (L_{xx} L_{yy} - L_{xy}^2) = \lambda_1 \lambda_2.$$

At image locations where the determinant is positive the image contains a blob-like structure. The Hessian matrix will there either be positive or negative definite, indicating presence of either bright or dark blobs. If the determinant of the Hessian matrix is negative, the matrix is indefinite, which indicates a saddle-like interest point [5].

The determinant of the Hessian operator has better scale selection properties under affine image transformations than the Laplacian operator or its Difference-of-Gaussians approximation [7]. It was also shown to perform significantly better for image-based matching using local SIFT-like or SURF-like image descriptors, leading to higher efficiency and precision scores [7]. In an approximation computed from Haar wavelets it is the basis for the interest point detector in SURF [2].

3. GPU Implementation and Computation Time Analysis

The GPU interest point implementation proceeds in steps shown in Figure 1. First, the input image is loaded and transferred to a GPU texture. The scale pyramid data structures, which make up the majority of the GPU memory required, are allocated once at the beginning, and reallocated only in case a bigger image is eventually processed later. The allocation typically takes several hundreds of milliseconds. Initial image upscaling by a factor of two, which is sometimes used in feature detection, is not performed. The scale space pyramid is then filled – a process that involves smoothing with Gaussian kernels with multiple std. deviations. Keypoints are detected as scale-space extrema of the determinant of the Hessian matrix and their locations are collected to a linear list. Optionally, the points are ordered by the response (the absolute value of the determinant) and only the top K points are kept for further processing. Keypoint orientations are then determined, with approximately 20% of the points ending with two or more orientations assigned. The points, now with the orientations, are again collected to a list and SIFT descriptors are computed.

Figure 2 shows the execution speed measured on three GPU cards. The photo shown on left, which represents a typical picture used in large-scale image retrieval tasks, was resized to eight different resolutions. Three CUDA-enabled graphics cards were tested: NVidia GeForce GT 730M (384 CUDA cores in 2 streaming multiprocessors, 1024MB DDR3 memory, 64-bit bus) is a representative of a common mobile/laptop GPU. NVidia GTX 750Ti (640 CUDA cores in 5 SMs, 2048MB GDDR5 memory, 128-bit bus) represents a gaming desktop card, and NVidia GTX Titan Black (2880 CUDA cores in 15xSMs, 6144MB GDDR5 memory, 384-bit bus) is a server card. Additionally, execution times of the reference

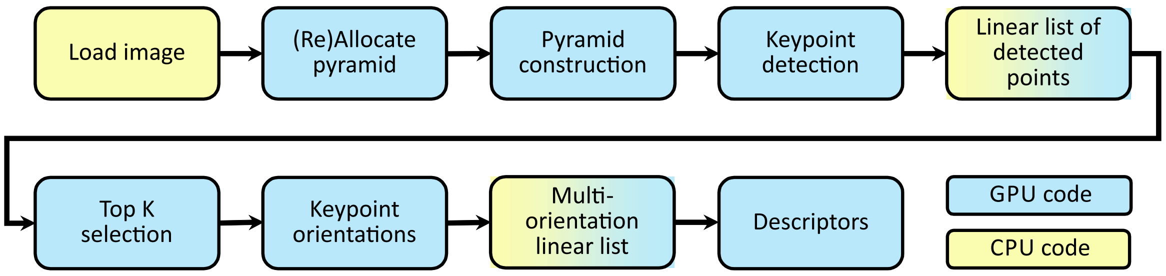
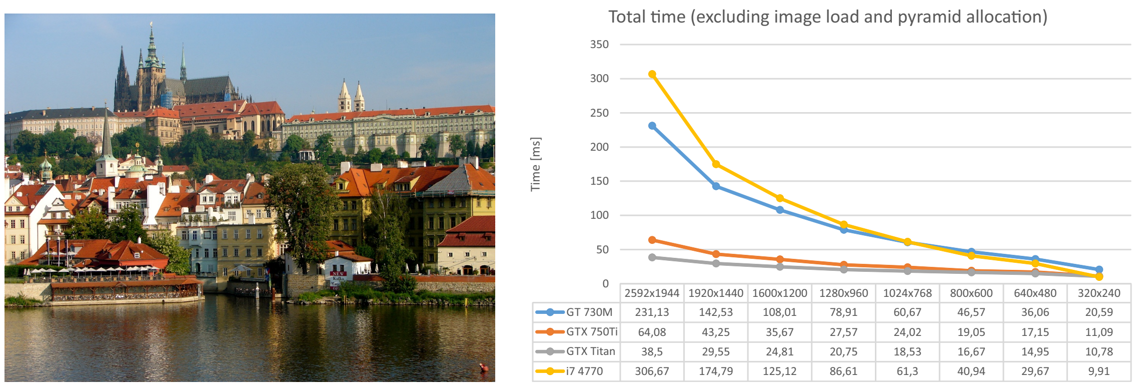


Figure 1. Block diagram of the computation pipeline. CPU code shown in yellow, GPU code in blue.



	2592x1944	1920x1440	1600x1200	1280x960	1024x768	800x600	640x480	320x240
GT 730M	231,13	142,53	108,01	78,91	60,67	46,57	36,06	20,59
GTX 750Ti	64,08	43,25	35,67	27,57	24,02	19,05	17,15	11,09
GTX Titan	38,5	29,55	24,81	20,75	18,53	16,67	14,95	10,78
i7 4770	306,67	174,79	125,12	86,61	61,3	40,94	29,67	9,91

Figure 2. Detection time for a test image (left) at eight different resolutions (right). Three GPUs were measured, together with a reference CPU implementation.

CPU implementation running on a current desktop CPU (i7 4470) are reported. While the mobile GPU is only slightly faster than the CPU, the other two GPU cards are roughly five and eight times faster.

Figure 3 shows a break down of load distribution over individual stages of the keypoint detection process (refer to Fig. 1). The analysis is shown for the desktop (left) and the mobile (right) GPUs. While the desktop card is about five times faster, the proportional distribution of the load is very similar.

Comparing the execution speed of the original Sift-GPU implementation (using the Difference of Gaussians) with our Hessian-based detector, see Fig. 4, we observe that the quality improvement demonstrated below in Experiments comes at no additional computational cost.

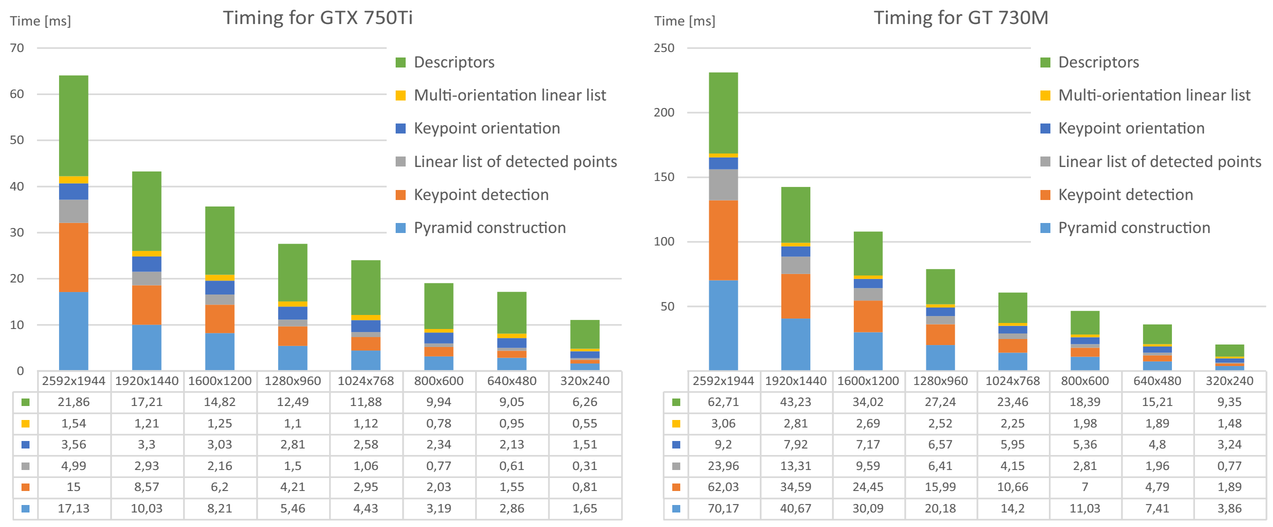
Finally in Figure 5 we show the timing when requesting only the best K keypoints. As expected, the stages preceding the top K selection are not affected. The stages following, orientation estimation and computation of the descriptor, take longer for more keypoints, although the increase is sublinear until the GPU processing power is saturated at around 8000 descriptors computed in parallel.

4. Experiments

The performance of the proposed GPGPU implementation of the determinant-of-Hessian detector was compared with other publicly available detectors, based on the Difference of Gaussians, multiscale Laplacian and the determinant of the Hessian matrix. In particular, we have evaluated Lowe's[8] original version of SIFT and its VLFeat re-implementation, CPU implementation of the Hessian and the Laplacian, and the original GPU code of SiftGPU. SURF detector [2], which is based on a fast approximation of the Hessian matrix, is also included. Two sets of experiments are presented: first one evaluating transformation invariance of the detectors in terms of repeatability and the number of correspondences, second one evaluating performance in a retrieval system.

4.1. Parameter Setting

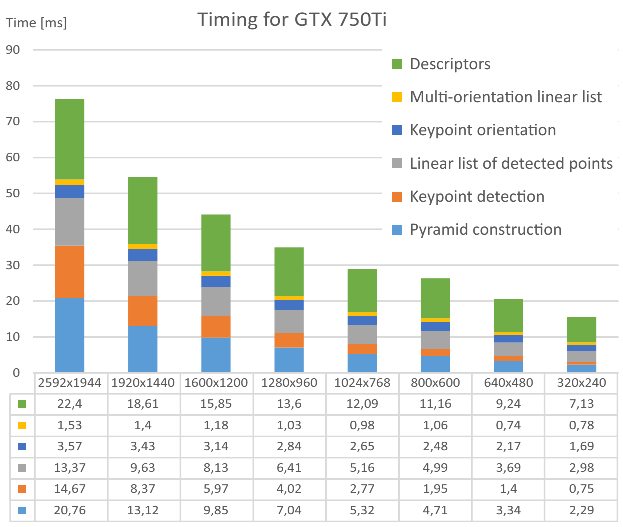
One of the advantages of the determinant of Hessian based detector is in responding to an additional type of local features – saddle points [6]. In our initial experiments on a large set of images, we observed that the number of saddle points in natural images is about the same as the number of bright and dark blobs together. Therefore the Hessian gives roughly twice as many points as the Laplacian/DoG



	2592x1944	1920x1440	1600x1200	1280x960	1024x768	800x600	640x480	320x240
🟩	21,86	17,21	14,82	12,49	11,88	9,94	9,05	6,26
🟨	1,54	1,21	1,25	1,1	1,12	0,78	0,95	0,55
🟦	3,56	3,3	3,03	2,81	2,58	2,34	2,13	1,51
⬜	4,99	2,93	2,16	1,5	1,06	0,77	0,61	0,31
🟧	15	8,57	6,2	4,21	2,95	2,03	1,55	0,81
🟦	17,13	10,03	8,21	5,46	4,43	3,19	2,86	1,65

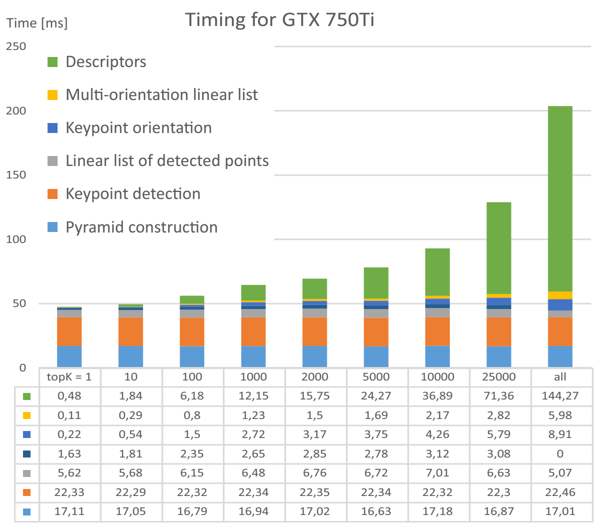
	2592x1944	1920x1440	1600x1200	1280x960	1024x768	800x600	640x480	320x240
🟩	62,71	43,23	34,02	27,24	23,46	18,39	15,21	9,35
🟨	3,06	2,81	2,69	2,52	2,25	1,98	1,89	1,48
🟦	9,2	7,92	7,17	6,57	5,95	5,36	4,8	3,24
⬜	23,96	13,31	9,59	6,41	4,15	2,81	1,96	0,77
🟧	62,03	34,59	24,45	15,99	10,66	7	4,79	1,89
🟦	70,17	40,67	30,09	20,18	14,2	11,03	7,41	3,86

Figure 3. Execution time of individual stages of the computation pipeline (refer to Fig. 1), evaluated at several image resolutions, with a default threshold on the detector response. The desktop GPU is about five times faster than the mobile GPU, but the relative load distribution between individual stages is virtually identical. Also the relation of the execution speed and image resolution is similar.



	2592x1944	1920x1440	1600x1200	1280x960	1024x768	800x600	640x480	320x240
🟩	22,4	18,61	15,85	13,6	12,09	11,16	9,24	7,13
🟨	1,53	1,4	1,18	1,03	0,98	1,06	0,74	0,78
🟦	3,57	3,43	3,14	2,84	2,65	2,48	2,17	1,69
⬜	13,37	9,63	8,13	6,41	5,16	4,99	3,69	2,98
🟧	14,67	8,37	5,97	4,02	2,77	1,95	1,4	0,75
🟦	20,76	13,12	9,85	7,04	5,32	4,71	3,34	2,29

Figure 4. Execution time of the original Sift-GPU code, evaluated at several image resolutions. Compare to the timing of our Hessian-based detector on the same hardware (Fig. 3 left). The improved qualitative performance, demonstrated in Section 4, comes with a negligible computational cost.



	topK = 1	10	100	1000	2000	5000	10000	25000	all
🟩	0,48	1,84	6,18	12,15	15,75	24,27	36,89	71,36	144,27
🟨	0,11	0,29	0,8	1,23	1,5	1,69	2,17	2,82	5,98
🟦	0,22	0,54	1,5	2,72	3,17	3,75	4,26	5,79	8,91
🟦	1,63	1,81	2,35	2,65	2,85	2,78	3,12	3,08	0
⬜	5,62	5,68	6,15	6,48	6,76	6,72	7,01	6,63	5,07
🟧	22,33	22,29	22,32	22,34	22,35	22,34	22,32	22,3	22,46
🟦	17,11	17,05	16,79	16,94	17,02	16,63	17,18	16,87	17,01

Figure 5. Execution times when a limited number of K best points is requested. Computed on the full size 2592x1944 image without a threshold on detector response. As expected, the processing time of the steps preceding the top K selection are not affected, while the later steps, most importantly the computation of the descriptor, scale with the number of points requested.

detectors, if detector configurations and thresholds are kept the same. To take an advantage of these additional points while keeping the representations comparable in size for the experiments, the detected points in each image were ordered by the absolute response value of the detector and the best 1000, 2000 and 4000 points were selected for evaluation. Finally, to diminish the slight differences in detection of dominant orientation, the orientations were fixed to vertical in the retrieval experiment, and were not used in the detector repeatability experiment.

4.2. Datasets and Evaluation Protocols

A standard benchmark protocol and dataset for evaluation of covariant interest point detectors was proposed by Mikolajczyk et al. [9]. It consists of eight sets, each of six images, with an increasing effect of image distortions: camera viewpoint, image scale, isotropic blur, underexposure and image compression. We have selected one scene with each distortion. Ground truth transformations are known, relating reference images of each set to all other images in that set. The transformations are used to compute repeatability scores by considering the *overlap error* ϵ of all pairs of detected points:

$$\epsilon(R_{\mathbf{E}_1}, R_{\mathbf{E}_2}) = 1 - \frac{R_{\mathbf{E}_1} \cap R_{\mathbf{H}_{12}^\top \mathbf{E}_2 \mathbf{H}_{12}^{-1}}}{R_{\mathbf{E}_1} \cup R_{\mathbf{H}_{12}^\top \mathbf{E}_2 \mathbf{H}_{12}^{-1}}},$$

where $R_\mathbf{E}$ represents the elliptic region defined by $x^\top R_\mathbf{E} x = 1$ and $\mathbf{H}_{12}$ is the known homography between the reference 1 and the test image 2. To compensate for different sizes of regions from different detectors, a scale factor is applied such that a region $R_{\mathbf{E}_1}$ is transformed to a normalized size (equivalent to a radius of 30 pixels). Before evaluating the overlap error, region $R_{\mathbf{E}_2}$ is scaled using the same factor.

The image retrieval performance was tested using the Oxford buildings dataset and protocol defined by Philbin et al. [10]. In short, five queries are defined for each of eleven landmarks in Oxford, and a ground truth shortlist of positive examples is given. For each query an average precision (AP) is computed as the area below the precision-recall curve. Finally, a mean AP (mAP) is reported for the whole set of 55 query images.

4.3. Evaluation of detectors Repeatability

Repeatability is of one of the important properties of interest point detectors. It is a measure that approximates the probability of the point redetection given the distortion between images. The detectors should be configured to provide comparable numbers of points to make the assessment fair. The repeatability score is complemented with the absolute number of corresponding points detected – the predicted upper bound of the matching problem. Figures 6, 7 and 8 show the measured scores when the number of detected points was limited to 1000, 2000, and 4000 respectively.

We observe that all the three DoG-based detectors (original Lowe's, from VLFeat and SiftGPU) perform virtually the same, as do the two Hessian-based detectors (CPU and GPU implementations). This strongly indicates that the measured performance is indeed inherent of the methods and not of a particular implementation. We also see that the Hessian performs in most cases better than the Laplacian and its DoG approximation. The additionally detected saddle points complement the blobs well and provide valuable correspondences between the images. With the exception of image blur, the fast but approximate SURF performs slightly worse than the other methods.

4.4. Evaluation in Image Retrieval

The repeatability of a detector predicts its pairwise matching potential. To assess the discrimination ability of a coupling of a detector (DoG, Laplacian, or Hessian) with a descriptor (SIFT), a large-scale image retrieval experiment was performed. The Oxford building dataset with about 5000 images was used. Each detector was again run in three configurations, requesting at most 1000, 2000, resp. 4000 best interest points. The SIFT descriptor was computed from a local neighborhood around each point and stored. As there are no significant orientation changes in the dataset, orientation of the interest points was fixed as vertical in this experiment. The measurement region sizes – radius of the SIFT index w.r.t. the detected scale of a interest point – were kept on their default values: 6.0 for DoG detectors (Lowe, VLFeat), 5.2 for Laplacian and CPU and GPU Hessian. The reasoning behind this is that the DoG detectors return slightly smaller (5-10%) intrinsic scale, determined by the smaller of the two subtracted Gaussians.

A standard Bag of Words (BoW) approach with and without Spatial Verification (SV) was used [13, 10]. SIFT descriptors were quantized into three different vocabularies for each detector, with: 500k visual words for 1000 points/image, and 1M visual words for 2000 resp. 4000 interest points per image. The TF-IDF scoring in an efficient inverted index was used to get the BoW ranking. The spatial verification estimated a similarity transformation between the query and each of the top 1000 ranked images. Finally, images were re-ranked based on number of correspondences. The ranking for each query was evaluated using Oxford buildings protocol and an mean Average precision computed as defined in [10].

The results are summarized in Table 1. The Hessian detectors consistently outperformed both the Laplacian and the Difference of Gaussians, regardless the size of the representation. Particularly for the highest number of interest points per image (4000), where both DoG implementations were struggling to deliver this many points, their performance dropped. Thus we can conclude that the complementary saddle points detected by the Hessian detector consistently

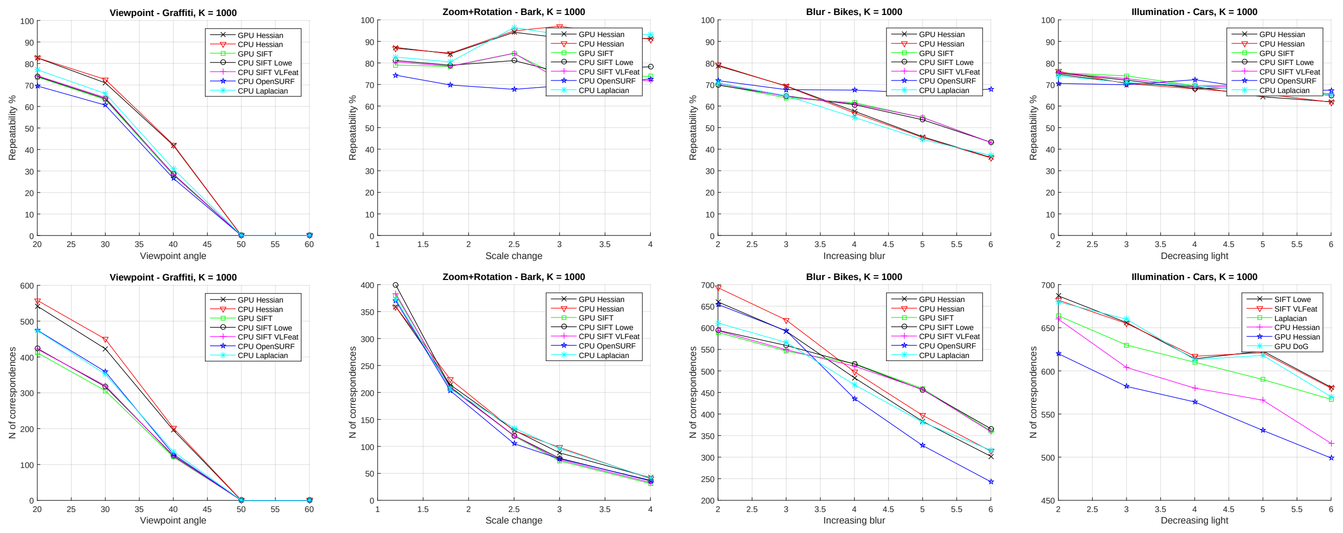


Figure 6. Repeatability score and number of correspondences on image sequences with (from left to right): a significant view angle change, scale change, image blur and exposure change. Number of features per image was limited to the best 1000 according to absolute response value.

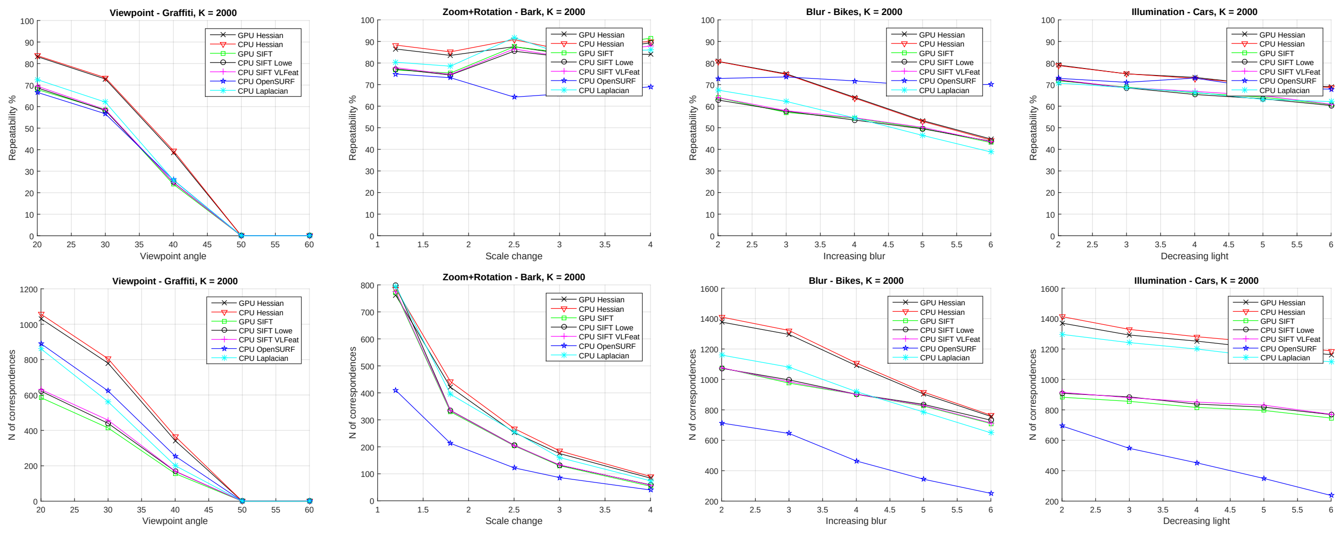


Figure 7. Repeatability score and number of correspondences on image sequences with (from left to right): a significant view angle change, scale change, image blur and exposure change. Number of features per image was limited to the best 2000 according to absolute response value.

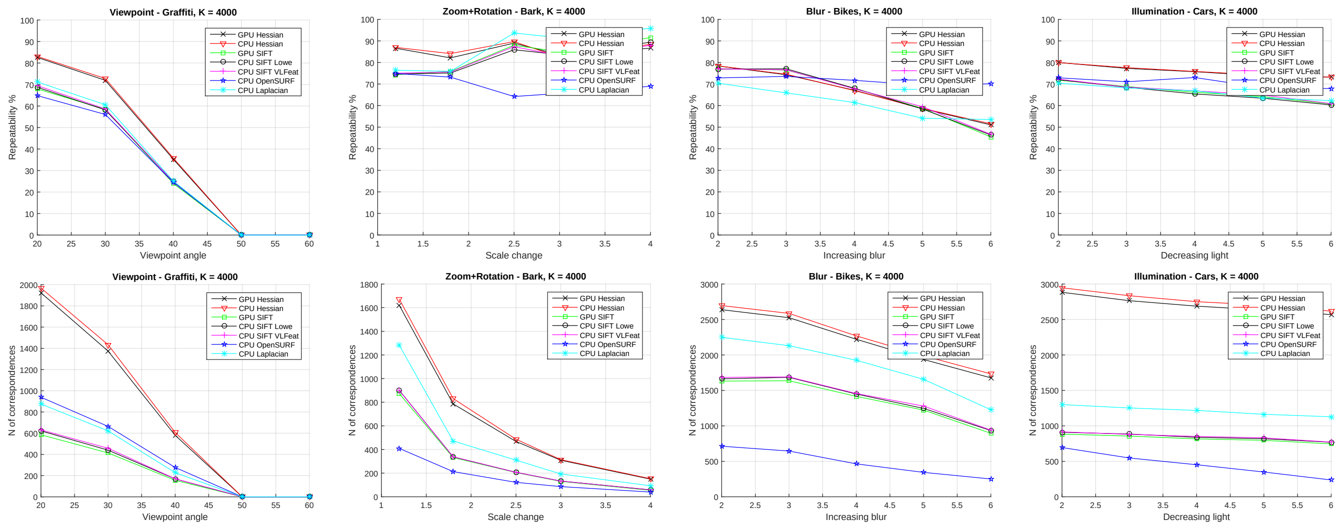


Figure 8. Repeatability score and number of correspondences on image sequences with (from left to right): a significant view angle change, scale change, image blur and exposure change. Number of features per image was limited to the best 4000 according to absolute response value.

Method	Max.feat.	Lowe DoG	VLFeat DoG	CPU Laplacian	CPU Hessian	GPU Hessian
BoW	1000	0.551	0.512	0.572	0.584	0.579
	2000	0.517	0.547	0.568	0.625	0.629
	4000	0.558	0.585	0.617	0.643	0.615
BoW+SV	1000	0.590	0.554	0.601	0.627	0.621
	2000	0.584	0.594	0.617	0.675	0.678
	4000	0.639	0.650	0.692	0.716	0.699

Table 1. Image retrieval experiment. The Bag of Words (BoW) method with and without Spatial Verification (SV) was evaluated with different interest point implementations. Features were limited to best 1000, 2000 resp. 4000 points per image based on detector's response. The values in the table are the measured mean average precisions, defined in [10].

improve the retrieval performance.

5. Conclusion

We have implemented an interest point detector based on the determinant of the Hessian matrix. Such a detector was previously shown, and the observation was confirmed in our experiments, to be superior in the quality of detected points to commonly used detectors based on the Difference of Gaussians. Starting with a publicly available GPU implementation of SIFT detector, we have implemented several modifications and experimentally verified that the performance indeed improved. The implementation, which is in CUDA for compatible NVidia graphics cards, was published and made available.

Acknowledgements

The authors were supported by Toyota Motor Europe.

References

[1] M. Agrawal, K. Konolige, and M. R. Blas. Censure: Center surround extremas for realtime feature detection and matching. In D. A. Forsyth, P. H. S. Torr, and A. Zisserman, editors, *ECCV (4)*, volume 5305 of *Lecture Notes in Computer Science*, pages 102–115. Springer, 2008. 2

[2] H. Bay, A. Ess, T. Tuytelaars, and L. van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding (CVIU)*, 110(3):346–359, June 2008. 2, 3, 4

[3] M. Ebrahimi and W. W. Mayol-Cuevas. SUSurE: Speeded Up Surround Extrema feature detector and descriptor for realtime applications. pages 9–14, Aug. 2009. 2

[4] H. Fassold and J. Rosner. A real-time gpu implementation of the sift algorithm for large-scale video analysis tasks. In *IS&T/SPIE Electronic Imaging*, pages 940007–940007. International Society for Optics and Photonics, 2015. 2

[5] T. Lindeberg. *Scale-Space Theory in Computer Vision*. Kluwer, 1994. 3

[6] T. Lindeberg. Feature detection with automatic scale selection. *IJCV*, 30(2):79–116, 1998. 3, 4

[7] T. Lindeberg. Image matching using generalized scale-space interest points. *Journal of Mathematical Imaging and Vision*, 52(1):3–36, 2015. 3

[8] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal on Computer Vision*, 20(2):91–110, 2004. 1, 3, 4

[9] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *IJCV*, 65(1-2):43–72, 2005. 5

[10] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007. 6, 8

[11] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In A. Leonardis, H. Bischof, and A. Pinz, editors, *Computer Vision ECCV 2006*, volume 3951 of *Lecture Notes in Computer Science*, pages 430–443. Springer Berlin Heidelberg, 2006. 2

[12] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571, Nov 2011. 2

[13] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. volume 2, pages 1470–1477, 2003. 6

[14] M. Soltan Mohammadi and M. Rezaeian. Siftcu: An accelerated cuda based implementation of sift. In *Third Symposium on Computer Science and Software Engineering, Sharif University, Tehran*, volume 3, 2013. 2

[15] C. Wu. SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). `http://cs.unc.edu/~ccwu/siftgpu`, 2007. 2

21st *Computer Vision Winter Workshop*
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

BaCoN: Building a Classifier from only N Samples

Georg Waltner Michael Opitz Horst Bischof
Institute for Computer Graphics and Vision
Graz University of Technology, Austria
{waltner, opitz, bischof}@icg.tugraz.at

Abstract. *We propose a model able to learn new object classes with a very limited amount of training samples (*i.e. 1 *to* 5*), while requiring near zero runtime cost for learning new object classes. After extracting Convolutional Neural Network (CNN) features, we discriminatively learn embeddings to separate the classes in feature space. The proposed method is especially useful for applications such as dish or logo recognition, where users typically add object classes comprising a wide variety of representations. Another benefit of our method is the low demand for computing power and memory, making it applicable for object classification on embedded devices. We demonstrate on the Food-101 dataset that even one single training example is sufficient to recognize new object classes and considerably improve results over the probabilistic Nearest Class Means (NCM) formulation.*

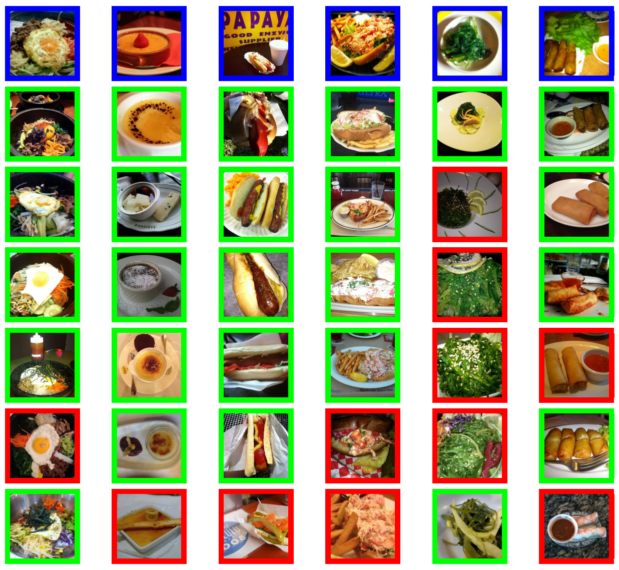


Table 1. One-shot learning results: The top row shows the used training sample (blue), the other rows are the first 6 results where the our proposed method yields different results than the probabilistic NCM version. Green framing indicates our improved NCM version is correct, while red stands for the opposite. From left to right: bibimap, creme brulee, hot dog, lobster roll sandwich, seaweed salad, spring rolls.

1. Introduction

With recent advances in object recognition [7], off-the-shelf features which are learned from a large number of annotated images have become freely available. As datasets grow, it will become increasingly computationally demanding to extend models built from these features to new object classes. Consider images of different food types - Japanese and European users will have quite different imaginations of an average lunch meal. Adapting a pretrained classifier to recognize meals that have not been seen during the training procedure is desirable. Similar to cognitive capabilities of humans, who can learn new classes from only very few samples, we aim for a computer vision system where new classes can be added incrementally. In this work, we consider classification methods which can integrate previously unseen classes from a single (one-shot) or a small number (n-shot) of training samples. The main purpose of these methods is to recognize new object classes from a very limited number of training samples. This is especially useful for open-ended recognition scenarios, such as logo detection or food recognition, where the number of object classes steadily grows during the life cycle of an object recognition system. However, integrating new classes in a classifier pretrained on different classes is not straightforward. On the one hand a new class often exhibits large variations, on the other hand the classifier trained on seen classes may not be capable to generalize to new classes. Retraining state-of-the-art classifiers such as CNN every time after such class additions leads to

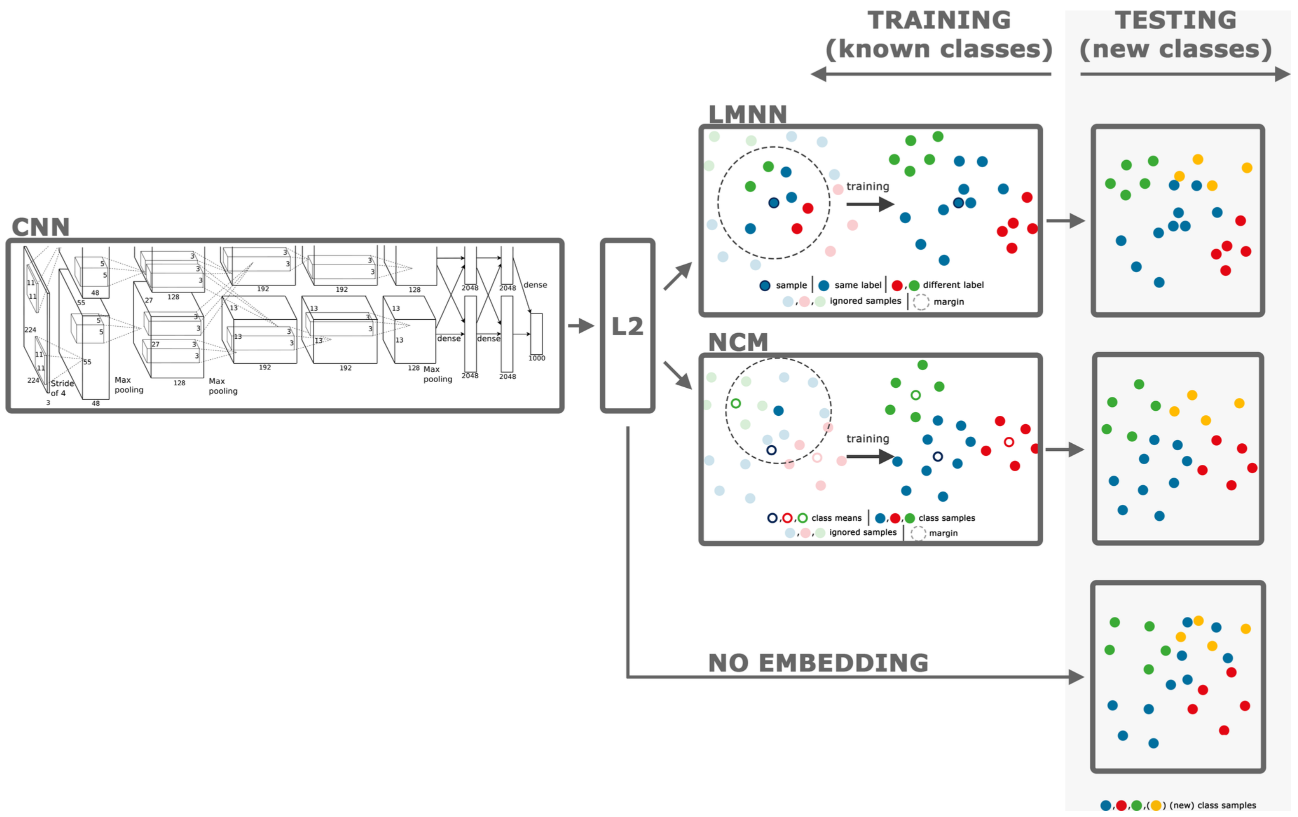


Figure 1. Overview of our method: After extracting CNN features from a fine-tuned CNN, we normalize the feature vectors and learn NCM or LMNN embeddings. These embeddings separate the classes in a way, so that newly added classes can be inserted without much loss in overall accuracy compared to insertion without a trained embedding. We leave the CNN and the trained embeddings fixed when adding new classes.

high accuracy, but is computationally inefficient and requires a significant amount of memory. For practical application this is often prohibitive, especially for embedded systems. An ideal system should therefore be able to integrate new object classes from few very seen samples on the fly; without the need for time- and memory-consuming retraining of the classifier and negligible performance loss.

2. Related Work

One approach in related works considering n-shot settings is the use of Bayesian learning, where probabilistic estimates are used to extend the algorithms to new classes. For example, [3] fit probabilistic density functions as category models and use them as prior knowledge for new classes, while using one or more samples for generation of the posterior model of the new class. Hierarchical Bayesian models are used by [12], where super-categories are automatically discovered based on available classes and serve as prior information to incorporate new classes. In [8], authors investigate one-shot learning of characters using Hierarchical Bayesian Program Learning (HBPL), where characters are modeled as a composition of primitives under certain causality constraints. Another approach for extension to new classes or categories is the use of attributes [9, 11], that can be seen as semantic descriptors which are shared by multiple classes. New classes are then added by generating a semantic description of the new class via attributes. Attribute-based approaches have been used for animal categorization and recognition [9] or for human-nameable visual attributes [11]. Similar to our idea of learning an optimal embedding, [6] statistically infers a Mahalanobis distance metric on similar and dissimilar feature pairs. In [16], a classifier is trained discriminatively for nearest prototype classification. Another distance metric learning approach was presented in [15], where the authors propose Large Margin Nearest Neighbor (LMNN) for classification. The LMNN classifier learns a Mahalanobis metric, so that same class samples are contracted and samples from different classes are pushed apart from each other. These methods are able to generalize to previously unseen samples, but in contrast to our approach they do not regard insertion of previously unseen object classes.

We employ a Nearest Class Mean (NCM) classifier [10, 14] for classification. Object classes in NCM classifiers are represented by the mean feature vector of the corresponding class samples and can be easily extended to new classes by computing the mean over newly added training samples. Our method learns discriminative embeddings to better separate the classes in feature space. Other than the probabilistic NCM approach of [10], we use CNN features. We propose the hinge loss for optimization and show that this improves overall accuracy. Additionally, we show how to robustify the learned NCM embeddings. In contrast to other approaches for one-shot and n-shot learning, our method does not need access to the full dataset as we do not employ classifier retraining, enabling the use of our system for embedded platforms like smartphones, where computing power and storage is limited. Furthermore, the most one-shot algorithms are Bayesian methods and use prior knowledge from the training data to generate posterior probabilities for new classes. We do not model such probabilities, but rely on the learned feature embeddings only. Figure 1 gives an overview of our method: We use l_2-normalized CNN features and learn additional layers that embed the features in an optimal way. After that we add new classes to evaluate the incremental learning capability of our classifier. Table 1 shows one-shot learning results for some classes of the Food-101 dataset [2].

3. One-Shot and N-Shot Classification

In the n-shot classification setting the classifier extends to new classes from a very limited number of samples (*i.e.* 1 to 5). NCM classifiers store a mean vector for each object class they recognize. This has the advantage that recognition of new classes can be incorporated by simply computing mean vectors for these classes. Mean vectors can be efficiently computed online, eliminating the need of explicitly storing feature vectors of all training samples that the class mean originates from. For one-shot learning, the class mean corresponds to one added class sample, for n-shot learning the mean is calculated from n samples of a new class. More formally, let $\boldsymbol{\mu}_c$ be the mean vector for the c-th class from the set $\mathcal{C}$ of available classes, defined as

$$\boldsymbol{\mu}_c = \frac{1}{N_c} \sum_{i=1}^{N_c} f(\mathbf{x}_i; \boldsymbol{\theta}), \tag{1}$$

where N_c is the number of samples $\mathbf{x}$ for class c, f is a feature extraction function and $\boldsymbol{\theta}$ are model-parameters. To predict the object class of a sample we seek the minimum distance to all class means by computing

$$\operatorname*{arg\,min}_{c=1,\dots,C} \left\| f(\mathbf{x}; \boldsymbol{\theta}) - \boldsymbol{\mu}_c \right\|_2, \tag{2}$$

where $C = |\mathcal{C}|$ is the total number of classes.

3.1. Feature Extraction

Motivated by their recent success in image recognition tasks, we utilize CNNs for feature extraction. Instead of training a deep network from scratch, we take a CNN model trained for the ImageNet Challenge [7] and fine-tune on our task-specific training data. This can be seen as domain transfer from one task to another and has proven to be superior to hand-crafted features [4]. As the later layers of the network correspond to high-level features, we use the last fully connected layer as 4096-dimensional feature representation and normalize each feature vector by dividing by its l_2-norm.

3.2. Embedding

After fine-tuning the CNN, we employ several distance metric learning methods to learn a discriminative linear embedding matrix $\mathbf{W} \in \mathbb{R}^{d \times 4096}$, with $d \in \{1024, 4096\}$. This embedding projects samples from the same object class next to each other in a high dimensional feature space, while simultanously pushing samples from different object classes far away from each other. Using the embedding $\mathbf{W}$, the class prediction from Equation (2) becomes

$$\operatorname*{arg\,min}_{c=1,\dots,C} \left\| \mathbf{W} \cdot f(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{W} \cdot \boldsymbol{\mu}_c \right\|_2, \tag{3}$$

In this work, we consider optimizing NCM loss functions and the LMNN loss with respect to $\mathbf{W}$ to learn our embedding. In the remainder of this section, we will formally explain the different methods.

NCM. As proposed in [10], embeddings for NCM classifiers are usually learned by minimizing the negative log-likelihood. The posterior probability $p(c|\mathbf{x})$ of class c given a sample $\mathbf{x}$ is defined as

$$p(c|\mathbf{x}) = \frac{e^{-\delta(\mathbf{x}, \boldsymbol{\mu}_c; \boldsymbol{\theta})^2}}{\sum_{i=1}^{C} e^{-\delta(\mathbf{x}, \boldsymbol{\mu}_i; \boldsymbol{\theta})^2}}, \tag{4}$$

where δ is defined as

$$\delta(\mathbf{x}, \boldsymbol{\mu}; \boldsymbol{\theta}) = \left\| \mathbf{W} \cdot f(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{W} \cdot \boldsymbol{\mu} \right\|_2. \tag{5}$$

To learn the embedding $\mathbf{W}$, minimize the negative log-likelihood

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} \ln p(y_i|\mathbf{x}_i) \tag{6}$$

of sample $\mathbf{x}_i$ and its corresponding class label y_i. In subsequent sections we refer to this method as probabilistic NCM (NCM_P), since we are optimizing a negative log-likelihood function.

LMNN. The loss function of the LMNN embedding [15] consists of two terms. One adds a penalty for samples that share the same class label but exceed a certain distance (margin), while the other penalizes samples with different class labels that are close in feature space. The loss is calculated on triplets instead of pairs, where a sample is complemented by a sample of the same and a sample of a different class. The set of triplets is given by

$$D = \{(i, j, k) : y_i = y_j, y_i \neq y_k\} \tag{7}$$

and $1 \leq i < j < k \leq N$, the LMNN loss function over the triplet set is then defined as

$$L(D) = \sum_{(i,j,k)\in D} d_{ij}^2 + l_{ijk}. \tag{8}$$

The distance function d of two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$d_{ij} = \|\mathbf{W}\cdot f(\mathbf{x}_i;\boldsymbol{\theta}) - \mathbf{W}\cdot f(\mathbf{x}_j;\boldsymbol{\theta})\|_2 \tag{9}$$

and the triplet loss function l_{ijk} is defined as

$$l_{ijk} = \max(0, 1 + d_{ij} - d_{ik}). \tag{10}$$

This embedding maximizes the distance in feature space between samples of different classes $(\mathbf{x}_i, \mathbf{x}_k)$, while concentrating samples that belong to the same class $(\mathbf{x}_i, \mathbf{x}_j)$. Following [13], during training we perform "hard" negative mining of triplets which violate the margin constraint imposed by l_{ijk}. Opposite to "soft" negatives, which do not violate the margin or violate the margin by only a small amount, "hard" negatives impose a high loss and therefore lead to faster training of the model and increased performance.

3.3. Large Margin Nearest Class Mean Classifiers

Inspired by LMNN we propose a large margin loss function for NCM classifiers. Ideally, samples from the same class are close to their own mean vector and are far away from other mean vectors in feature space. Let $\mathbf{x}$ be a data sample, y the class label and $\boldsymbol{\theta}$ the parameters of the feature extraction function f. We propose to train a NCM layer on top of the CNN features with the following NCM loss function

$$L(\mathbf{x}, y; \boldsymbol{\theta}) = \lambda\cdot\delta_y^2 + \sum_{c\in\mathcal{C}\setminus\{y\}} \max(0, 1+\delta_y-\delta_c)^2, \tag{11}$$

where $\delta_y = \delta(\mathbf{x}, y; \boldsymbol{\theta})$ and $\delta_c = \delta(\mathbf{x}, c; \boldsymbol{\theta})$ are distance functions as defined in Equation (5) and λ is a weighting parameter. The first part enforces the samples of one class to be embedded near the class mean of the data sample, while the second term penalizes if samples are within the margin of other class means. This large margin version of the NCM classifier will be referred to as NCM_{LM}

3.4. Robust NCM

Due to variations in shape, illumination and appearance, feature vectors from an object class usually exhibit intra-class variance. We model this uncertainty by assuming that a feature vector for a sample $\mathbf{x}$ associated with class c is generated by a normal distribution $\mathcal{N}(\boldsymbol{\mu}_c, \sigma_c)$. We incorporate this variation in our model by computing the standard deviation σ_c for all classes $c \in \mathcal{C}$ over the training set. During optimization we add random noise ϵ to our feature vectors to account for this uncertainty. More formally, the loss function we minimize is

$$L(\mathbf{x}, y; \boldsymbol{\theta}) = \lambda\cdot\hat{\delta}_y^2 + \sum_{c\in C\setminus\{y\}} \max(0, 1+\hat{\delta}_y-\hat{\delta}_c)^2, \tag{12}$$

where

$$\hat{\delta}(\mathbf{x}, \boldsymbol{\mu}; \boldsymbol{\theta}) = \left\|\mathbf{W}\cdot\hat{f}(\mathbf{x};\boldsymbol{\theta}) - \mathbf{W}\cdot\boldsymbol{\mu}\right\|_2. \tag{13}$$

and

$$\hat{f} = f(\mathbf{x}) + \boldsymbol{\Sigma}_y^{\frac{1}{2}}\epsilon\cdot\gamma. \tag{14}$$

$\boldsymbol{\Sigma}_y$ is the diagonal covariance matrix of class y, $\epsilon \in \mathbb{R}^{4096} \sim \mathcal{N}(0, 1)$ is a random vector drawn from a normal distribution and γ is a hyper-parameter, which defines the impact of the distortions. In our experiments we fix λ to 0.01 and γ is set to 0.5. During training we first compute the standard deviation of each feature per object class. We then add the noise to our feature vectors, to make the embedding $\mathbf{W}$ more robust against inter-class variations. This robustification is done in real time during training and can be seen as data augmentation, making the impact of outliers on the means smaller. We refer to this method as $NCM_{LM\text{-}R}$.

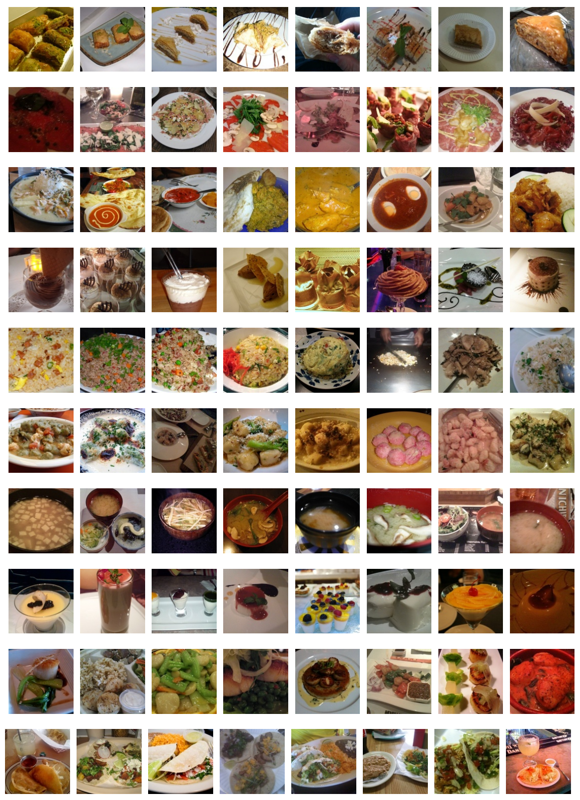


Figure 2. First 8 samples of randomly chosen classes from the Food-101 dataset [2]. From top to bottom: baklava, beef carpaccio, chicken curry, chocolate mousse, fried rice, gnocchi, miso soup, panna cotta, scallops, tacos.

4. Experiments

For evaluation of our method we use the publicly available Food-101 dataset [2]. It consists of 101 food classes with 1000 images per class. The images were taken in real world environments, exhibiting a lot of variation in illumination conditions or food arrangement (see Figure 2 for some examples) and are well suited for the targeted application case where users add data continuously.

Following the protocol in [2], we randomly split the 1000 samples of each class into 750 for training and 250 for testing. For training of the CNN, we then apply a 80%/20% split for training and validation (600 and 150 samples respectively). This results in a training-, validation- and test-set with 60.600 (60%), 15.150 (15%) and 25.250 (25%) samples, respectively. Further from the 101 classes we randomly select 50 training classes on which we train our classifiers and 51 classes on which we evaluate the generalization capability of our method to novel classes. For the sake of completeness, we also evaluate the embeddings on the 50 training classes only.

For fine-tuning CaffeNet on the Food-101 dataset, we train our network with Stochastic Gradient Descent (SGD) and momentum. We follow standard fine-tuning protocols [7] and use a low initial learning rate of 0.001 and a momentum of 0.9. We anneal the learning rate by a factor of 10 after each 20.000 iterations. To determine convergence, we measure the accuracy on a validation set after 500 gradient updates.

We optimize our embeddings with SGD and momentum. For training the embeddings, when not otherwise stated, we fix the weights of the CNN and train just the last embedding layer. This allows us to use large learning rates of 0.25-0.5 with a momentum term of 0.9. Further, we use large minibatch sizes of 1024 and train for about 20 epochs. We exponentially anneal the learning rate at epoch 15 and 18. To determine convergence, we measure the accuracy on our validation set after each training epoch.

In our experiments we use Caffe [5] for fine-tuning, while the evaluations on the embedding methods are implemented in Python utilizing the Theano library [1].

4.1. Experiments with Known Classes

To obtain feature representations and a softmax baseline, we fine-tune the pretrained ImageNet CaffeNet model from [7] on the 50 training classes from the Food-101 dataset as described above. In the following, we compare our methods to the softmax classifier ($CNN_{softmax}$) and to a probabilistic NCM (NCM_P) version related to the work of [10]. The first results are obtained by nearest class mean classification, using euclidean (CNN_{euc}) and cosine (CNN_{cos}) distance measures between the class means of all trainings samples and the test samples. Subsequently we train our NCM and LMNN embedding layers on top of the fine-tuned network while leaving the net weights fixed (NCM_{LM}, $NCM_{LM\text{-}R}$, $LMNN$). A summary of the results is depicted in Table 2.

Interestingly, the nearest class mean classification performs better than the probabilistic version of NCM, implying that the CNN features already separate the classes well. Our robust NCM version improves results over the probabilistic version by about 2% and is very competitive in comparison to the end-to-end trained softmax classifier of the network. The NCM embedding trained on the hinge loss and the LMNN embedding also reach comparable accuracy.

Method	Emb.	$n = 1$	$n = 5$	$n = 10$	$n = 20$	$n = 50$	$n = 100$
CNN_{euc}	–	44.15 ± 0.08	49.30 ± 0.31	$\mathbf{54.26 \pm 0.30}$	$\mathbf{57.66 \pm 0.20}$	$\mathbf{60.03 \pm 0.15}$	$\underline{60.83 \pm 0.13}$
CNN_{cos}	–	44.92 ± 0.24	49.82 ± 0.34	53.92 ± 0.32	$\underline{57.26 \pm 0.21}$	$\underline{59.86 \pm 0.16}$	60.76 ± 0.13
LDA	–	44.51 ± 0.01	44.92 ± 0.12	45.85 ± 0.12	48.44 ± 0.18	57.87 ± 0.17	$\mathbf{63.61 \pm 0.16}$
NCM_P	1024	45.55 ± 0.33	50.11 ± 0.32	51.89 ± 0.25	53.08 ± 0.20	54.03 ± 0.13	54.39 ± 0.09
NCM_P	4096	45.62 ± 0.34	50.23 ± 0.33	52.03 ± 0.25	53.23 ± 0.20	54.20 ± 0.14	54.57 ± 0.09
NCM_{LM}	1024	46.23 ± 0.32	51.46 ± 0.34	53.49 ± 0.26	54.88 ± 0.18	56.02 ± 0.15	56.44 ± 0.11
NCM_{LM}	4096	45.97 ± 0.28	51.43 ± 0.34	53.51 ± 0.27	54.93 ± 0.19	56.06 ± 0.13	56.50 ± 0.11
$NCM_{LM\text{-}R}$	1024	$\mathbf{46.30 \pm 0.32}$	$\mathbf{51.95 \pm 0.35}$	$\underline{54.15 \pm 0.26}$	55.67 ± 0.21	56.89 ± 0.15	57.37 ± 0.11
$NCM_{LM\text{-}R}$	4096	$\underline{46.28 \pm 0.32}$	$\underline{51.94 \pm 0.35}$	54.13 ± 0.28	55.66 ± 0.21	56.85 ± 0.15	57.31 ± 0.11
$LMNN$	1024	45.78 ± 0.25	51.60 ± 0.32	53.82 ± 0.28	55.34 ± 0.21	56.57 ± 0.16	57.05 ± 0.12
$LMNN$	4096	45.29 ± 0.18	51.58 ± 0.33	54.10 ± 0.28	55.81 ± 0.21	57.14 ± 0.14	57.63 ± 0.11
SVM	–	46.52 ± 0.40	50.02 ± 0.35	52.25 ± 0.30	55.08 ± 0.24	59.74 ± 0.19	63.38 ± 0.17

Table 3. Classification accuracy over the full Food-101 test-set (250 samples per class) after adding $n \in \{1, 5, 10, 20, 50, 100\}$ training samples for each of the 51 test-classes. Accuracy and standard deviation are calculated over 100 runs. The baseline accuracy for end-to-end training of the CNN on all classes with all available data is 66.63%. The best and the second best result in each column is shown in bold and underlined.

Method	Emb. size	Accuracy
CNN_{euc}	–	68.60
CNN_{cos}	–	68.64
NCM_P[10]	1024	67.66
NCM_P[10]	4096	67.75
NCM_{LM}	1024	69.00
NCM_{LM}	4096	69.14
$NCM_{LM\text{-}R}$	1024	**69.68**
$NCM_{LM\text{-}R}$	4096	<u>69.61</u>
$LMNN$	1024	69.20
$LMNN$	4096	69.11
$CNN_{softmax}$	–	70.26

Table 2. Classification results of the 50 classes used for fine-tuning the CNN model for feature extraction. Our proposed robust NCM version reaches almost the same accuracy as the end-to-end trained softmax classifier while improving the results over the standard probabilistic NCM classifier by 2%. Best (bold) and second best (underlined) embeddings are marked.

4.2. Introducing Unseen Classes

To assess how our method generalizes to new classes from only a limited number of samples, we use n random samples from the training set of the remaining 51 classes to compute the mean vectors from the output of the embeddings. The embeddings and the CNN remain fixed and are not retrained, hence the addition of new classes reduces to storing the new class means. We choose $n \in \{1, 5, 10, 20, 50, 100\}$ and report the accuracy on the full Food-101 test-set, where every class is represented by 250 samples. Since for small values of n the results might have a large standard deviation, we repeat these experiments 100 times using different training samples to compute the class means that represent the new classes. Table 3 shows, that fine-tuning the network in the training phase (known classes) with metric learning methods generally improves accuracy in the testing phase for smaller values of n. Training the CNN with Caffe on the full dataset of 101 classes converges after approximately 100.000 iterations to 66.63%.

We also trained two more standard classifiers on the CNN features, namely SVM and LDA. It is remarkable, that although the SVM has access to the full dataset, the performance compared to our proposed methods is inferior for $n \in \{5, 10, 20\}$. The same applies for utilizing a LDA classifier, where only a big number of new samples achieves a performance improvement compared to our proposed methods.

5. Conclusion

We introduced embedding methods for one-shot and n-shot object class recognition. Our proposed extensions to NCM classifiers consistently improve the accuracy over the standard NCM training formulation in a scenario where the amount of classes to be recognized by the classifier doubles. Our methods perform best for settings where only very few new samples ($n \leq 10$) per class are available. The extension of the classifier to new object classes is independent of the old training data and is efficient in terms of computational expense and memory. This is especially useful for recognition systems running on embedded devices, where CPU power and memory is limited.

Acknowledgements

This work was supported by the Austrian Research Promotion Agency (FFG) under the projects MANGO (836488) and DIANGO (840824).

References

[1] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a CPU and GPU Math Expression Compiler. In *Proceedings of the Scientific Computing with Python Conference*, June 2010.

[2] L. Bossard, M. Guillaumin, and L. Van Gool. Food-101 – Mining Discriminative Components with Random Forests. In *European Conference on Computer Vision*, 2014.

[3] L. Fei-Fei, R. Fergus, and P. Perona. One-Shot Learning of Object Categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, 2006.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[5] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[6] M. Köstinger, P. Wohlhart, P. M. Roth, and H. Bischof. Joint Learning of Discriminative Prototypes and Large Margin Nearest Neighbor Classifiers. In *IEEE International Conference on Computer Vision*, 2013.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, 2012.

[8] B. M. Lake, R. R. Salakhutdinov, and J. Tenenbaum. One-Shot Learning by Inverting a Compositional Causal Process. In *Advances in Neural Information Processing Systems*, 2013.

[9] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to Detect Unseen Object Classes by Between-class Attribute Transfer. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[10] T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka. Distance-Based Image Classification: Generalizing to New Classes at Near-Zero Cost. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2624–2637, 2013.

[11] D. Parikh and K. Grauman. Relative Attributes. In *IEEE International Conference on Computer Vision*, 2011.

[12] R. Salakhutdinov, J. B. Tenenbaum, and A. Torralba. One-Shot Learning with a Hierarchical Nonparametric Bayesian Model. In *Workshop on Unsupervised and Transfer Learning in conjunction with the International Conference on Machine Learning*, 2012.

[13] F. Schroff, D. Kalenichenko, and J. Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[14] A. R. Webb and K. D. Copsey. *Statistical Pattern Recognition*. Wiley, 3rd edition, 2011.

[15] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance Metric Learning for Large Margin Nearest Neighbor Classification. In *Advances in Neural Information Processing Systems*, 2005.

[16] P. Wohlhart, M. Köstinger, M. Donoser, P. M. Roth, and H. Bischof. Optimizing 1-Nearest Prototype Classifiers. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

21st *Computer Vision Winter Workshop*
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

Cuneiform Detection in Vectorized Raster Images

Judith Massa[1], Bartosz Bogacz[1], Susanne Krömker[2] and Hubert Mara[1]
Interdisciplinary Center for Scientific Computing (IWR)
[1]Forensic Computational Geometry Laboratory (FCGL)
[2]Visualization and Numerical Geometry (NGG)
Heidelberg University, Germany
{judith.massa|bartosz.bogacz|susanne.kroemker|hubert.mara}@iwr.uni-heidelberg.de

Abstract. *Documents written in cuneiform script are one of the largest sources about ancient history. The script is written by imprinting wedges (Latin: cunei) into clay tablets and was used for almost four millennia. This three-dimensional script is typically transcribed by hand with ink on paper. These transcriptions are available in large quantities as raster graphics by online sources like the Cuneiform Database Library Initative (CDLI). Within this article we present an approach to extract Scalable Vector Graphics (SVG) in 2D from raster images as we previously did from 3D models. This enlarges our basis of data sets for tasks like word-spotting. In the first step of vectorizing the raster images we extract smooth outlines and a minimal graph representation of sets of wedges, i.e., main components of cuneiform characters. Then we discretize these outlines followed by a Delaunay triangulation to extract skeletons of sets of connected wedges. To separate the sets into single wedges we experimented with different conflict resolution strategies and candidate pruning. A thorough evaluation of our methods and its parameters on real word data shows that the wedges are extracted with a true positive rate of 0.98. At the same time the false positive rate is 0.2, which requires future extension by using statistics about geometric configurations of wedge sets.*

1. Introduction

Documents were written in cuneiform script for more than three millenia in the ancient Middle East [26]. Cuneiform characters were typically written on clay tablets by imprinting a rectangular stylus and leaving a wedge (*cuneus* in Latin) shaped trace, i.e., triangular markings. As clay was always cheaply and easily available, everybody capable of writing could produce robust documents. Therefore, the content of cuneiform tablets ranges from simple shopping lists to treaties between empires. The number of known tablets is assumed to be in the hundreds of thousands, which is constantly increasing as new tablets are excavated by archaeologists on a regular basis. By roughly estimating the number of words on those tablets, we can assume that the total amount of text in cuneiform script is comparable to those in Latin or Ancient Greek.

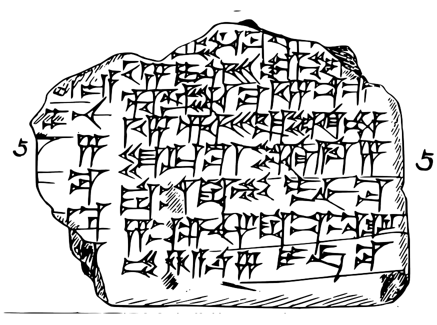
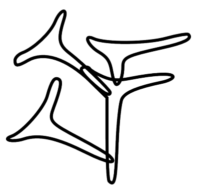
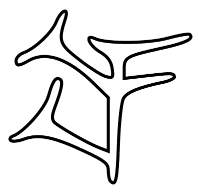


Figure 1: A tracing of the tablet VAT6546 [23]

Since 1999, a number of projects have been launched to facilitate the work of Assyriologists. The *Digital Hammurabi Project* is concerned with the digitization of cuneiform tablets [27]. Achievements of the project include the creation of high-resolution 3D models [17] as well as 3D and 2D visualization techniques for the models. Similarly, projects in Leuven deal with the efficient production of 3D models of tablets [28] and techniques to visualize the models [13]. The *Cuneiform Digital Library Initiative* [15] incorporates a number of projects aimed at cataloging cuneiform documents and making them available online as transliteration, tracing and 2D image. In [11], the software framework *CuneiformAnalyzer* is introduced. It assists the researchers in script



(a) (b)

Figure 2: (a) Paths described by four distinct wedge shapes and (b) the path described by the compound shape formed by them.

analysis by detecting and segmenting wedge impressions of 3D models [10]. Furthermore, the program simplifies collation of fragments and reconstruction of tablets with methods from 3D computer graphics. The *GigaMesh* project contributes with visualization methods and extraction of cuneiform characters from tablets [21, 20].

Our method extracts wedge-shaped impressions from raster images. These images are hand-drawn transcriptions of cuneiform tablets of varying quality and two different styles of marking wedges. We vectorize images of the transcriptions and match patterns and shapes to detect these constellations of wedges in the vectorized transcriptions.

2. Related Work

In [7] the problem of content-based image retrieval of Scalable Vector Graphics (SVG) documents is tackled. Their approach uses a description language to simplify comparisons between shapes. It represents an object by a basic shape, like a unit circle, and a transformation entailing its scale and translation from the origin. The resulting framework handles composites of simple SVG shapes, but no SVG path elements, which are able to represent arbitrary shapes. The similarity measure chosen is a weighted sum of shape, color, transformation, spatial and position similarity.

In [18] the problem of hierarchically clustering shapes described as vector graphics is addressed. Based on [29], Kuntz uses Fourier descriptors [6] to describe and compare single basic SVG shapes. The descriptors serve as feature vectors which then are clustered using state-of-the-art clustering algorithms.

We cannot apply Kuntz' method since it does not deal with shapes that are part of a compound described as one object. Yet, our input consists of SVG *path* elements that usually describe such compound shapes (Figure 2).

3. Implementation

Our implementation proceeds in five distinct steps. The first three steps transform raster image transcriptions into a set of skeletonized wedge constellations. The final two steps extract and prune wedge candidates from these constellations.

3.1. Vectorization

For the vectorization step, Selinger's *potrace* algorithm [1] is used. A directed graph G_1 is constructed by traveling along the edges between black and white pixels. Thereby, each vertex v in the graph corresponds to a pixel corner, which is adjacent to four pixels in the bitmap image of which at least one has to be black and one has to be white. An edge (v_i, v_{i+1}) between two vertices is created if the corresponding corners are neighbors in the bitmap image and the edge separates a black and a white pixel. Then, a path $p = \{v_0, \dots, v_n\}$ is a sequence of vertices, where there is an edge between each pair of consecutive vertices v_i and v_{i+1} for $i = 0, \dots, n-1$. A path is called *closed* if $v_0 = v_n$. Whenever a closed path is found, the color of the pixels enclosed by it is inverted. The algorithm is applied recursively to the new image until there are no black pixels left.

For each of the resulting paths a polygon is calculated. Therefore another directed graph G_2 is constructed, where each edge represents a straight path and the set of vertices of Graph G_2 is a subset of the vertices of Graph G_1 reduced to the endpoints of the straight paths. A path $p = \{v_0, \dots, v_n\}$ is called straight if for all index triples (i, j, k) with $0 \leq i < j < k \leq n$ there exists a point w on the straight line through v_i and v_k such that $d(v_j, w) \leq 1$. The function d denotes the Euclidean norm. Furthermore, not all four possible vertex-to-vertex directions $v_{(i+1)} - v_i$ may occur in the path (Figure 3).

Each edge then is assigned a penalty $P_{i,j}$ for using the corresponding straight path for the resulting polygon. The penalty is the product of the Euclidean length and the standard deviation of vertex distances.

$$D_{i,j} = \sum_{k=i}^{j} dist(v_k, \overline{v_i v_j})^2 \tag{1}$$

$$P_{i,j} = |v_i - v_j| \cdot \sqrt{\frac{1}{j \ominus i + 1} \cdot D_{i,j}} \tag{2}$$

[1]http://potrace.sourceforge.net. Project page of *potrace*. Last visited on 4/11/15.

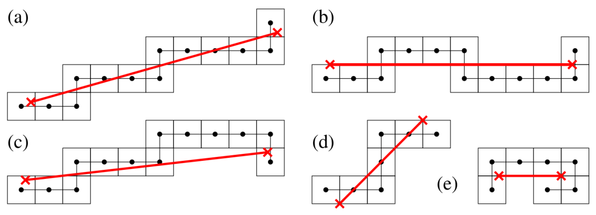


Figure 3: Shows how *potrace* checks if paths are straight. The dots represent the vertices of the paths and the squares the 1/2-neighborhoods of the vertices. Paths in (a), (b) and (d) are straight and (c) and (e) are not.

with $j \ominus i = j - i$ if $i \leq j$ and $j \ominus i = j - i + n$ if $j \leq i$ and $dist(a, \overrightarrow{cd})$ the Euclidean distance of a point to a straight segment. Finding an optimal polygon then is the equivalent to finding an optimal cycle in graph G_2, with the quality measured by the tuple (k, P), where k is the number of straight paths that make up the cycle and P is the sum of respective penalties. With that, a polygon with a smaller penalty but more segments is considered worse than a polygon with less segments but higher penalty.

After choosing a polygon, Bézier curves are calculated and by doing this, a smoothing of the corners is achieved where it seems reasonable. Optionally, consecutive curves are joined if the segments agree in convexity and the total direction change does not exceed 89 degrees.

3.2. Discretization

We tested minimizing the maximum distance between the polygon line segments and the contour segments, but found that even if a polygon approximates an arbitrary shape well, it is still not assured that the resulting Voronoi skeleton will be a good approximation to the skeleton of the wedge constellation. The key for a good discrete skeleton turned out to be the limitation of the distance of two sample points along the shape outline. However, calculating the distance along a Bézier curve $B(t)$ is a complex task [12] since the length s of the complete curve is

$$s = \int_0^1 \sqrt{B'_x(t)^2 + B'_y(t)^2} dt, \tag{3}$$

which has no closed-form solution. Yet, we know that the curve length s of a Bézier of degree 3 has the sum of the distances of consecutive control points C_i as upper bound:

$$s \leq \sum_{i=0}^{2} \|C_{i+1} - C_i\|. \tag{4}$$

Since smaller distances along the path can only improve the quality of the resulting skeleton, this upper bound is used for discretizing the silhouette.

3.3. Skeletonization

In order to deal with occluding wedge marks in the detection step, shape skeletons are used as intermediate representations. Different definitions for shape skeletons have been stated since [2, 22, 25, 19]. Based on [25], we define a shape skeleton as the infinite set of points within the shape boundaries that have more than one closest point on the shape outline. The skeleton can be computed efficiently with time complexity $O(n \log n)$ by using Voronoi diagrams [16]. The Voronoi diagram for a set of sites S divides a space into $|S|$ partitions called the Voronoi regions. In $\mathbb{R}^2$ a Voronoi region is the interior of a convex polygon, whose boundaries, the Voronoi edges, are equidistant to two of the input sites. As input sites, the polygon vertices obtained in the previous discretization step are used.

The Voronoi diagram (Figure 4b) is computed by solving the dual problem first: the Delaunay triangulation (Figure 4a). Each Voronoi vertex represents the circumcenter of a Delaunay facet and a Voronoi ridge connects two such points of neighboring facets. An implementation of the *quickhull* algorithm [1] is used to calculate the 2-dimensional Delaunay triangulation from a 3-dimensional convex hull.

The Voronoi ridges with end points outside the original shape boundaries and ridges crossing the contour are removed. The skeleton, represented as an undirected graph (Figure 4c), consists of more and in general shorter segments. These, in turn, are made up of longer segments the more vertices form the approximated polygon. Since the short segments are rarely meaningful, considering their directions, a new skeleton (Figure 4d) is constructed. The computation is done by a graph traversal algorithm. It follows a series of consecutive edges until an end node is incident to more than two edges in the original skeleton.

3.4. Extraction

The basic shape of a wedge impression can be described by a Y- or T-junction. We call this junction

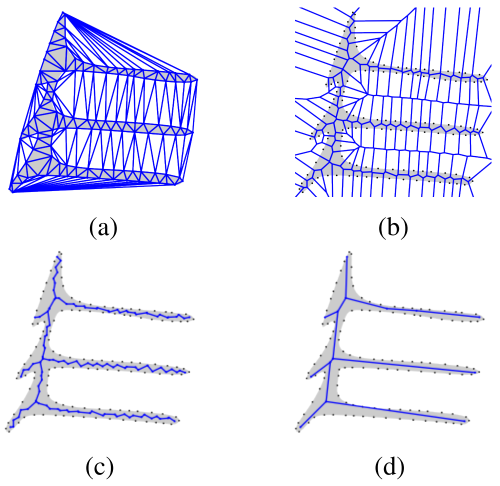


Figure 4: Visualization of important steps of the skeleton computation and simplification process: (a) the Delaunay triangulation, (b) the Voronoi diagram, (c) the inner elements of the Voronoi diagram and (d) the resulting skeleton.

the wedge-head and the ridges extending from the junction the wedge-arms.

After having computed a shape skeleton, the detection of the wedge-heads of the impressions can be approached. There are two different ways a wedge impression can be drawn: as contour lines or shapes filled with ink. For a single wedge impression, the filled shape results in a single closed curve after bitmap tracing and the shape contour is represented as two closed curves, where only the area between both curves is filled with color. Usually, the representation as unfilled shape contour is intended, but for small wedges, the thickness of the pencil used for the original ink tracing sometimes leads to solid shapes. The two representations result in two different skeletons as shown in Figure 5. These two cases are considered separately and certainty values w_{loc} are calculated for locations in the skeleton graph that seem likely to contain a wedge-head. In both cases, w_{loc} ranges from zero to one with values close to one indicating a high probability of a wedge-head at the considered location. The result of this step is a set of wedge-heads for which the certainty value exceeds the threshold $t_{loc}^{contour}$ for contour wedges or t_{loc}^{solid} for solid wedges.

Wedge-Head Detection of Shape Contours Having a wedge impression represented as a contour, the respective skeleton graph shows a cycle resembling a triangle at the position of the wedge-head (Figure 5a). This fact is used to locate the wedge-head of a wedge contour. A cycle in an undirected graph is an ordered set of vertices

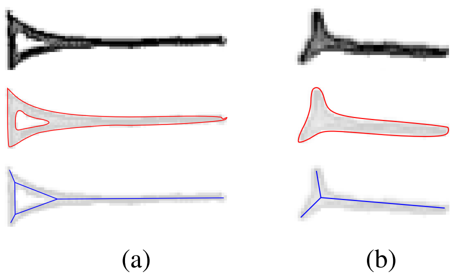


Figure 5: Two different ways of representing a wedge impression in ink tracings: a) as unfilled contour and b) as filled shape. The result of the bitmap trace is drawn in red, the simplified Voronoi skeleton in blue.

$$\mathcal{C} = (v_0, v_1, \ldots, v_n) \tag{5}$$

where circular consecutive vertices are adjacent and no vertex appears twice.

To avoid outliers to be taken into consideration as wedge-heads, we only look for short cycles. Two concepts of length are possible: the number of edges forming the cycle

$$l_{edge}(\mathcal{C}) = |\mathcal{C}| \leq t_{edge} \tag{6}$$

and the accumulated distances l_{dist} along the cycle path, using P_v as the coordinate of a node v, that can be formally defined by

$$l_{dist}(\mathcal{C}) = \sum_{i=0}^{|\mathcal{C}|-1} |P_{v_i} - P_{v_j}| \leq t_{dist} \tag{7}$$

with $v_i, v_j \in \mathcal{C}$ and $j = (i+1) \bmod |\mathcal{C}|$ thresholds t_{edge} and t_{dist}.

Using l_{edge} as the cycle length, this results in a time complexity of $O(|E| \cdot t_{edge})$, using l_{dist}, the complexity will be $O(|E| \cdot \left\lfloor \frac{t_{dist}}{\min\{\|P_u - P_v\| : (u,v) \in \mathcal{E}\}} \right\rfloor$ in the worst case. These concepts are used next to each other in the algorithm.

The cycle extraction proceeds as follows: A depth-first search tree is built and the back-edges are extracted. For each back-edge (u, v) a depth-limited search for node v is conducted with u as root node; paths from v to u represent cycles when joined with $\{(u, v)\}$, except the direct path $\{(v, u)\}$. At last, the set of cycles is reduced to contain only unique cycles.

A set of unique cycles contains no two cycles that are equivalent, i.e., if they are induced by the same set of graph edges. This is tested by:

$$\mathcal{C}_1 \equiv \mathcal{C}_2 \quad \Longleftrightarrow \quad \mathcal{E}_{\mathcal{C}_1} \setminus \mathcal{E}_{\mathcal{C}_2} = \emptyset, \tag{8}$$

where $\mathcal{E}_{\mathcal{C}}$ denotes the edge set of a cycle $\mathcal{C}$.

The depth-limited search stops following a search branch when either l_{edge} or l_{dist} exceed their respective thresholds, t_{edge} or t_{dist}, or when a target node is discovered. It returns a list of paths from the root to the target node.

Triangle Similarity Once all unique cycles of a skeleton graph have been extracted, their resemblance to a triangle can be analyzed. This can be achieved by comparing the triangle with the smallest error that can be created with the cycle's vertices with the original cycle (Figure 6). With

$$A_{err}^{\mathcal{C},(i_0,i_1,i_2)} = \sum_{j=0}^{2} A_{(c_{i_j}, c_{i_j+1 \bmod |\mathcal{C}|}, \ldots, c_{i_{j+1}})} \tag{9}$$

being the sum of error areas between triangle and polygon, we have

$$w_{loc}^{contour}(\mathcal{C}) = 1 - \min_{i_0,i_1,i_2;} \left\{ \frac{A_{err}^{\mathcal{C},(i_0,i_1,i_2)}}{A_{\mathcal{C}} + A_{err}^{\mathcal{C},(i_0,i_1,i_2)}} \right\} \tag{10}$$

for $0 \leq i_0 < i_1 < i_2 \leq |\mathcal{C}| - 1$, $\mathcal{C} = (c_0, \ldots, c_n)$ and $n \geq 2$.

The advantage of this similarity measure is that the three vertices of the triangle are also vertices of the skeleton graph, thus providing us with feasible starting points for the wedge-arm tracing. Since the cycles form simple polygons, the enclosed areas can be calculated with the shoelace or surveyor's area formula [4].

Wedge-Head Detection of Solid Shapes Solid imprints have their centers at junctions v of a skeleton $\mathcal{S}$ with a particularly long distance from the shape contour. This distance is approximated by the distance of the coordinate P_v of the vertex v to all sites s with $s \in S(v)$, where S is the site set of the underlying Voronoi diagram and

$$S(v) = \{s \in S | P_v \text{ is vertex of } V(s)\} \tag{11}$$

is the set of sites with P_v being a vertex of their Voronoi region $V(s)$. The equation

$$d(P_v, s_1) = d(P_v, s_2) \qquad \forall s_1, s_2 \in S(v) \tag{12}$$

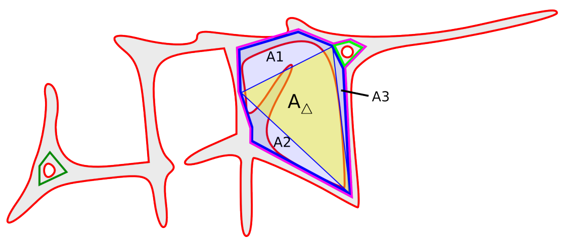


Figure 6: The triangle with area A$_\triangle$ shows the best triangle for the blue polygon. Since the sum of the error areas A$_{err}$ = A1+A2+A3 is almost equal to the triangle area A$_\triangle$, the polygon with area A$_\triangle$+A$_{err}$is not one of the polygons chosen for wedge-head positions.

always holds by definition of the Voronoi diagram. Therefore, any random $s \in S(v)$ can be chosen to get a measure for the distance to the contour and use it as hint for plausible locations of heads of solid wedges (Figure 7). Percentiles of the site-to-vertex distances

$$\mathcal{D} = \{d(P_v, s) | v \in \mathcal{S} \land s \in S(v)\} \tag{13}$$

are used instead of the minimum and maximum to account for outliers. In order to arrive at a range from zero to one with values close to one for long distances $d(p_v, s)$ and close to zero for short distances, the first percentile is used as minimum d_{lower}, the 99th percentile of these distances as maximum d_{upper} and the position within this range is used as certainty value.

$$w_{loc}^{solid}(v) = \begin{cases} 0 & \text{if } d(P_v, s) \leq d_{lower} \\ 1 & \text{if } d(P_v, s) \geq d_{upper} \\ d^*(P_v, s) & \text{else} \end{cases} \tag{14}$$

$$d^*(P_v, s) = \frac{d(P_v, s) - d_{lower}}{d_{upper} - d_{lower}} \tag{15}$$

For a junction, where n edges meet, $\binom{n}{3}$ wedge-heads are retrieved.

After locating the position of a wedge-head, the extents of the impression are calculated. For a wedge-head, multiple wedges are proposed. Vertices $\{v_i\}_{i=1,2,3}$ must fulfill two conditions:

1. The line segment between the coordinates of a wedge vertex v_i and tracing start point S may not intersect with the shape boundary.

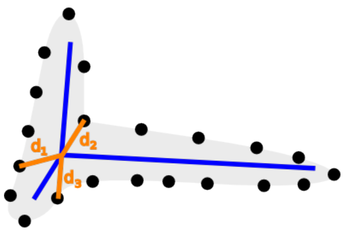


Figure 7: Possible locations of solid wedges are found by looking at skeleton junctions: if their distance to the closest discretization points is above a threshold, it is likely that the center of a wedge impression is located here. Due to the definition of the Voronoi skeleton, the equation $d_1 = d_2 = d_3$ holds at every skeleton junction.

2. A vertex of a wedge must be located within the infinite area between the lines through the coordinates of S and the wedge-head vertices v_j^H and v_k^H as shown in Figure 8. The condition can be checked by testing if

$$\measuredangle(\overrightarrow{P_S P_{v_i}}, \vec{v}) \leq \frac{\alpha}{2} \qquad (16)$$

with

$$\alpha = \measuredangle(\overrightarrow{P_{v_j^H} P_S}, \overrightarrow{P_{v_k^H} P_S}) \qquad (17)$$

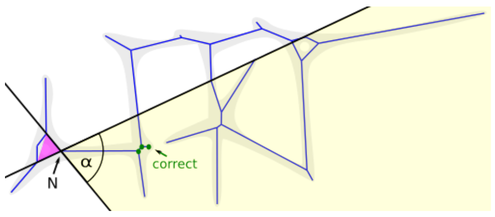
and $\vec{v}$ being the angle bisector of $\overrightarrow{P_{v_j^H} P_S}$ and $\overrightarrow{P_{v_k^H} P_S}$.

For contour wedges, the tracing start node for a wedge vertex v_i is the respective wedge-head vertex v_i^H and for a solid wedge, the start node is the wedge center. The reason why for the contour vertex, the line checked for the conditions above starts at the head vertex instead of the wedge center is that the center in this case is not a part of the shape skeleton and is typically located inside a hole in the shape.

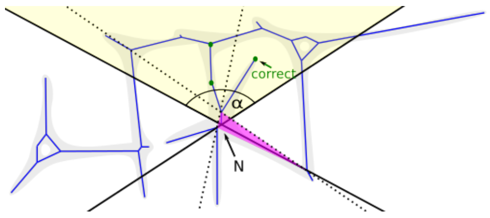
From the respective start node the algorithm follows all paths within the area of valid nodes shown in Figure 8. A path may have sections of a certain number of nodes that are inadmissible as wedge vertices. The algorithm returns multiple arms for one direction resulting in multiple wedge suggestions for a wedge-head. If $n_{arm\,1}$, $n_{arm\,2}$ and $n_{arm\,3}$ are the number of arms returned for the respective wedge-head vertex, the number of wedges is $n_{arm\,1} \cdot n_{arm\,2} \cdot n_{arm\,3}$.

3.5. Wedge Set Reduction

The certainty measures for wedge-head locations w_{loc} can only be used as first hints to possible locations. After wedge-arm tracing, we still get wedge



(a)



(b)

Figure 8: The pink area shows the place where wedge vertices may be located given the wedge-head of (a) a contour wedge and (b) a solid wedge. The marked junctions are valid as wedge vertex since the straight line to N does not cross the shape boundary. The dotted line in (b) shows that if we had chosen N for solid wedges as for contour wedges, we would not be able to find the correct wedge vertex.

candidates that can be easily identified as improbable by computing the angles between the arms. We want the arms to be evenly spread, so we punish angles that deviate from 120 degrees. Having α_1, α_2 and α_3 as internal wedge angles, we use

$$w_{angle}(\alpha_1, \alpha_2, \alpha_3) = p(\alpha_1) \cdot p(\alpha_2) \cdot p(\alpha_3) \qquad (18)$$

with

$$p(\alpha) = \frac{120}{120 + |120 - \alpha|} \qquad (19)$$

as measure for the angle quality of wedges. This measure is used in a preliminary reduction step to eliminate all wedges whose angle quality exceed a given threshold.

The simplest strategy using a *threshold* takes the remaining wedges and removes those that share heads with other wedges having higher angle quality. All of the other strategies proceed by iteratively testing wedges against the set of chosen wedges and adding them to the result set if no conflicts arise. Wedges are not added if the number of wedge-head edges, that are not used by any other wedge as head or arm edge, goes below a threshold. Furthermore, contour wedges may not share a head edge with another wedge.

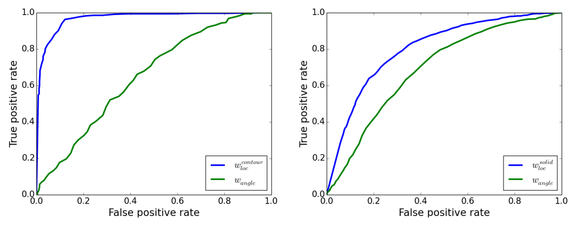
Balanced Strategies For documents where the number of solid wedges is greater or equal to the number of contour wedges, we implemented balanced strategies. There are six different strategies of this kind: *Balanced-loc*, *balanced-angle* and *balanced-size* sort the wedges by w_{loc}, w_{angle} and size respectively. *Balanced-sides-loc*, *balanced-sides-angle* and *balanced-sides-size* sort the wedges first by the number of arms that contain at least one edge that is not already used by a chosen wedge. The second kind of balanced strategies recalculates the number of free arms after each iteration. As measure for the size of a wedge, the average length of the lines from the center to the three edges is taken.

Contour-Fill Strategies Most documents contain more contour wedges than solid wedges. For these documents, the contour-fill strategies have been implemented. The strategies are *contour-fill-loc*, *contour-fill-angle* and *contour-fill-size*, *contour-fill-sides-loc*, *contour-fill-sides-angle* and *contour-fill-sides-size*. They proceed as their respective balanced counterpart but consider the set of contour wedges first before adding solid wedges to the set of chosen wedges. The candidate set of solid wedges only calculated after the set of chosen contour wedges is computed and vertices that are incident to a wedge-head edge are excluded.

4. Results

The algorithm has been tested on 94 tracings from [23] and [14]. The groundtruth is determined by manually deciding for each cycle and skeleton junction if they are valid positions of wedge-heads. Since a tracing rarely contains less than 500 wedge-marks, two typical tracings have been chosen for the evaluation. They differ in the representation of fractures, in size and in the percentage of solid wedges. As result we have 1252 annotated cycles and 3792 annotated junctions serving as groundtruth.

For the evaluation of the discrimination capabilities of w_{loc} and w_{angle}, the Receiver Operating Characteristic (ROC) [9] will be used. The ROC shows the quality of a detector by assigning it a point in the ROC space, with the false positive rate (FPR) as x-coordinate and the true positive rate (TPR) as y-coordinate. Therefore, the point assigned to an optimal detector has the coordinates (0,1). As measure for the overall performance of a discriminator function, the F-score is given [24].



(a) (b)

Figure 9: ROC curves of the measures used for the detector of (a) contour wedges and (b) solid wedges

Contour Wedges Candidates for contour wedges are found by searching for cycles in the skeleton graph. The similarity of the cycle to a triangle is then used as quality measure for the detector for early rejection of improbable locations for wedge heads. Figure 9a shows that early rejection is reasonable, since the chosen function for the location proves to be a good estimator. However, the green curve shows that the chosen measure for the angles between the arms of a reconstructed wedge is less optimal as discriminator.

Figure 10a shows the F-score for the contour wedge detector using thresholding only. It shows high scores of about 0.8 to 0.9 for thresholds for the location quality of about 0.7 to 0.9. The threshold for the angle quality should not be chosen too high, but 0.7 at most. The maximum score of 0.90 is achieved for $t_{loc}^{contour} = 0.79$ and the threshold $t_{angle} = 0.45$.

Solid Wedges Candidates for solid wedges are found by searching for skeleton junctions with great distance to the shape contour. Figure 9b shows that weight is a worse discriminator for solid wedges than for contour wedges. The discrimination quality for the angle quality measure for solid wedges looks very similar to the respective curve for contour wedges.

Figure 10b shows the F-scores for the solid wedge detector. For solid wedges, this detector achieves a score of about 0.62 at maximum. In contrast to the F-scores for the contour wedge detector this score is quite low. The reason for this can be found in the fact, that there are a lot more locations to check since every skeleton junction is considered. Especially when junctions are located next to each other, false hits occur frequently. The best score is achieved for $t_{loc}^{solid} = 0.68$ and $t_{angle} = 0.56$.

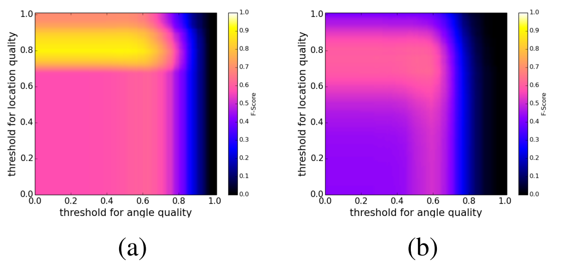


Figure 10: F-scores for detector of (a) contour wedges and (b) solid wedges.

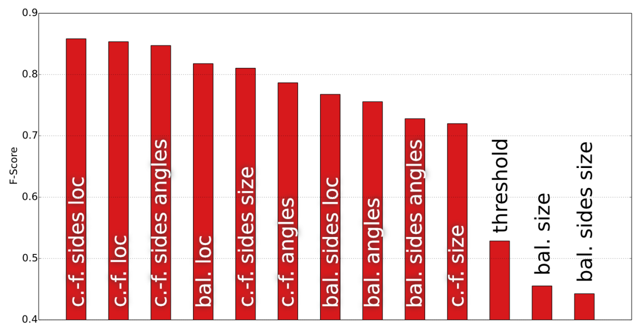


Figure 11: F-scores of reduction strategies for test case VAT6546. The strategies are sorted in descending order by their performance.

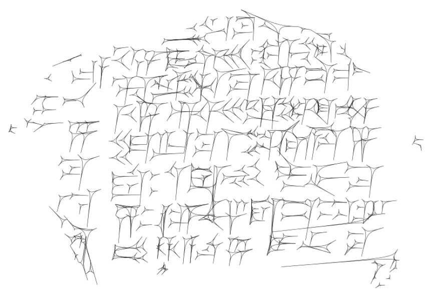


Figure 12: The extracted wedge marks of test case VAT6546.

4.1. Wedge Set Reduction

The reduction strategies serve to overcome the shortcomings of pure thresholding. We demonstrate their differences with a tracing of the tablet VAT6546 [23]. It represents fractures with lines and shows 215 contour and 120 solid wedges. Figure 11 shows the F-scores for this case. The best F-score is achieved by the *contour-fill-sides-loc* method with 86% (Figures 12 and 11).

Figure 13 compares the strategies concerning TPR and FPR. It shows a clear ordering between similar methods that differ merely in the measure for the sorting of the wedges.

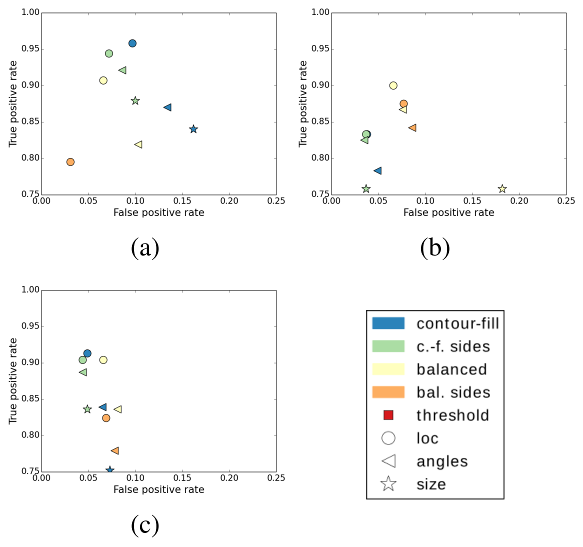


Figure 13: Receiver Operating Characteristic (ROC) space showing performance of wedge set reduction strategies for test case VAT6546 (a) for contour wedges and (b) for solid wedges and (c) for all wedges.

5. Summary and Outlook

In this work we presented an algorithm that uses bitmap tracing and skeletonization as intermediate steps to detect wedge impressions in raster graphics of cuneiform documents. We have shown the weaknesses of the measures used to construct an initial wedge set and have shown how we can use conflict set reduction strategies to improve the results significantly.

This work is part of ongoing research on optical character recognition for cuneiform characters [3] and used as one of many sources of wedge constellations. The presented method will allow us to perform word spotting on raster image databases as the CDLI. We will also examine if statistic approaches as in [5] or [8] can be used to enhance the detection results.

References

[1] C. Barber, D. Dobkin, and H. Huhdanpaa. The Quickhull Algorithm for Convex Hulls. *ACM Transaction on Mathematical Software (TOMS)*, 22(4):469–483, 1996. 3

[2] H. Blum. A Transformation of Extracting New Descriptors of Shape. In *Models for the Perception of Speech and Visual Form*, pages 362–380. MIT Press, 1967. 3

[3] B. Bogacz, J. Massa, and H. Mara. Homogenization of 2D & 3D Document Formats for Cuneiform

Script Analysis. In *Proc. of the 3rd International Workshop on Historical Document Imaging and Processing (HIP15)*, 2015. 8

[4] B. Braden. The Surveyor's Area Formula. *The College Mathematics Journal*, 17(4):326–337, 1986. 5

[5] M. Cammarosano, G. Müller, D. Fisseler, and F. Weichert. Schriftmetrologie des Keils: Dreidimensionale Analyse von Keileindrücken und Handschriften. *Die Welt des Orients*, 44(1):2–36, 2014. 8

[6] R. Cosgriff. Identification of Shape. ASTIA AD 254 792 820-11, Ohio State University Research Foundation, 1960. 2

[7] E. Di Sciascio, F. Donini, and M. Mongiello. A Knowledge Based System for Content-based Retrieval of Scalable Vector Graphics Documents. In *Proceedings of the 2004 ACM Symposium on Applied Computing*, pages 1040–1044, 2004. 2

[8] D. Edzard. Keilschrift. In *Ĭa... - Kizzuwata*, volume 5 of *Reallexikon der Assyriologie und vorderasiatischen Archäologie*, pages 545–567. de Gruyter, 1980. 8

[9] T. Fawcett. An Introduction to ROC Analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. 7

[10] D. Fisseler, F. Weichert, G. Müller, and M. Cammarosano. Towards an interactive and automated script feature analysis of 3D scanned cuneiform tablets. In *The 4th Conference on Scientific Computing and Cultural Heritage (SCCH)*, pages 1–10, 2013. 2

[11] D. Fisseler, F. Weichert, G. Müller, and M. Cammarosano. Extending Philological Research with Methods of 3D Computer Graphics Applied to Analysis of Cultural Heritage. In *12th Eurographics Workshop on Graphics and Cultural Heritage (GCH)*, pages 165–172, 2014. 1

[12] J. Gravesen. Adaptive Subdivision and the Length and Energy of Bézier Curves. *Computational Geometry*, 8(1):13–31, 1997. 3

[13] H. Hameeuw and G. Willems. New Visualization Techniques for Cuneiform Texts and Sealings. *Akkadica*, 132(2):163–178, 2011. 1

[14] S. Jakob. *Die mittelassyrischen Texte aus Tell Chuēra in Nordost-Syrien*, volume 3 of *Ausgrabungen in Tell Chuēra in Nordost-Syrien*. Harrassowitz, 2009. 7

[15] J. Kantel, P. Damerow, S. Köhler, and C. Tsouparopoulou. 3D-Scans von Keilschrifttafeln – ein Werkstattbericht. In *26. DV-Treffen der Max-Planck-Institute*, pages 41–62. Gesellschaft für wissenschaftliche Datenverarbeitung, 2010. 1

[16] D. Kirkpatrick. Efficient Computation of Continuous Skeletons. In *Proceedings of the 20th Annual IEEE Symposium on Foundations of Computer Science*, pages 18–27, 1979. 3

[17] S. Kumar, D. Snyder, D. Duncan, J. Cohen, and J. Cooper. Digital Preservation of Ancient Cuneiform Tablets Using 3D-Scanning. In *Proceedings of Fourth International Conference on 3-D Digital Imaging and Modeling*, pages 326–333, 2003. 1

[18] M. Kuntz. Clustering SVG Shape. In *8th International Conference on Scalable Vector Graphics*, 2010. 2

[19] D. Lee. Medial Axis Transformation of a Planar Shape. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 4(4):363–369, 1982. 3

[20] H. Mara and S. Krömker. Vectorization of 3D-Characters by Integral Invariant Filtering of High-Resolution Triangular Meshes. In *12th International Conference on Document Analysis and Recognition (ICDAR)*, pages 62–66, 2013. 2

[21] H. Mara, S. Krömker, S. Jakob, and B. Breuckmann. GigaMesh and Gilgamesh – 3D Multiscale Integral InvariantCuneiform Character Extraction. In *Proceedings of the 11th International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST)*, 2010. 2

[22] U. Montanari. Continuous Skeletons from Digitized Images. *Journal of the ACM (JACM)*, 16(4):534–549, 1969. 3

[23] O. Neugebauer, editor. *Register, Glossar, Nachträge, Tafeln*, volume 2 of *Mathematische Keilschrift-Texte*. Springer, 1935. 1, 7, 8

[24] D. Powers. Evaluation: From precision, recall and f-measure to roc, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2(1):37–63, 2011. 7

[25] F. Preparata. The Medial Axis of a Simple Polygon. In *Mathematical Foundations of Computer Science 1977*, pages 443–450. Springer, 1977. 3

[26] W. von Soden. *The ancient Orient: an introduction to the study of the ancient Near East*. Wm. B. Eerdmans Publishing Co., 1994. 1

[27] L. Watkins and D. Snyder. The Digital Hammurabi Project. In *Proceedings of Museums and the Web (MW)*, 2003. 1

[28] G. Willems, F. Verbiest, W. Moreau, H. Hameeuw, K. Van Lerberghe, and L. Van Gool. Easy and Cost-Effective Cuneiform Digitizing. In *The 6th International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST)*, pages 73–80, 2005. 1

[29] C. Zahn and R. Roskies. Fourier Descriptors for Plane Closed Curves. *IEEE Transactions on Computers (TC)*, 21(3):269–281, 1972. 2

21st *Computer Vision Winter Workshop*
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

2D tracking of Platynereis dumerilii worms during spawning

Daniel Pucher, Walter G. Kropatsch, Nicole M. Artner
Pattern Recognition and Image Processing (PRIP)
Vienna University of Technology, Austria
`http://www.prip.tuwien.ac.at`

Stephanie Bannister , Kristin Tessmar-Raible
Max F. Perutz Laboratories
University of Vienna, Austria
`https://www.mfpl.ac.at/`

Abstract. *Platynereis dumerilii are marine worms that reproduce by external fertilisation and exhibit particular swimming behaviours during spawning. In this paper we propose a novel worm tracking approach that enables the 2D tracking and feature extraction during the spawning process of these worms. The gathered data will be used in the future to characterise and compare male and female spawning behaviours.*

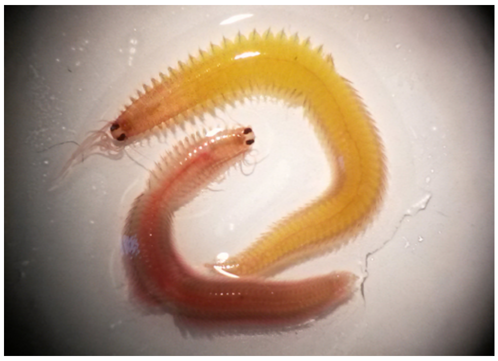


Figure 1. Image of a male (red) and female (yellow) worm.

1. Introduction

Platynereis dumerilii are marine polychaete worms (Lophotrochozoa, annelida, nereididae), which swim only when sexually mature, in order to reproduce. The timing of reproductive spawning events in this species is synchronized with the moon phase, whereby spawning in nature occurs primarily during new moon. This together with chemical pheromone signaling allows mature male and female worms to locate one another and engage in spawning behaviors that constitute the nuptial dance. See Figure 1 for an image of a male and female worm.

The spawning behaviors of male and female worms are important for successful fertilization of the gametes. The spawning process consists of four general phases: Pre-spawning, engaged spawning, gamete release and post-spawning. During pre-spawning, male and female worms typically swim independently of one another, usually with lower speeds, and display a linear body shape. Engaged spawning is initiated when male and female worms come into close contact and sense chemical pheromones secreted into the water by the opposite sex. This is accompanied by a noticeable change in swimming behavior for both sexes: swimming speeds increase (particularly for males), and worms either begin to swim in circles, or swim in tighter circles (particularly for females). Other changes in the plane of swimming are more frequently observed in both sexes during engaged spawning behavior. During gamete release, sperm and eggs are secreted into the water, which particularly for female worms, results in a dramatic change in body area, length and overall shape. The time individual spawning phases take varies and depends on the worms and their willingness to engage. Some worm pairs are better matches than others which can result in shorter spawning phases.

Our goal is to analyse these spawning behaviours in a quantitative manner, and to characterise and compare male and female-specific spawning behaviours.

2. Task formulation

The aim is to develop methods that enable the tracking of spawning worms from captured videos and extract features to quantify behaviours. For the tracking, it is important that we distinguish male and female worms in every frame of a captured video, label them and keep track of those labels. This paper focuses on the extraction of features for the analysis of behaviours. The tracking task is simplified by only considering videos with single worms. In order to quantify behaviours, we currently extract the following worm features:

1. *Skeleton*

   The skeleton describes the center line of a worm and is defined by two endpoints and an ordered list of points between them. We use the skeleton to calculate the curvature of a worm and to generate a normalized shape representation.

2. *Head position*

   The head position is an important feature for the calculation of the velocity and the worm trajectory. We define it as an endpoint of the skeleton. The tangent of the skeleton in this endpoint can give us information on the orientation of the worm.
   To choose the right endpoint, we currently select it at the beginning of a video and keep track of that selection.

3. *Velocity*

   As the swimming speeds increase for both sexes, the velocity is a good indication for the beginning of the engaged spawning.

4. *Trajectory of the worm head*

   The mapping of the swimming trajectories gives us information on the interaction between two worms. Furthermore, for individual worms, the curvature of the trajectory can be compared to the curvature of the worm. A high correlation indicates a circular movement and increases the robustness of the curvature estimation. The trajectory can also give an indication on where we can expect the worm to be in a following frame.

5. *Curvature*

   Measurements of body curvature tell us both about the gross and fine body movements of the worm during the different spawning phases. The gross curvature of the worm's body in general provides information on the directionality of swimming. For example, a mostly straight linear profile would be indicative of linear swimming, while smoothly curved body profiles would indicate circular swimming. Good resolution of finer-scale body curvatures along the length of the worm is also important. For example, a linear profile with several bends could indicate an acceleration of swim speed, or 'wriggling' movements, depending on the amplitude of the curvatures. Such wriggling movements can be seen for males when they are stopping to secrete sperm. Similarly, as gametes are released from the tail, mapping fine-scale curvatures at the tip of the tail could be used to map gamete release events, or characterize sex-specific gamete release behaviours. For example, we have observed fast small tail flicks in males during sperm release, and curling of the tip of the tail in females just prior to egg release. The calculation of the curvature is based on the skeleton of the worm.

6. *Normalized shape*

   To make the comparison of different worms (or of the same worm at different times in a video) easier, we create normalized shape representations. To do this we follow a recent strategy which is known as co-registration where shapes are first straightened or flattened to then register different views/deformations of the same normalized shape [1].

7. *Length and area*

   During the gamete release phase the body length and area changes, especially for female worms. Therefore, these features are a good indicator for the beginning of this phase.

3. Existing tracking approaches

The tracking of animals and the extraction of features to quantify behaviours is not a new field of application. Caenorhabditis elegans (C. elegans) are roundworms that have been used as model systems in neuroscience for years and the demand for robust computational methods has lead to a number of different tracking systems like Nemo

[10], OptoTracker [8] or a tracking system developed by Chatenay and Schafer [2]. These worm trackers are capable of tracking worms and extracting a variety of different features. Unfortunately they are developed for C. elegans worms who differ in their appearance as well as their locomotion from Platynereis dumerilii. Furthermore some of them are only capable of tracking single worms and others terminate the tracking of animals if they collide and assigns new tracks after they separate again. This does not guaranty a continuous trajectory of a single worm for a whole video sequence, which is an important requirement for our behaviour analysis. Other animal tracking projects like AnTracks (www.antracks.org) or ”Visual Ants Tracking” by Ying [11] are capable of tracking animals, but do not allow the extraction of features, which match our requirements.

Therefore, we propose a new system that is capable of tracking Platynereis dumerilii worms and offers feature extraction including a new method to compute normalized shape forms.

4. Experimental setup

The setup of our worm tracker consists of a light-tight box, a mounted infrared camera and an ordinary PC to capture the videos. The worms are placed inside a spherical bowl we refer to as arena. Figure 2 shows the arena with two worms.

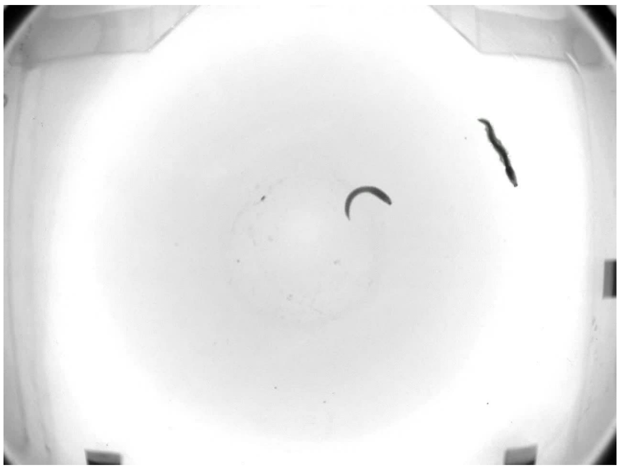


Figure 2. Image of the arena with two worms taken from a captured video.

The camera takes videos at a size of 1280x960 pixels with 60 frames per second. The infrared camera is important as the spawning in nature occurs at night and we want to reproduce this environment in the lab. The single camera setup has some limitations regarding 3D movements of the worms, as they might conceal parts of their body from the cameras viewpoint, resulting in a flawed representation. Analysis of spawning videos have shown, that the worms move horizontally near the water surface. Therefore, we decided to use this single camera setup and neglect the few cases where the gathered data is flawed due to 3D movement. Although, we might change the setup in the future using three cameras instead of one to solve the issue with the 3D movement.

5. Segmentation and tracking

Basically, male and female worms can be distinguished by their color and anterior / posterior segment border, which can be seen in Figure 3.

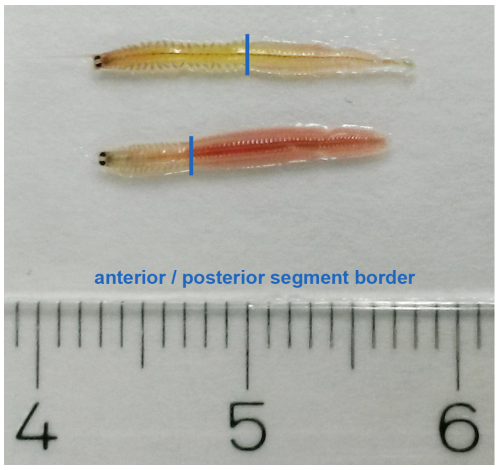


Figure 3. Image of a female (top) and a male (bottom) worm with their segment borders (Scale in cm).

The segment border divides a worm into a head and a tail part and the position of the border is different for male and female worms. Relative to their whole body length, male worms have a longer tail than female worms. Therefore, the segment border is closer to the head. Unfortunately, the segment border is not always clearly visible.

Figure 4 shows three frames of the same worm in the same video just a few seconds apart. These frames illustrate the problem with the segment border. The worms tend to turn sideways when moving fast and in such cases the segment border is not visible to the camera. This prevents us from using the segment border as a feature to distinguish male and female worms.

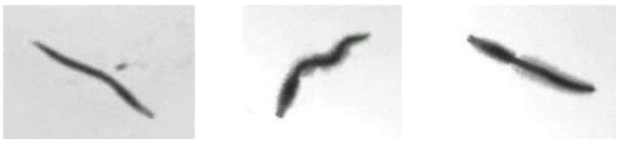


Figure 4. Three different frames of a single worm taken from the same video just a few seconds apart. In the first frame on the left the worm turned sideways, therefore the segment border is not visible.

Furthermore, due to the infrared capture, we do not have color information in the captured videos and the available grayvalues are not distinctive enough to distinguish male and female worms.

Therefore, we choose an approach that does not rely on the shape and color of the worms, but on their continuous motion over time. First, we label the worms at the beginning of a captured video. Then, we calculate the distance between the head positions in consecutive frames and assign the label based on the smaller deviation. This approach already works well for single worms, but it is too simple to track pairs of worms, as they tend to overlap and the distance of head positions alone is not a robust criteria.

In this paper, we focus on tracking of single worms. We will extend our approach to setups with worm pairs in the future. Although, we only track single worms at the moment, it is still possible to analyse separate spawning behaviours in male and female worms, as we add eggs or sperm manually to the arena and the worms react to them. This allows us to analyse isolated spawning behaviours.

To track a single worm we first need to segment it from the background. We do this with a simple background subtraction for every frame of the video. For the subtraction, it is important that there is at least one frame at the beginning of the video with an empty arena, which serves as the background image. As this image serves as the background image for the whole video, it is assumed that the arena does not move during the video.

After the background subtraction the resulting image is converted to a binary image, based on a global threshold. The binary image gives us a collection of regions that correspond to changes in relation to the empty arena. Ideally there is only one region for a single worm. Unfortunately, as the worm produces some noise when moving in the arena (particles or bubbles in the water, reflections on the edge of the arena) we also get some noise in our binary image. Therefore, we only consider regions whose area is above a given threshold as worms. As the regions generated by noise are very small, this approach works very well in our current setup.

6. Feature extraction

Features are extracted for every frame of the captured video and are based on the binary region and/or the skeleton of a worm.

6.1. Skeleton

Given the binary region of the worm, we use morphological thinning to compute the skeleton. In our case this approach is superior to the morphological skeletonization with the medial axis transfrom algorithm as the latter tends to generate more spurious branches. See Figure 5 for a comparison between the two approaches for a sample worm. The thinning approach also tends to create a smoother skeleton.

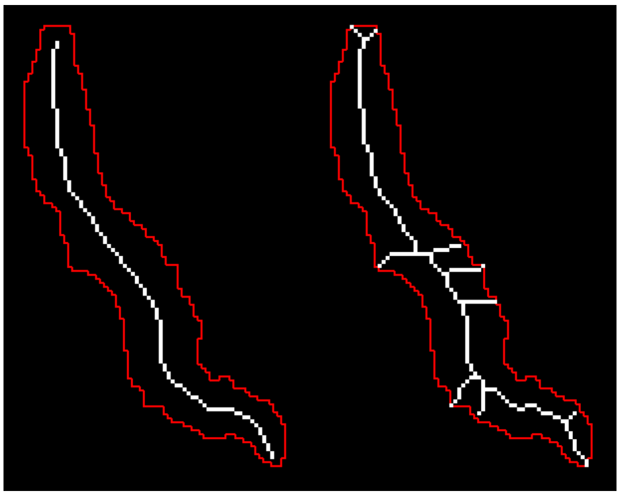


Figure 5. Illustration of the worm skeletons (white) computed from the binary segmentation image (outlined by the red line). The left skeleton was computed using the morphological thinning, the right one using skeletonization (MAT) technique.

The skeleton is defined as an 8-connected curve $s = \langle p_1, ..., p_n \rangle$ where $p_i = (x_i, y_i)$ with $i = 1, ..., n$. We order the points p_i of the skeleton from head to tail by comparing the endpoints of the skeleton in one frame with the endpoints in the previous frame. The position of the head in the first frame of the video has to be specified by the user.

6.2. Head position

As we store the skeleton points in a head to tail order, we get the head position from the first point $p_1 = (x_1, y_1)$ in s.

6.3. Trajectory of the worm head

Given a list of head positions $h = \langle h_1, ..., h_n \rangle$ where $h_i = (x_i, y_i)$ with $i = 1, ..., n$ we can generate a connected trajectory. Furthermore, its curvature can be computed to provide information on the swimming direction. Figure 11 shows a section of the trajectory from a single worm video.

6.4. Velocity

Velocity is defined as the rate of change of position with respect to time. We get the change of position by considering the head positions of a single worm in two consecutive frames and calculating the Euclidean distance between the two positions. The time between two frames is given by $1/fps$, where fps is the number frames per second of the video source.

6.5. Curvature

According to Hermann and Klette [6], the estimation of the curvature along a discrete curve can roughly be divided into three categories: the derivative of the tangent angle, the derivative of the curve and the radius of the osculating circle. We chose a method based on osculating circles as it is fast and the implementation is simple. Gray [5] defines the osculating circle of a curve C at a given point P in the continuous space as the circle that has the same tangent as C at point P as well as the same curvature. We approximate these circles with the circumscribed circles of triangles on the discrete skeleton curve. Casey [3] defined the circumscribed circle as the unique circle that passes through each of the triangle's three vertices.

Given the definition for the skeleton s at the beginning of this section, let k be $1 \leq k \leq \lfloor \frac{n}{2} \rfloor$ if n is odd and $1 \leq k \leq (\lfloor \frac{n}{2} \rfloor - 1)$ if n is even, where n is the number of points on the skeleton. For each point p_i on s we define a triangle between the three points p_{i-k}, p_i and p_{i+k}. Then the radius of the triangles circumscribed circle is computed to calculate the curvature at p_i. See Figure 6 for a visualization.

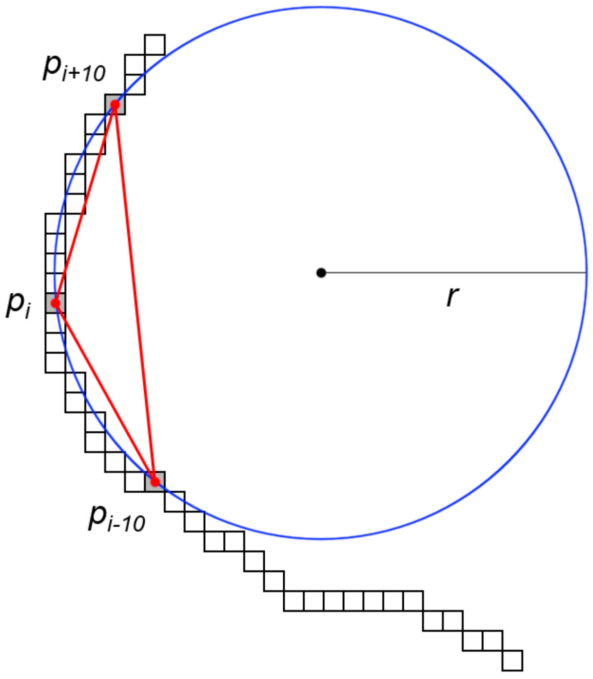


Figure 6. Illustration of the circumscribed circle (blue) for a single point p_i on a skeleton. The circle passes through every vertex of the triangle (red) formed by the points p_{i-k}, p_i and p_{i+k} with $k = 10$.

The radius of the circumscribed circle is defined as $radius = \frac{abc}{4*area}$, where a, b and c correspond to the edge lengths of the triangle and $area$ is the area of the triangle. The area of a triangle is given by $area = abs(\frac{1}{2} * determinant)$ where $determinant$ refers to the determinant of the triangle matrix, which is formed from the three triangle points:

$$determinant = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$$

As the sign of the determinant gives an indication on the orientation of the triangle and therefore an indication on the direction of the curvature, we don't use the absolute value. So we define the area as $area = \frac{1}{2} * determinant$. With this information the radius is then defined as $radius = \frac{abc}{2*determinant}$. The curvature is given by the inverse of the radius $c = \frac{1}{radius}$. As we do not take the absolute value of the determinant when calculating the radius, the curvature is a signed value that is positive if the curvature is on the right side and negative if the curvature is on the left side of the skeleton curve. See Figure 7 for a visualization.

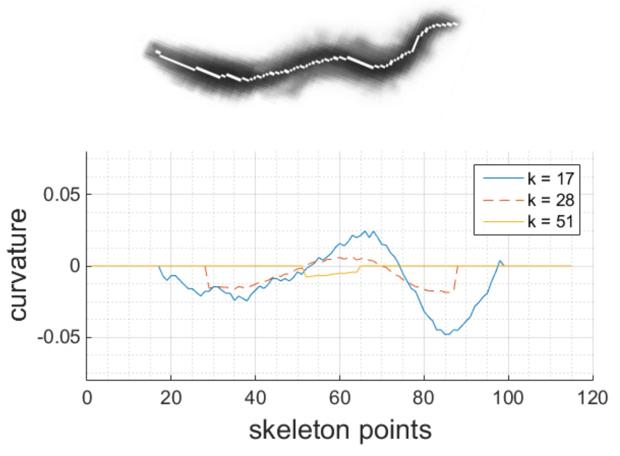


Figure 7. Image of a worm with its skeleton (top) and a plot of the estimated curvature of the worm for different k (bottom).

An important factor in the accuracy of this algorithm is the parameter k that defines a neighbourhood around the point of interest on the curve. We tested the accuracy on a discrete circle with a radius of 40 pixels using Bresenham's circle algorithm. The results can be seen in Figure 8. The Parameter k starts at $0.05 * n$ as the error gets too big for smaller values. As k increases we can see that the error gets smaller. The same is true for a constant k but an increasing radius, which corresponds to the multigrid convergence theorem, where we expect the accuracy to increase as the grid resolution (or in our case the circle radius) increases [7, Chapter 10].

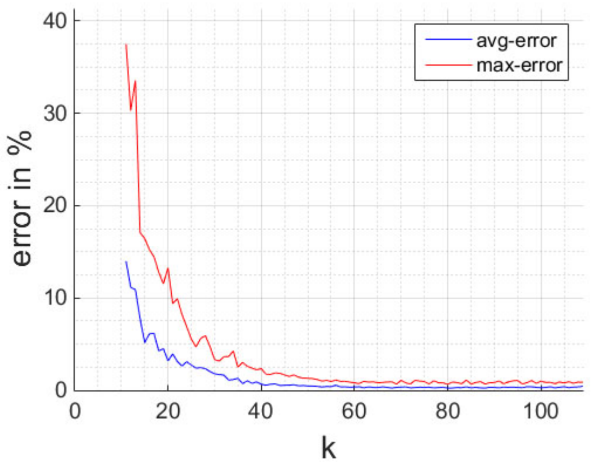


Figure 8. Plot of the avg- and max-error for the curvature estimation of a circle with radius 40 and increasing k.

So the accuracy gets better with increasing k. Unfortunately, this accuracy comes at a price, as small curvatures are overlooked in the process if k is too big. Another problem with a fixed neighbourhood k are points at the beginning and the end of the skeleton curve. For points p_a with $a \leq k$ there are no neighbourhood points p_{a-k} defined, as the index would become null or negative. The same is true for points p_b with $b > n - k$ where no neighbourhood points p_{b+k} are defined, as the index would get bigger than n. We currently solve this problem by disregarding those points on the curve. In Figure 7 the curvature values always start at the index $1 + k$ and end at the index $n - k$. Another problem is the determination of a good value for the parameter k. In Figure 7 the blue line shows the curvature for $k = 17$ which equals $0.15 * n$ and gives the best results on the tested worms.

6.6. Normalized shape

We achieve the normalized shape representation of a worm with a backward medial axis transform approach. The starting point is the distance transform of the binary worm image which labels each pixel with the Euclidean distance to the nearest boundary in the binary image. For every point p_i of of sorted list s of skeleton points, we use the coordinates to look up the distances in the distance transform. Those distances then serve as the radii for the circles. See Figure 9 for a visualization.

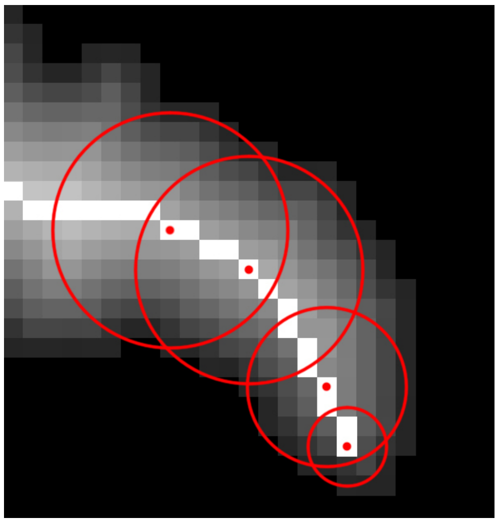


Figure 9. Part of the distance transform of a worm with circles drawn for four points on the skeleton.

To get a suitable representation of the worm, the distances between the skeleton points in the video frame need to stay the same on the normalized shape representation. Therefore the Euclidean distance between the points is calculated and taken into account when drawing the circles. Figure 10 shows the results of this method, where in the first visualization only the outlines of a few circles are drawn to show the general idea behind this approach.

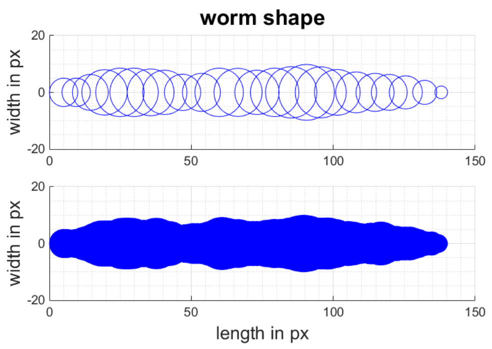


Figure 10. Plots of the normalized representation of a worm using only the outlines of 24 circles to visualize the general idea (top) and a complete shape visualization with all 115 filled circles (bottom) for that worm.

6.7. Length

To calculate the length of a worm, we use the geodesic distance of its skeleton plus the radii of the circles at the first and last skeleton point. The circle radii are needed as our skeleton endpoints do not lie at the edge of the worm. The geodesic distance is computed using the Euclidean distance.

6.8. Area

For the area of a worm, we simply calculate the sum of all foreground pixels of the binary image of the worm, which is the zeroth moment.

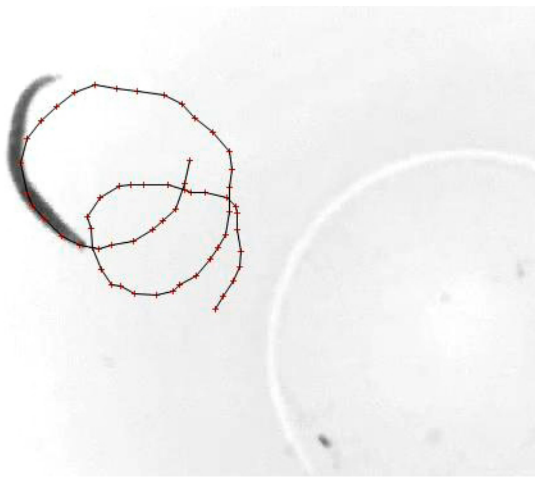


Figure 11. Part of a video frame with an overlay of the head trajectory. The trajectory is taken from a short sequence of a single worm video.

7. Single worm experiments

Some experiments with single female worm videos were conducted. Figure 12 shows two plots of smoothed worm lengths. For the smoothing a moving average filter was applied to the original data. The plots show the length of the worms around the time of the gamete release where the female worms secrete their eggs into the water and get smaller and therefore shorter. This can also be observed in the plots.

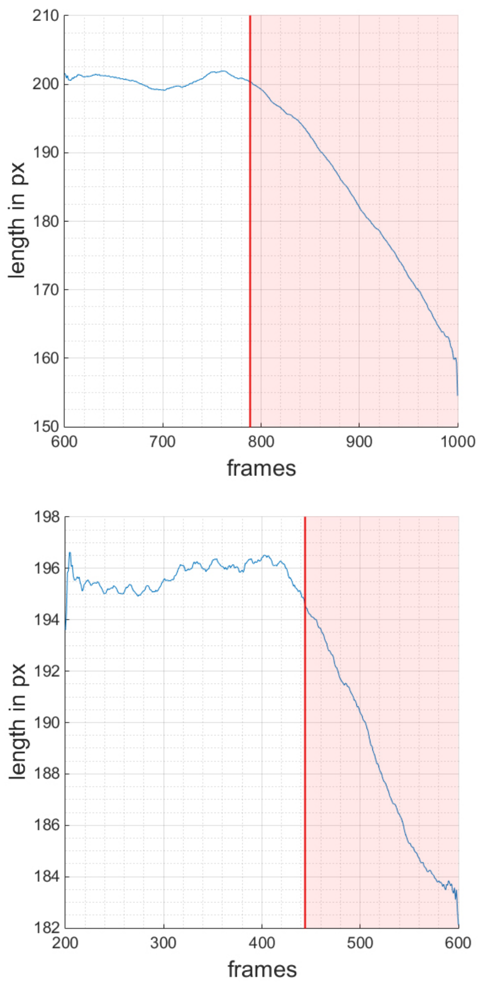


Figure 12. Two plots of smoothed worm lengths for two different female worms right around the time of the gamete release (marked in red).

Figure 13 shows how the length of a female worm changes during an entire spawning process. Annotation A marks a special case where the worm is overlapping itself resulting in a faulty binary area and skeleton. The problem here is the 3D movement of the worm.

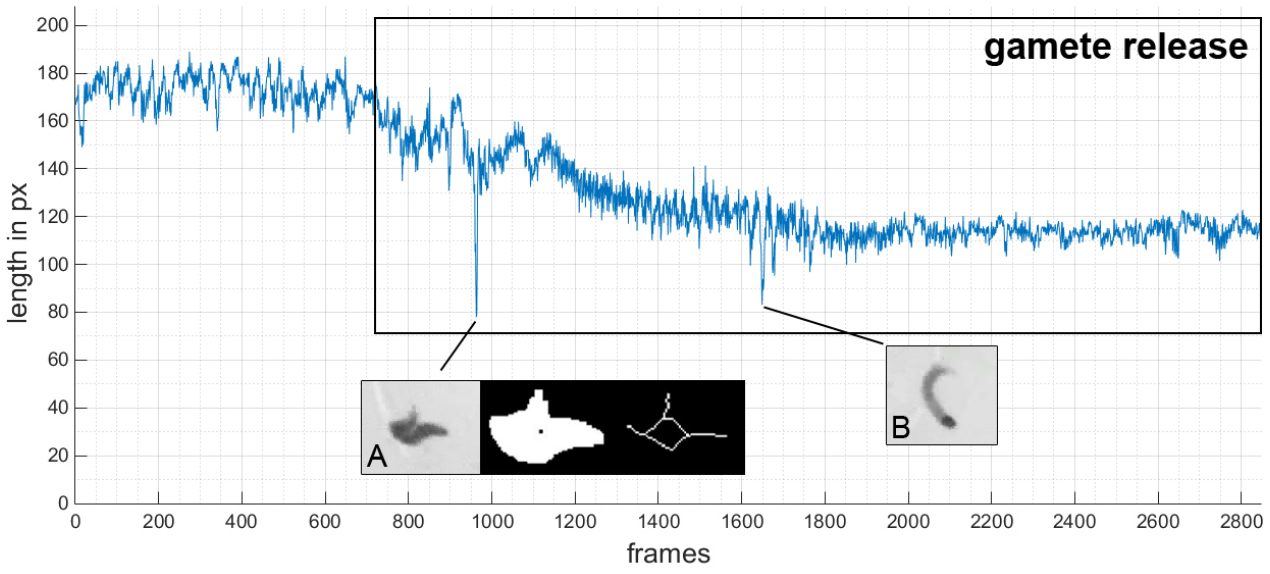


Figure 13. Change of worm length over time. During the gamete release the worm gets shorter. Annotation A: Wrong skeleton due to 3D movement of the worm. Annotation B: Wrong length due to 3D movement of the worm.

Another special case where the 3D movement also results in error-prone data is marked with annotation B. Here the end of the tail is not visible to the camera which makes the worm appear shorter in the video.

8. Conclusion and Future work

In this paper we proposed a novel worm tracking approach for Platynereis dumerilii worms, that enables both tracking and feature extraction from captured videos. Although our tracking approach is not suitable for tracking two worms in difficult cases our methods to extract worm features already show promising results.

The method we currently use to track single worms works for two worms if they are physically separated, but as they get close to each other or overlap, the current method might fail. In the future, we will extend the method to consider cases where the worms are close to each other or even overlap. Ideas to achieve this include the comparison of more features than just the head positions of consecutive frames. A combination of all other features could yield an appropriate approach in distinguishing male and female worms.

The current feature extraction is robust in most cases, but there exist special cases, where a single worm overlaps itself due to 3D movement in the water. This results in regions and skeletons, which do not represent the worm correctly and therefore the extracted features are flawed as well. One approach will be to look into the watershed method to segment the worms as it might be superior to the simple threshold based method we use now, especially for worms that overlap.

Our approach to compute the curvature also has some flaws and is not robust enough. In the future we will look into alternative approaches to compute the curvature of discrete curves. Other methods that try to estimate the osculating circles rely on digital straight segments (DSS) recognition [6] [4] and Roussillon and Lachaud [9] base their method around maximal digital circular arcs.

9. Acknowledgement

The research leading to these results has received funding from the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013)/ERC Grant Agreement 337011 to KT-R.

We also thank the anonymous reviewers for their valuable input.

References

[1] N. Aigerman, R. Poranne, and Y. Lipman. Lifted bijections for low distortion surface mappings. *ACM Trans. Graph.*, 33(4):69:1–69:12, July 2014. 2

[2] J. B. Arous, Y. Tanizawa, I. Rabinowitch, D. Chatenay, and W. R. Schafer. Automated imaging of neuronal activity in freely behaving

caenorhabditis elegans. *Journal of Neuroscience Methods*, 187(2):229 – 234, 2010. 3

[3] J. Casey. *A sequel to the first six books of the elements of Euclid, containing an easy introduction to modern geometry, with numerous examples.* 7^{th} *edition, revised and enlarged.* Dublin. Hodges. University Press Series. 184 S, (1895). 5

[4] D. Coeurjolly, S. Miguet, and L. Tougne. Discrete Curvature based on Osculating Circles Estimation, 2001. 4th International Workshop on Visual Form 2001, Capri, Italy, Springer Lecture Notes in Computer Science 2059, pages 303-312. 8

[5] A. Gray. *Modern Differential Geometry of Curves and Surfaces with Mathematica*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1996. 5

[6] S. Hermann and R. Klette. A comparative study on 2d curvature estimators. In *Computing: Theory and Applications, 2007. ICCTA '07. International Conference on*, pages 584–589, March 2007. 5, 8

[7] R. Klette and A. Rosenfeld. *Digital Geometry*. Morgan Kaufman Publishers, 2004. 6

[8] D. Ramot, B. E. Johnson, T. L. Berry, Jr, L. Carnell, and M. B. Goodman. The parallel worm tracker: A platform for measuring average speed and drug-induced paralysis in nematodes. *PLoS ONE*, 3(5):e2208, 05 2008. 3

[9] T. Roussillon and J.-O. Lachaud. Accurate curvature estimation along digital contours with maximal digital circular arcs . In *14-th International Workshop on Combinatorial Image Analysis (IWCIA)*, LNCS, pages 43–55. Springer, 2011. 8

[10] G. D. Tsibidis and N. Tavernarakis. Nemo: a computational tool for analyzing nematode locomotion. *BMC Neuroscience*, 8(1):1–7, 2007. 3

[11] F. Ying. *Visual ants tracking*. PhD thesis, University of Bristol, 2004. 3

21^{st} *Computer Vision Winter Workshop*
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

Significance of Colors in Texture Datasets

Milan Šulc, Jiří Matas
Czech Technical University in Prague, Faculty of Electrical Engineering,
Department of Cybernetics, Center for Machine Perception,
Technická 2, 166 27 Praha 6, Czech Republic
{sulcmila,matas}@fel.cvut.cz

Abstract. *This paper studies the significance of color in eight publicly available datasets commonly used for texture recognition through the classification results of "pure-color" and "pure-texture" (colorless) descriptors. The datasets are described using the state-of-the-art color descriptors, Discriminative Color Descriptors (DD) [15] and Color Names (CN) [28]. The descriptors are based on partitioning of the color space into clusters and assigning the image probabilities of belonging to individual clusters. We propose a simple extension of the DD and the CN descriptors, adding the standard deviations of color cluster probabilities into the descriptor. The extension leads to a significant improvement in recognition rates on all datasets. On all datasets the 22-dimensional improved CN^{σ} descriptor outperforms all original 11-, 25- and 50-dimensional descriptors. Linear combination of the state-of-the-art "pure-texture" classifier with the CN^{σ} classifier improves the results on all datasets.*

1. Introduction

Visual recognition based on texture and color are well established computer vision disciplines with several surveys available, e.g. [3, 10, 19, 20, 27, 30]. The state-of-the-art in texture recognition has been recently dominated in terms of accuracy by methods based on deep Convolutional Neural Networks (CNNs) [5, 6], yet the pre-CNN approaches may be preferable in real-time applications for their performance without parallel processing. Although it has been shown that several texture description methods can benefit from adding color information [13], a large number of the pre-CNN texture recognition techniques has been evaluated only on gray-scale images. Since many publicly available datasets used for texture recognition contain color information, we decided to evaluate the accuracy of color-statistics based methods to measure the significance of color information in the datasets.

The first contribution of this paper is a study of the significance of color information in available datasets commonly used for evaluation of texture recognition methods. In total we evaluate 8 texture datasets, namely FMD (Flickr Material Database), ALOT (A Lot Of Textures), KTH-TIPS (Textures under varying Illumination, Pose and Scale), KTH-TIPS2a, KTH-TIPS2b, CUReT (Columbia-Utrecht Reflectance and Texture), VehApp (Vehicle Appearance) and AniTex (Animal Texture).

The second contribution of the paper is an improvement of the state-of-the-art color descriptors, Discriminative Color Descriptors (DD) [15] and Color Names (CN) [28]. DD and CN are based on partitioning of the color space into clusters and assigning each color the probabilities of belonging to individual clusters. Our extension to the DD and the CN descriptors adds the standard deviation for each color cluster to the descriptor. This leads to an improvement in recognition rates on all 8 tested datasets, as shown in the experiments in Section 5.

The third contribution of the paper are experiments combining a state-of-the-art "pure-texture" descriptor with the improved CN^{σ} descriptor, leading to further increase in recognition accuracy.

The rest of the paper is organized as follows: Sections 2.1 and 2.2 review the state of the art in texture and color recognition, respectively. The selected "pure-color" image descriptors and our extension to them are introduced in Section 3. Publicly available color-image databases commonly used for texture classification are described in Section 4. Section 5 describes the experiments and presents the results.

The observations are discussed and conclusions are drawn in Section 6.

2. State of the Art

2.1. Texture-Based Classification

A large number of texture recognition techniques has been proposed, many of them being described in the surveys [3, 19, 20, 30]. In this section we only review the recent development and the state-of-the-art.

Several recent texture recognition algorithms report excellent results on standard datasets while ignoring the available color information. A number of them is based on the popular Local Binary Patterns, such as the Pairwise Rotation Invariant Co-occurrence Local Binary Pattern of Qi et al. [22] or the Fast Features Invariant to Rotation and Scale of Texture of Sulc and Matas [26]. A cascade of invariants computed by scattering transforms was proposed by Sifre and Mallat [24] in order to construct an affine invariant texture representation. Mao et al. [18] use a bag-of-words model with a dictionary of so called active patches: raw intensity patches that undergo further spatial transformations and adjust themselves to best match the image regions. While the Active Patch Model doesn't use color information, the authors claim that adding color will further improve the results. Cimpoi et al. [4], using Improved Fisher Vectors (IFV) for texture description, show further improvement when combined with describable texture attributes learned on the Describable Textures Dataset (DTD) and with color attributes.

Recently, Cimpoi et al. [5, 6] pushed the state-of-the-art in texture recognition using a new encoder denoted as FV-CNN-VD, obtained by Fisher Vector pooling of a very deep Convolutional Neural Network (CNN) filter bank of Simonyan and Zisserman [25]. The CNN filter bank operates on (preprocessed) RGB images. The method achieves state-of-the-art accuracy, yet may not be suitable for real-time applications when evaluated without a high-performance GPU.

2.2. Color Statistics for Classification

Color information is processed by many state-of-the-art descriptors in Computer Vision, including the neurocodes of Deep CNNs or different extensions of SIFT incorporating color. Yet we are interested in simpler color statistics, not making use of spatial information.

Standard approaches to collect color information include color histograms (based on different color representations), color moments and moment invariants. Sande et al. [27] provide an extensive evaluation of such descriptors. The Color Names (CN) descriptor by Weijer et al. [28] is based on models learned from real-world data obtained from Google by searching for 11 color names in English. The Color Names have shown to be a successful color attribute for object detection [12] and recognition [14]. The model assigns each pixel the probability of belonging to one of the 11 color clusters. A similar approach is used by the Discriminative Color Descriptor (DD) of Khan et al. [15], where the color values are clustered together based on their discriminative power in a classification problem with the objective to minimize the drop of mutual information of the final representation.

Khan et al. [13] study the strategies of combining color and texture information. They carried out a comparison of pure color descriptors on the publicly available KTH-TIPS2a, KTH-TIPS2b, and FMD datasets, and on another small dataset denoted as Texture-10. Since the results of Color Names and Discriminative Color Descriptors outperformed other color descriptors in texture classification, we will describe the usage of CN and DD in more detail in Section 3 and use the models in our experiments in Section 5.

3. Selected Color Descriptors

Based on the findings of Khan et al. [13] and on our preliminary results, we consider the Color Names [28] and Discriminative Color Descriptors [15] the best match for our experiments for their superior classification accuracy.

While each of the approaches creates the color models based on a different criteria, the result is a soft assignment of clusters to each RGB value. In both cases the assignment is performed using a lookup table, which creates a mapping from RGB values to probabilities over C clusters c_i, i.e. $p\left(c_i \mid x\right)$. In this work we use the lookup tables provided by the authors of the methods, i.e. the 11-dimensional Color Names representation by [28] and the universal color 11-, 25- and 50-dimensional representations by [15].

The models assume uniform prior over the color names $p(c_i)$. The conditional probabilities for each cluster c_i given an image I are computed as an average over all N pixels x_n in the region:

$$p\left(c_i \mid I\right)=\frac{1}{N} \sum_{x_n \in I} p\left(c_i \mid x_n\right) \tag{1}$$

The standard descriptor D for image I is then a vector containing the probability of each cluster:

$$D(I)=\begin{bmatrix} p\left(c_1 \mid I\right) \\ p\left(c_2 \mid I\right) \\ \vdots \\ p\left(c_C \mid I\right) \end{bmatrix} \tag{2}$$

We propose to add another statistics to the color descriptor, the standard deviation of the color cluster probabilities in the image:

$$\sigma(c_i \mid I)=\sqrt{\frac{1}{N} \sum_{x_n \in I}\left[p\left(c_i \mid x_n\right)-p\left(c_i \mid I\right)\right]^2} \tag{3}$$

We concatenate the standard deviations to the original descriptor to get the extended representation:

$$D^{\sigma}(I)=\begin{bmatrix} p\left(c_1 \mid I\right) \\ p\left(c_2 \mid I\right) \\ \vdots \\ p\left(c_C \mid I\right) \\ \sigma(c_1|I) \\ \sigma(c_2|I) \\ \vdots \\ \sigma(c_C|I) \end{bmatrix} \tag{4}$$

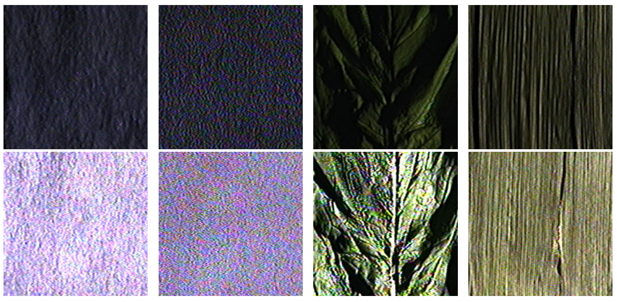

(a) Felt (b) Polyester (c) Lettuce leaf (d) Corn husk

Figure 1: Examples of four texture classes from the CUReT database.

4. Color Texture Datasets

This section reviews publicly available texture datasets that contain color information. Databases available only in the gray-scale version, such as Brodatz, UIUCTex or UMD, are omitted.

4.1. CUReT

The Columbia-Utrecht Reflectance and Texture (CUReT) image database [8] commonly used for texture recognition[1] contains 5612 images of 61 classes. There are 92 images per class, with different combinations of view- and illumination-direction.

The standard experimental protocol divides the dataset into two halves, using 46 training images per class for training and 46 images for testing. Examples of four selected classes from the dataset are displayed in Figure 1.

4.2. KTH-TIPS

The Textures under varying Illumination, Pose and Scale (KTH-TIPS) database [9, 11] was collected by Fritz, Hayman and Caputo with the aim to supplement the CUReT database, concerning texture variations in real-world conditions. The dataset contains 81 images for each of 10 selected materials, taken with different combination of pose, illumination and scale. The dataset contains samples of different color for several materials, each of the samples appears several times. In the experimental protocol the dataset is randomly divided into halves, 40 images per class are used for training and the remaining 41 images are used for testing. It is thus probable, that each of the samples appear in the training data set.

4.3. KTH-TIPS2

The KTH-TIPS2 database [2, 17], gathered by Mallikarjuna, Targhi, Hayman and Caputo, largely followed the procedure used for the previous KTH-TIPS database, with some differences in scale an illumination. The database also contains images from the previous KTH-TIPS dataset. The objective of the database is to provide a better means of evaluation: It contains 4 physical samples for each of 11 materials and images of no physical sample are present in both training and test set. The database contains 108 images of each physical sample. There are two version of the database: KTH-TIPS2a and KTH-TIPS2b. In

[1]http://www.robots.ox.ac.uk/ vgg/research/texclass/setup.html

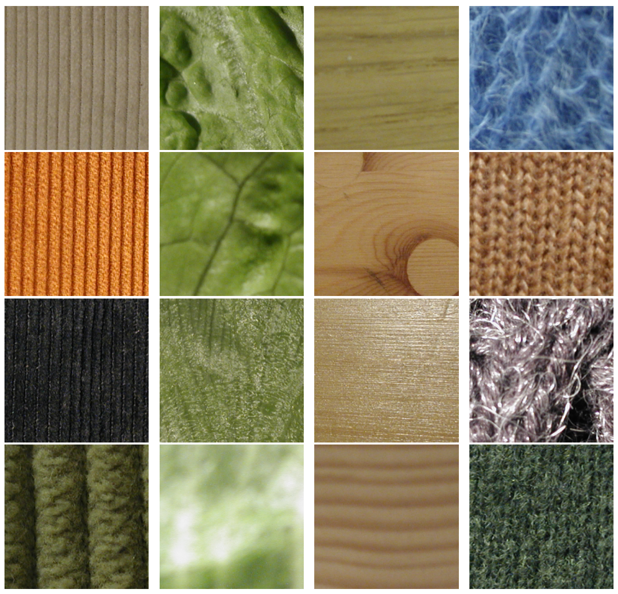

(a) Corduroy (b) Lettuce (c) Wood (d) Wool

Figure 2: Examples of four texture classes from the KTH-TIPS2 database. Each image belongs to a different physical sample.

the KTH-TIPS2a dataset, 144 images are are missing (namely there are four samples with only 72 images). In the experimental protocol, three samples from each class form the training set and the remaining sample is used for testing. In the case of the KTH-TIPS2b dataset, one sample forms the training set and the remaining three form the test set. Examples from all four samples of four selected classes from the database are displayed in Figure 2.

4.4. ALOT

The Amsterdam Library of Textures (ALOT) [1] is similar in spirit to the CUReT dataset, yet the number of materials is much higher: it contains 250 texture classes, 100 images per class. The pictures were taken under various viewing and illumination directions and illumination colors. For evaluation, 20 images per class are used for training and the remaining 80 images per class are used for testing. Examples from the ALOT database are displayed in Figure 3.

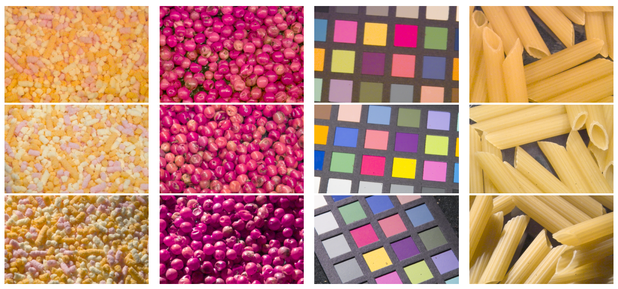

(a) Fruit sprinkles (b) Pepper (red) (c) Color calibration checker (d) Macaroni

Figure 3: Examples of four texture classes from the ALOT database.

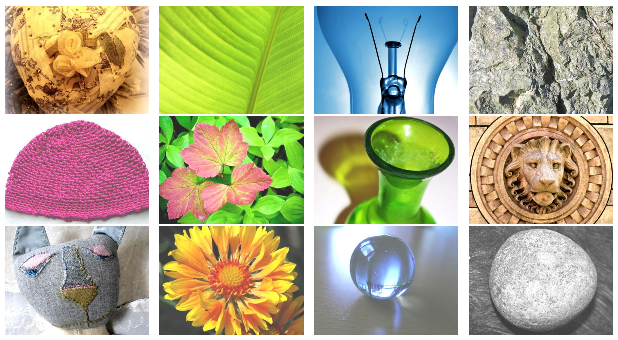

(a) Fabric (b) Foliage (c) Glass (d) Stone

Figure 4: Examples of four texture classes from the FMD database.

4.5. FMD

The Flickr Material database (FMD) was developed by Sharan et al. [23] with the intention of capturing a range of real world appearances of common materials. The dataset contains 1000 images downloaded manually from Flickr.com (under Creative Commons license), belonging to one of the following materials: Fabric, Foliage, Glass, Leather, Metal, Paper, Plastic, Stone, Water or Wood. There are exactly 100 images for each of the 10 material classes. Unlike the dataset described above, FMD was not primarily created for texture recognition, and it includes images of objects with various textures for each material. The dataset also includes binary masks for background segmentation. The standard evaluation protocol divides the images in each class into two halves, 50 images for training and 50 for testing. Examples from the FMD dataset are displayed in Figure 4.

4.6. AniTex

The Animal Texture dataset (AniTex) constructed by Mao et al. [18] contains 3120 texture patch images cropped randomly from the torso regions inside the silhouettes of different animals in the PASCAL VOC 2012 database. There are only 5 classes (cat, dog, sheep, cow and horse), 624 images each. The authors created the dataset to explore less ho-

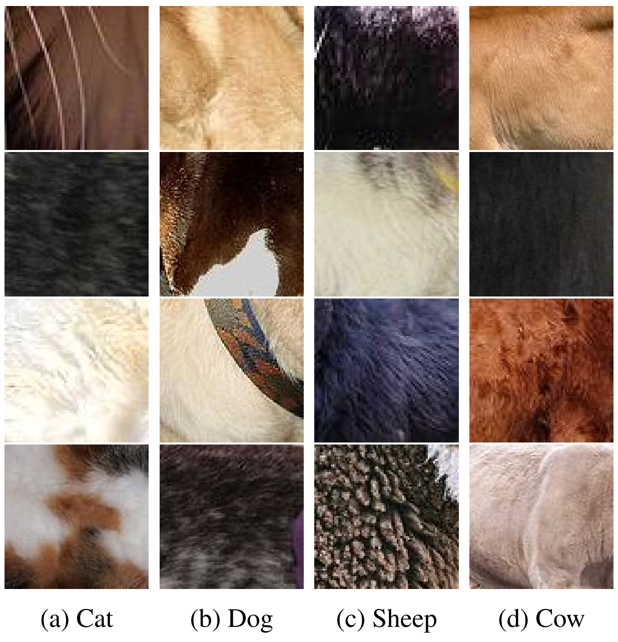


Figure 5: Examples of four texture classes from the AniTex database.

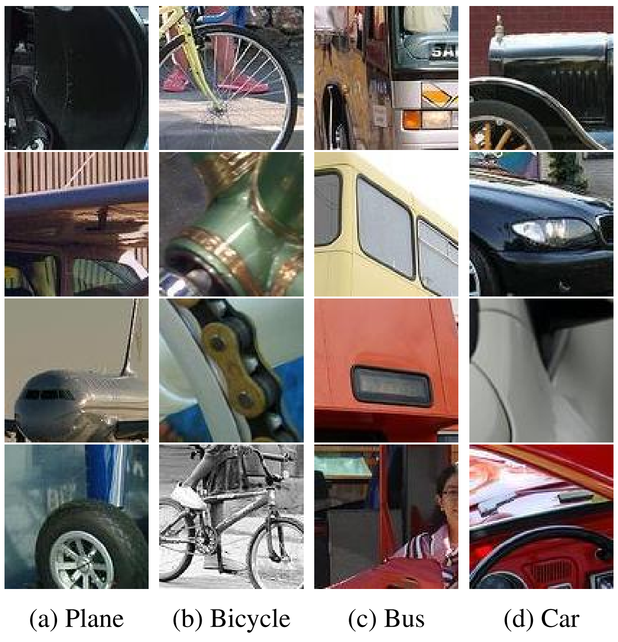


Figure 6: Examples of four texture classes from the VehApp database.

mogeneous texture and appearance than available in standard texture datasets. The patches in the dataset come from images under different conditions such as scaling, rotation, viewing angle variations and lighting condition change. For evaluation, the dataset is randomly divided into 2496 training and 624 testing images. Examples from the AniTex dataset are displayed in Figure 5.

4.7. VehApp

The Vehicle Appearance dataset (VehApp) was created by the same authors as AniTex [18] with the same intentions. It contains 13723 images cropped from PASCAL VOC images containing vehicles of 6 classes (aeroplane, bicycle, car, bus, motorbike, train). The images are evaluated in a way similar to AniTex: 80% images are randomly chosen into the training set, the remaining 20% is used for testing. Examples from the VehApp dataset are displayed in Figure 6.

5. Experiments

We compute 8 descriptors for each image in every database: the standard 11-dimensional Color Name descriptor CN and our extended 22-dimensional version CN$^\sigma$; the 11-, 25- and 50- Discriminative Color Descriptors DD11, DD25, DD50 and the extended versions DD11$^\sigma$, DD25$^\sigma$, DD50$^\sigma$ of double dimensionality.

The multiclass classification is then performed for each descriptor separately by combining binary SVM classifiers in a One-vs-All scheme. Linear SVM classifiers were used together with an approximate feature map of Vedaldi and Zisserman [29]. The χ^2 kernel approximations and the histogram intersection kernel approximations were considered, the latter was chosen based on slightly superior performance in preliminary experiments. The Platt's probabilistic output [16, 21] was used in order to estimate the posterior class probabilities to choose the result in the One-vs-All scenario. To minimize the effect of the random splits into training and testset, each experiment is performed 10 times on a different split, with the exception of the KTH-TIPS2 databases with 4 experiments based on the material samples.

All 8 color descriptors are compared in terms of class recognition accuracy in Table 1. The best published results of ”pure-texture” (color-less) methods and the results of the state-of-the-art FV-CNN [5] method are attached to the table for comparison. The comparison of the best ”pure-color” and ”pure-texture” results on all 8 datasets is illustrated in Figure 7.

An experiment on combining efficient classifiers of ”pure-texture” and ”pure-color” was performed as follows: Each image was described using the CN$^\sigma$ color descriptor (using the same method as above) and the Ffirst [26] texture descriptor (with $n_{\text{conc}} = 3$

Table 1: Recognition accuracy of selected color descriptors on publicly available databases commonly used for texture recognition.

	CUReT	TIPS	TIPS2a	TIPS2b	ALOT	FMD	AniTex	VehApp
# classes	61	10	11	11	250	10	5	6
CN	$85.9_{\pm0.6}$	$99.3_{\pm0.9}$	$46.7_{\pm2.0}$	$39.0_{\pm2.5}$	$51.0_{\pm0.5}$	$26.3_{\pm2.4}$	$38.0_{\pm2.0}$	$34.7_{\pm1.0}$
DD11	$68.7_{\pm0.9}$	$95.5_{\pm1.3}$	$43.5_{\pm6.5}$	$36.1_{\pm1.0}$	$38.2_{\pm0.4}$	$24.0_{\pm1.1}$	$32.4_{\pm1.6}$	$33.2_{\pm1.0}$
DD25	$83.4_{\pm0.8}$	$96.8_{\pm0.9}$	$44.0_{\pm7.6}$	$36.0_{\pm2.3}$	$60.9_{\pm0.5}$	$23.9_{\pm1.4}$	$36.0_{\pm1.7}$	$36.9_{\pm0.6}$
DD50	$87.7_{\pm1.0}$	$99.0_{\pm0.7}$	$46.9_{\pm4.8}$	$38.5_{\pm1.5}$	$65.5_{\pm0.4}$	$22.6_{\pm1.4}$	$37.4_{\pm1.1}$	$39.1_{\pm1.0}$
CN$^{\sigma}$	$\mathbf{94.2_{\pm0.6}}$	$\mathbf{99.8_{\pm0.3}}$	$51.7_{\pm5.7}$	$\mathbf{42.6_{\pm1.4}}$	$73.9_{\pm0.5}$	$\mathbf{28.0_{\pm2.2}}$	$\mathbf{41.7_{\pm1.8}}$	$39.1_{\pm0.7}$
DD11$^{\sigma}$	$81.9_{\pm0.8}$	$97.6_{\pm1.0}$	$48.5_{\pm3.8}$	$38.3_{\pm1.9}$	$60.1_{\pm0.5}$	$22.7_{\pm1.6}$	$35.9_{\pm2.1}$	$35.8_{\pm0.5}$
DD25$^{\sigma}$	$88.9_{\pm0.7}$	$99.4_{\pm0.3}$	$49.1_{\pm3.7}$	$39.9_{\pm4.5}$	$75.0_{\pm0.5}$	$23.9_{\pm1.1}$	$39.9_{\pm1.6}$	$39.3_{\pm0.7}$
DD50$^{\sigma}$	$91.0_{\pm0.7}$	$99.6_{\pm0.2}$	$\mathbf{53.2_{\pm4.6}}$	$42.0_{\pm2.8}$	$\mathbf{78.0_{\pm0.5}}$	$25.3_{\pm1.7}$	$38.9_{\pm0.8}$	$\mathbf{41.2_{\pm0.9}}$
FV-CNN[5]	$99.0_{\pm0.2}$	–	–	$81.8_{\pm2.5}$	$98.5_{\pm0.1}$	$79.8_{\pm1.8}$	–	–
Pure-texture	$99.8_{\pm0.1}$[24]	$99.7_{\pm0.1}$[4]	$88.2_{\pm6.7}$[26]	$76.0_{\pm2.9}$ [26]	$95.9_{\pm0.5}$ [26]	$57.4_{\pm1.7}$[22]	50.8[18]	63.4[18]

descriptors per image, each describing $c = 7$ consecutive scales). An approximate intersection kernel map is applied to both color and texture descriptors, which are then classified using the One-vs-All Support Vector Machines with Platt's probabilistic outputs. The final scores in Table 2 were then combined using 3 axiomatic approaches, denoted as:

1. PROD: The dot product of both of the scores is used for final decision.
2. SUM: The sum of both of the scores is used for final decision.
3. SUM$_{0.3}$: The weighted sum of both of the scores is used for final decision, where the weight of color is only 30% of the weight of texture, taking into account the lower performance of the color descriptors on most datasets.

In terms of combining probability distributions [7], the SUM and SUM$_{0.3}$ schemes represent a *linear opinion pool* and the PROD scheme represents a *logarithmic opinion pool*.

6. Observations and Conclusions

A set of experiments with color-based image descriptors was performed on 8 datasets commonly used for texture classification, leading to interesting insights in color-based classification and in the understanding of available texture-recognition datasets.

One can see that using the simple color descriptors is sufficient for excellent results in specific cases, such as the KTH-TIPS dataset, where materials of the same color appear in both training and test data. Satisfying results can also be obtained on the CUReT and ALOT datasets. The KTH-TIPS2a and KTH-TIPS2b datasets are more difficult for "pure-color" classification, since testing data may come from samples of different colors than training data, as illustrated in Figure 2. The FMD, AniTex and VehApp

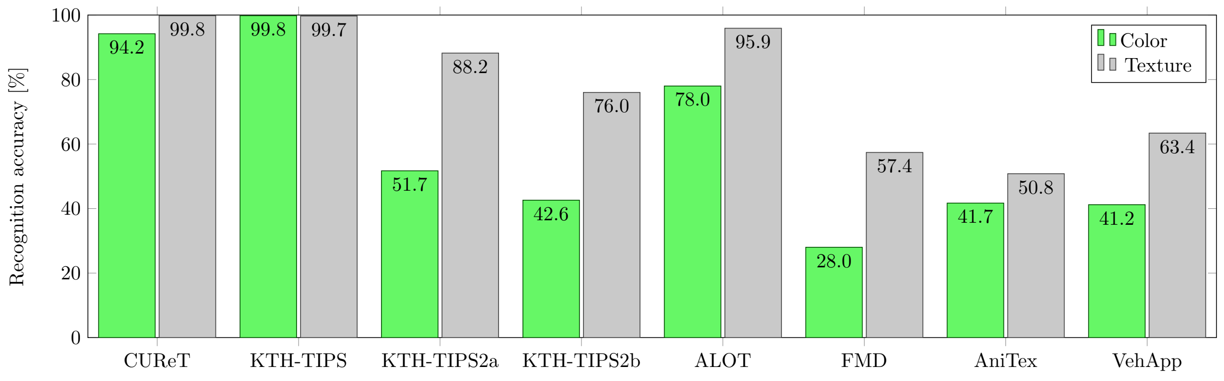


Figure 7: Comparison of the best published results of "pure-texture" descriptors and the best results obtained using "pure-color" descriptors.

Table 2: Recognition accuracy for combinations of "pure-texture" (Ffirst) and "pure-color" (CN^σ) descriptors.

	CUReT	TIPS	TIPS2a	TIPS2b	ALOT	FMD	AniTex	VehApp
# classes	61	10	11	11	250	10	5	6
CN^σ	94.24±0.60	99.83±0.31	51.73±5.71	42.64±1.43	73.86±0.46	27.98±2.20	41.67±1.77	39.07±0.67
Ffirst	99.65±0.09	99.51±0.53	88.29±6.77	76.60±4.29	96.43±0.23	50.22±1.90	45.72±1.78	54.41±0.66
PROD	99.41±0.15	99.98±0.08	68.13±5.06	60.12±4.06	94.65±0.20	46.58±2.37	49.97±1.50	56.47±0.76
SUM	99.04±0.20	**100.00±0.00**	77.59±5.87	60.35±5.13	92.06±0.29	45.70±2.47	**50.08±1.56**	56.56±0.98
$SUM_{0.3}$	**99.68±0.12**	99.85±0.26	**88.76±6.40**	**77.17±4.23**	**97.05±0.14**	**52.24±1.68**	48.99±1.83	**56.62±0.92**

datasets are quite difficult for their heterogeneous nature, both in terms of texture and color. Yet the color statistics might still provide useful information when combined with other descriptors.

An extension to the Color Names (CN) and Discriminative Color Descriptors (DD) has been proposed (denoted as CN^σ, DD^σ), significantly improving the recognition accuracy on all 8 tested datasets. The comparison of Color Names (CN) and Discriminative Color Descriptors (DD) descriptors brings a surprising observation: on 6 out of the 8 texture datasets, Color Names outperform even the higher-dimensional Discriminative Color Descriptors DD25, although the opposite may be expected from the findings on different tasks [15]. The improved CN^σ outperforms other "pure-color" descriptors on 5 out of 8 datasets, the best results on the remaining 3 datasets are achieved by the improved $DD50^\sigma$ descriptor.

Combining a state-of-the-art "pure-texture" classifier [26] with the "pure-color" classifier of CN^σ leads to an improvement on all 8 tested datasets. The weights of the classifiers in the combination should be set according to the classifiers performance. Note that by combining the classifiers a 100% accuracy was achieved on the KTH-TIPS. Significant improvements are also achieved on the AniTex and VehApp databases, where [26] performs rather poorly.

The state-of-the-art "pure-texture" and "pure-color" classifiers and their combinations obtain excellent results on simpler texture-recognition problems. They are outperformed by the recent FV-CNN model [5] in the more difficult tasks. Yet the low computational complexity of some "pure-texture" and "pure-color" descriptors is beneficial and their performance may be still interesting for future works, e.g. when used in a cascade classification scheme and followed by FV-CNN in case of ambiguity.

Acknowledgements

Milan Šulc was supported by CTU student grant SGS15/155/OHK3/2T/13, Jiří Matas by The Czech Science Foundation Project GACR P103/12/G084.

References

[1] G. J. Burghouts and J.-M. Geusebroek. Material-specific adaptation of color invariant features. *Pattern Recognition Letters*, 30(3):306–313, 2009. 4

[2] B. Caputo, E. Hayman, and P. Mallikarjuna. Class-specific material categorisation. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1597–1604. IEEE, 2005. 3

[3] C.-h. Chen, L.-F. Pau, and P. S.-p. Wang. *Handbook of pattern recognition and computer vision*. World Scientific, 2010. 1, 2

[4] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 3606–3613. IEEE, 2014. 2, 6

[5] M. Cimpoi, S. Maji, I. Kokkinos, and A. Vedaldi. Deep filter banks for texture recognition, description, and segmentation. *arXiv preprint arXiv:1507.02620*, 2015. 1, 2, 5, 6, 7

[6] M. Cimpoi, S. Maji, and A. Vedaldi. Deep filter banks for texture recognition and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3828–3836, 2015. 1, 2

[7] R. T. Clemen and R. L. Winkler. Combining probability distributions from experts in risk analysis. *Risk analysis*, 19(2):187–203, 1999. 6

[8] K. J. Dana, B. Van Ginneken, S. K. Nayar, and J. J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Transactions on Graphics (TOG)*, 18(1):1–34, 1999. 3

[9] M. Fritz, E. Hayman, B. Caputo, and J.-O. Eklundh. The kth-tips database, 2004. 3

[10] T. Gevers, A. Gijsenij, J. Van de Weijer, and J.-M. Geusebroek. *Color in computer vision: fundamentals and applications*, volume 23. John Wiley & Sons, 2012. 1

[11] E. Hayman, B. Caputo, M. Fritz, and J.-O. Eklundh. On the significance of real-world conditions for material classification. In *Computer Vision–ECCV 2004*, pages 253–266. Springer, 2004. 3

[12] F. S. Khan, R. M. Anwer, J. van de Weijer, A. D. Bagdanov, M. Vanrell, and A. M. Lopez. Color attributes for object detection. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3306–3313. IEEE, 2012. 2

[13] F. S. Khan, R. M. Anwer, J. van de Weijer, M. Felsberg, and J. Laaksonen. Compact color–texture description for texture classification. *Pattern Recognition Letters*, 51:16–22, 2015. 1, 2

[14] F. S. Khan, J. Van de Weijer, and M. Vanrell. Modulating shape features by color attention for object recognition. *International Journal of Computer Vision*, 98(1):49–64, 2012. 2

[15] R. Khan, J. Van de Weijer, F. Shahbaz Khan, D. Muselet, C. Ducottet, and C. Barat. Discriminative color descriptors. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2866–2873. IEEE, 2013. 1, 2, 7

[16] H.-T. Lin, C.-J. Lin, and R. C. Weng. A note on platts probabilistic outputs for support vector machines. *Machine learning*, 68(3), 2007. 5

[17] P. Mallikarjuna, M. Fritz, A. Targhi, E. Hayman, B. Caputo, and J. Eklundh. The kth-tips and kth-tips2 databases. `http://www.nada.kth.se/cvap/databases/kth-tips`, 2006. 3

[18] J. Mao, J. Zhu, and A. L. Yuille. An active patch model for real world texture and appearance classification. In *Computer Vision–ECCV 2014*, pages 140–155. Springer, 2014. 2, 4, 5, 6

[19] M. Mirmehdi, X. Xie, and J. Suri. *Handbook of texture analysis*. Imperial College Press, 2009. 1, 2

[20] M. Pietikäinen. Texture recognition. *Computer Vision: A Reference Guide*, pages 789–793, 2014. 1, 2

[21] J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3), 1999. 5

[22] X. Qi, R. Xiao, C.-G. Li, Y. Qiao, J. Guo, and X. Tang. Pairwise rotation invariant co-occurrence local binary pattern. *PAMI*, 36(11):2199–2213, 2014. 2, 6

[23] L. Sharan, R. Rosenholtz, and E. Adelson. Material perception: What can you see in a brief glance? *Journal of Vision*, 9(8):784–784, 2009. 4

[24] L. Sifre and S. Mallat. Rotation, scaling and deformation invariant scattering for texture discrimination. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1233–1240. IEEE, 2013. 2, 6

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[26] M. Šulc and J. Matas. Fast features invariant to rotation and scale of texture. In L. Agapito, M. M. Bronstein, and C. Rother, editors, *Computer Vision–ECCV 2014 Workshops, Part II*, volume 8926 of *LNCS*, pages 47–62, Gewerbestrasse 11, CH-6330 Cham (ZG), Switzerland, September 2015. Springer International Publishing AG. 2, 5, 6, 7

[27] K. E. Van De Sande, T. Gevers, and C. G. Snoek. Evaluating color descriptors for object and scene recognition. *PAMI*, 32(9):1582–1596, 2010. 1, 2

[28] J. Van De Weijer, C. Schmid, J. Verbeek, and D. Larlus. Learning color names for real-world applications. *Image Processing, IEEE Transactions on*, 18(7):1512–1523, 2009. 1, 2

[29] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *PAMI*, 34(3), 2011. 5

[30] J. Zhang and T. Tan. Brief review of invariant texture analysis methods. *Pattern recognition*, 35(3):735–747, 2002. 1, 2

21st *Computer Vision Winter Workshop*
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

A Novel Concept for Smart Camera Image Stitching

Majid Banaeyan* , Hanna Huber* , Walter G. Kropatsch* and Raphael Barth+
Vienna University of Technology
*Pattern Recognition and Image Processing group
{majid,hanna,krw}@prip.tuwien.ac.at
+Indiecam
raphael@indiecam.com

Abstract. *As panoramic images are widely used in many applications, efficient image stitching methods that provide visually pleasant image mosaics are needed. In this paper we discuss a novel concept for smart camera image stitching based on graph pyramids. For a multi-camera system, the images have to be aligned accordingly to create an image mosaic. Instead of calculating the corresponding transformations centrally, we aim at enabling each camera to individually calculate the transformation of the image it takes. Graph pyramids used for image segmentation provide information about the segmentation process. We analyze how this information can be used to calculate the transformations for image alignment.*

1. Introduction

Panoramic views form the basis of many applications including augmented reality applications. Producing video content with high quality seamless and artefact-free 360° of coverage is challenging per se and even more challenging if all related processing, especially seamless stitching, has to work automatically and in real-time for live productions.

A suitable approach has to solve a system conflict between omnidirectional simultaneous video capture on one hand, which cannot be done from the nodal point due to mechanical collision problems, and parallax-free stitching of panoramas without any parallax, ghosting and distortion artefacts on the other hand.

Using high on-board computing power of smart cameras and a dedicated communication network between cameras could be used to integrate the entire image processing for automatic real-time stitching into the cameras themselves, avoiding the need for further peripheral processing devices.

Common image stitching techniques take images taken from different views and align them using image registration in overlapping regions. So far, all images are collected and aligned centrally, which suffers from high computational cost. Thus, we aim at parallelizing parts of this process by developping smart cameras that are able to perform some of the image transformations themselves.

The camera systems we consider use fish-eye lenses. General camera models such as the pinhole model cannot be applied to these lenses, because they do not conform to the perspective projection due to their large field-of-view. Simple models are given for different projections of ideal fish-eye lenses. They provide a formula for the radius r which is the distance between an image point and the principal point. The principal point is the point where the optical axis intersects the image plane. In case of the equidistant projection the radius is given by

$$r = f\theta, \tag{1}$$

where f is the focal length and θ is the the incident angle of the ray from an object point. However, this formula does not reflect the behavior of real lenses. Instead, extended models are developped which take into account the high level of distortion. Parameter values are estimated using calibration, defining a final model for a particular camera [16]. In fish-eye lenses, both radial as well as tangential distortion is present. While radial distortion reduces the spatial resolution towards the periphery of an image and distorts rectilinear objects, geometric shifts are the result of tangential distortion [15].

In this paper, we define our concept of smart camera image stitching and present ideas how to realize it. We will first give an overview of related work in the fields of fish-eye lenses, image stitching and smart cameras in section 2. After declaring our functional goal in section 3, we discuss open problems that we aim to solve in order to realize it and present our ideas for possible solutions including novel approaches in section 4. Finally, in section 5, we conclude our paper.

2. Related Work

In this section we present a selection of state-of-the-art techniques in fields that are related to smart camera image stitching.

2.1. Smart Cameras

The name of smart camera goes back to the middle of 1970s [29] when Ron Schneidermann applied it in developing systems for controlling the shutter. Then in 1981 the optical mouse was invented by Richard Lyon [24, 25] which was the first realized smart camera including an imaging device and embedded processing unit as a compact system. ”Smart camera is a label which refers to cameras that have the ability to not only take pictures but also more importantly make sense of what is happening in the image.” [4, Chapter 2, page 21] Smart cameras employ various concepts of computer vision and machine vision which can extract useful information from images resulting in special decisions based on that information. Smart cameras can be classified into three main categories including integrated, compact-system and distributed smart cameras [4, chapter 2]. Integrated smart cameras can be further subdivided into three types including single-chip [2, 11], embedded [20] and stand-alone smart cameras. Distributed smart cameras involve some sort of networking and have recently attracted significant interest in academic and industries fields [28]. Indeed, some problems such as depth information in foreground detection and occlusion are difficult to be solved by single smart cameras. In this case, using multiple cameras with a powerful computing platform is an advantage. However, we encounter some physical limitations of the acquisition hardware. Although current professional cameras capture images at a horizontal resolution of about 4k to 5k [27], they are insufficient for large scales and wide-angle viewpoints. Additionally, there are some noticeable distortions caused by wide-angle optics such as distortion in the border regions of images in fish-eye lenses which result in additional loss in image resolution.

Smart camera networks have a wide range of applications in various areas including surveillance systems, security monitoring, traffic control and telemedicine [1]. For instance, Kawamura et. al [17] proposed a reliable surveillance system for railway stations. Their system tracks suspicious behavior by applying multiple camera fields of view. Smart sensors communicate with each other over a wireless mesh network. Moreover, as an application for the airport, Shirmohammadi et. al [31] introduced a decentralized target tracking scheme. Smart camera nodes automatically identify neighboring sensors with overlapping fields and produce a communication graph which reflects how the nodes will interact to fuse measurements in the network.

2.2. Multi-View Setups and Image Stitching

Moreover, numerous publications deal with panoramic images. For image registration, feature-based methods which use distinct image points are generally favored over area-based techniques which compare images window by window [39]. Lowe et al. [6, 21, 7] introduced scale-invariant feature points (SIFT) which have been widely used since. They use a 128-dimensional feature vector. Ke and Sukthankar [18] adopted this approach, but reduced the dimension of the descriptor to 36. Alternatively, Bay et al. [3] presented a faster method based on Haar wavelets using speeded up robust features (SURF).

All these features work well with standard perspective projection since they are invariant to affine transformations and provide a sufficient number of corresponding points to recover the parameters of the homography. Multi-view images taken from cameras at different positions lead to parallax errors. These errors cannot be fully eliminated. Still, these effects can be reduced. Global image transformations that are calculated by fitting a homography to matched feature points cannot handle parallax well. Zhang and Liu [38] address this problem by combining the transformation using a homography with local content-preserving warping. The homography is no longer chosen as the best fir for all feature point pairs, but considers only neighboring feature points. Additionally, they use a tolerant fitting threshold. Perazzi et al. [27] describe an algorithm for generating videos from unstructured camera arrays. They apply the basic concept of local warping to remove the parallax and define a new error measure with increased sensitivity to stitching artifacts. Their method tries to smooth out the blurring, ghosting and some other distortions caused usually when videos which feed from unstructured camera arrays are combined to create a single panoramic video. Deen et al. [9] create image mosaics for scientific purposes. Thus, they focus on correct rather than visually pleasant results. Parallax errors are reduced by performing pointing correction.

Existing tools for panoramic image stitching as well as camera calibration include Hugin [1] and PTGui [2], which are both based on Panorama Tools [3] by Dersch.

German et al.[14] investigate the application of different map projections to panoramic images including projections of fish-eye lens images. Multi-view setups are addressed by Sturm et al.[33] who develop a multi-view geometry model for central and non-central cameras based on structure-from-motion and by Luo et al.[23] who focus on saliency detection in multi-camera setups.

2.3. Fish-Eye Lenses

Schwalbe [30] develops a geometric model for fish-eye lens cameras based on the approximately linear relation between the incident angle of the ray from an object point and the distance from the corresponding image point to the principal point. Distortion is accounted for by using conventional distortion polynomials. Alternatively, Kannala and Brand [16] present a flexible camera model which is applicable for fish-eye as well as narrow-angle lenses. They use a polynomial imaging function as well as two additional terms for radial and tangential distortion, respectively. The final camera model includes 23 parameters. It provides both a forward as well as a backward model. Moreover, Luhmann et al. [22] deal with the correction of chromatic aberration in fish-eye images. Standard distortion correction methods use odd polynomial models as used by Mallon and Whelan [26]. These models describe the distorted radius r_d as a polynomial function of the undistorted radius r_u, using only odd terms. For high level distortion, however, fewer terms are needed with the division model [10]. Based on this approach, Aleman-Flores et al. [12] formulate a one-parameter model.

In order to determine lens parameters, various calibration procedures [37, 19, 32, 36, 34] have been developped. In many cases, they extract features such as lines or corners from the image of a calibration pattern for which the world coordinates are known [15]. A self-calibration method based on circle-fitting which does not require information about the objects' world coordinates is presented by Bräuer-Burchardt and Voss [5]. However, the distortion of an image needs to exactly fit the chosen distortion model. Aleman-Flores et al. [12] determine the distortion parameter automatically by introducing it into Hough space and detecting distorted lines.

3. Our Goal: a 360° Image Mosaic

We consider a multi-camera system of small high-quality cameras, in order to create a 360° image mosaic. The system consists of six fish-eye lens cameras. At this point we use the *indieGS2K* model produced by Indiecam[4]. Two adjoining cameras share an overlapping region, respectively. Position and optical parameters can be chosen arbitrarily, but will be fixed for a specific system. Each camera creates an image using fish-eye projection. Additionally, it holds the information about the other cameras' settings. In the end, an image mosaic using equirect-angular projection is created. This means that the horizon is a straight line in the middle of the image and vertical lines in real world are vertical lines in the image [14]. Before the actual stitching can be performed, the respective images have to be transformed accordingly.

Eventually, our goal is to develop the respective coordinate transformation model. For any two images I_j and I_{j+1} from the six cameras with overlapping view, a function $F_j : I_j \to I_{j+1}$ has to be found such that $F(p_j) = p_{j+1}$ for all corresponding pixels (p_j, p_{j+1}) in the overlapping region with $p_j \in I_j$, $p_{j+1} \in I_{j+1}$. The resulting algorithm should take the image of one camera as well as the settings (position, optics) of the other as input.The output will be the accordingly transformed image.

[1]http://hugin.sourceforge.net/
[2]https://www.ptgui.com/
[3]http://panotools.sourceforge.net/

[4]www.indiecam.com

4. A Novel Concept for Image Alignment

At this point, the following problems have to be solved in order to determine the image transformation:

1. Calibrate the fish-eye lens and determine the distortion.
2. Calculate the transformation from the fish-eye projection to the equirectangular projection.
3. Perform a geometrical classification of possible setups. Considering two cameras C_1 and C_2, calculate critical points and distances in order to distinguish between the following classes:
    - region in which points can only be seen by C_1
    - region in which points can only be seen by C_2
    - closer part of the overlapping region with visible parallax errors
    - part of the overlapping region with negligible parallax errors
4. Calculate the coordinate transformation

In order to solve these problems described in the previous section, we consider the following approaches.

4.1. Lens Calibration and Image Alignment using Graph Pyramids

While traditional lens and distortion models have been studied extensively, we follow a different approach. Our goal is to extract the distortion information using graph pyramids.

4.1.1 Overview

Traditionally, lens calibration is based on a geometric model depending on parameters. The respective parameter values are determined during the calibration procedure. This is a characteristic that previous lens calibration methods have in common, even though different models and procedures have been developped. By establishing a model for which the parameters are specified, these methods already make fundamental assumptions about the structure of the distortion. On the contrary, we propose a calibration method that determines the distortion including its structure. In a graph pyramid as used for structurally consistent image segmentation (SCIS) [8], the information about the segmentation process is stored. As this process is performed based on the structure of the underlying image, it also contains information about the distortion. A target coordinate system is defined by the continuous curves of a checkerboard pattern which follow the isolines of the coordinate system. By applying the segmentation to this pattern and storing the details of the segmentation process, the distortion information of the coordinate system is retrieved.

4.1.2 Features of the SCIS Algorithm

The SCIS algorithm segments an image based on Local Binary Patterns and the Combinatorial Pyramid. It works on the local structure of the image and preserves structural correctness [8, Chapter 4, page 39] and topology of an image. For this purpose, five topological classes based on Local Binary Patterns of regions are applied which by combination with the dual graph are able to remove redundant structural information. As a result, by using this approach the image graph will be simplified and connected regions will be merged without introducing structural errors [8].

The SCIS algorithm performs image segmentation using a graph-based image representation. It provides the image at any level of segmentation as well as the information about the segmentation process up to that level. The latter contains information about the distortion structure.

Initially, each pixel corresponds to a vertex and each edge to a neighborhood relation in the graph, which represents the base level of the combinatorial pyramid. Subsequently, pixels are merged to regions which are in turn merged to larger regions based on their intensity values. On higher levels, each vertex corresponds to an image region. Merging corresponds to edge contraction and removal. The SCIS algorithm creates the entire pyramid as well as the contraction history. The latter is represented by the *contraction kernels*. Thus, it is able to reconstruct the segmented image at any level. An example of a combinatorial pyramid is shown in Figure 1.

An evaluation study of stereo matching by Joanneum Research [13] shows that the SCIS algorithm achieves the highest matching quality compared to different compression methods.

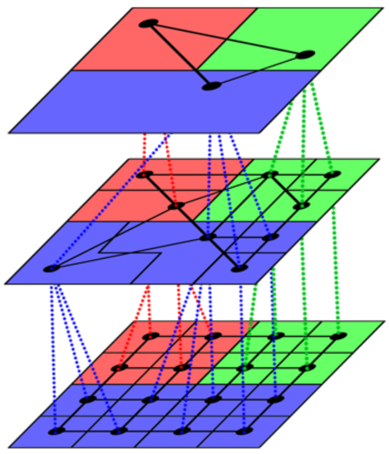


Figure 1. Example of a Combinatorial Pyramid. Image taken from [8]

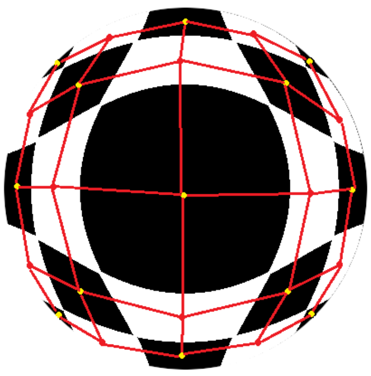


Figure 2. Distorted checkerboard pattern with corresponding primal graph at the top of the pyramid. Each vertex (yellow) corresponds to a patch. Vertices of the adjacent patches are connected by an edge (red).

4.1.3 Calibration Procedure

The canonical representation of the combinatorial pyramid stores it as a single array. The elements in this array are half-edges, called *darts*. They are ordered according to the contraction history. In order to extract information about the distortion from the combinatorial pyramid, we consider the image of a checkerboard pattern, where each patch is assigned an absolute coordinate. At the top level of the pyramid, each vertex corresponds to a single patch (see Figure 2).

As a result we get the contraction history. The top level delivers a single vertex for every patch of the checkerboard with its adjacency. All contracted edges of a patch form a spanning tree of the corresponding region in the primal graph. We do not know anything about the contraction kernels inside the homogeneous regions.

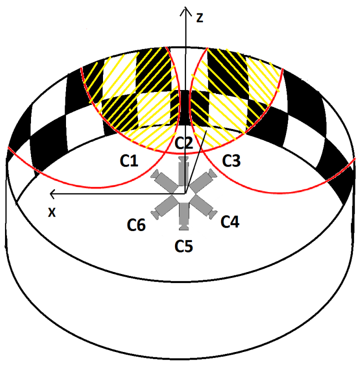


Figure 3. Multi-camera calibration setup for six cameras $C1$ - $C6$.

Since they have all the same value it cannot be said which edges are contracted or which are removed. For making the process more precise we can consider two solutions. One is to apply geometry of target coordinates and perform linear interpolation. However, this approach has the drawback that we do not know the size of the distorted patch, which is particularly problematic in our case where we expect severe deformation. The second approach is to shift the checkerboard pattern and create a new image from a different viewpoint. By iteratively applying this process, the regions inside the patches will be refined. For instance, we can take M captures with different offsets. Next, the idea is to freeze only the boundaries of which we are sure that they are precisely delineated. Indeed, by taking two different positions (randomly) and overlapping with the two contraction kernels, both boundaries should be preserved. Therefore, the random space of patches will be smaller and smaller as the process is used more and more.

There are two ways for applying this strategy. On the one hand, it can be performed sequentially by freezing the contraction kernels corresponding to the boundaries from the previous iteration. On the other hand, it can be performed randomly. Given the contraction kernels at every point and knowing the position of a boundary, we can integrate the contraction kernels using high weights at boundaries and low weights in between. For homogeneous regions, the contraction kernels provided by the shifting approach will converge towards the proper kernel.

With the contraction kernels provided, the information about the distortion is stored implicitly, allowing us to apply it to any new image. Conveniently, the canonical representation stores this in-

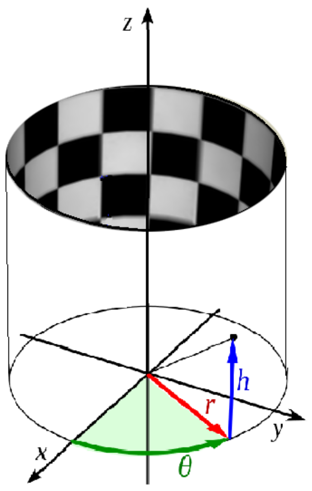


Figure 4. Calibration pattern using a cylindrical target coordinate system with radius r, azimuthal angle θ and height h.

formation in an ordered array. Thus, the calibrated kernels which have to be applied to get to a particular level of the pyramid can be re-used.

The calibration setup for a multi-camera system is illustrated in Figure 3.

4.1.4 Advantages of Calibration using Graph Pyramids

Apart from the fact that the graph-based approach does not make any assumptions about the structure of the distortion, it yields other advantages compared to previous calibration methods. Accuracy can be increased simply and reached to the resolution of original images by increasing the number of shifts. Additionally, we do not need a global model of the geometric projection for calibration, which is needed for many estimation methods of the parameters. Finally, our method does not depend on a particular coordinate system. Instead, any target coordinate system can be chosen. It is defined by the checkerboard pattern where the continuous curves correspond to the the isolines of a target coordinate system. Thus, various geometries can be used for this approach such as cylindrical (see Figure 4) or spherical (see Figure 5) coordinate systems. In particular, the coordinate system of the final mosaic can be chosen as target coordinate system. In this case, the transformation provided by the calibration method does not only consider lens distortion, but also includes remapping to equirectangular projection as well as image alignment, and this simultaneously for all six cameras.

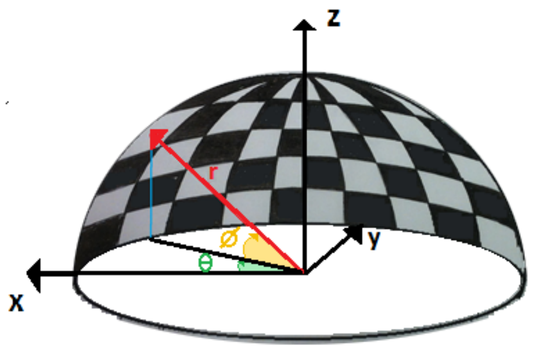


Figure 5. Calibration pattern using a spherical target coordinate system with radius r, azimuthal angle θ and elevation angle ϕ.

4.2. Projection remapping

The remapping from fish-eye to equirectangular projection can also be handled by the graph-based calibration method presented in the previous section. For comparison, it can be addressed individually following German et al. [14]. Information about the camera's roll, which is the rotation angle about the optical axis, and pitch, which is the elevation angle from the horizontal axis, allows the remapping from a fish-eye to an equirectangular projection. Roll and pitch can be determined manually or by using horizontal or vertical control lines.

4.3. Setup Classification

The classification of the setup with regard to parallax errors can be performed using partial edge contours as used by Wang et al. [35]. The edge contour of an obstacle is mapped from one image to the other. The parallax is then calculated as the transverse distance between corresponding edge contour pixels.

4.4. Image Transformation

Similar to the projection remapping, the image tranformation used for image alignment can be determined by the graph-based approach. For comparison, the calibration of the multi-camera system can be performed using feature extraction and matching. For this purpose, SIFT features [21] will be used. In order to reduce parallax errors, the image transformation will be calculated following the parallax-tolerant approach used by Zhang and Liu [38].

5. Conclusion

We presented a novel concept for the smart camera image stitching. It aims at reducing the cost of the stitching process by enabling each camera of a multi-camera system to align the image that takes individually. Lens calibration can be performed using graph pyramids, which yields several advantages compared to traditional lens calibration methods. Additionally, the same method can be used to directly determine the image transformation required for image alignment. Currently, the work is in progress, but in near future we are planning to experimentally prove the applicability of the proposed ideas.

References

[1] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury. A survey on wireless multimedia sensor networks. *Computer Networks*, 51:921–960, 2007. 2

[2] L. Albani, P. Chiesa, D. Covi, G. Pedegani, A. Sartori, and M.Vatteroni. VISoc: A smart camera SoC. In *Proceedings of the 28th European Solid-State Circuits Conference*, pages 367–370, 2002. 2

[3] H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. In *In ECCV*, pages 404–417, 2006. 2

[4] A. N. Belbachir, editor. *Smart Cameras*. Springer, 2010. 2

[5] C. Bräuer-Burchardt and K. Voss. A new algorithm to correct fish-eye- and strong wide-angle-lens-distortion from single images. *Proceedings 2001 International Conference on Image Processing 2001, Vol.1, pp.225-228*, 2001. 3

[6] M. Brown and D. G. Lowe. Recognising panoramas. In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, pages 1218–1225, Washington, DC, USA, 2003. IEEE Computer Society. 2

[7] M. Brown and D. G. Lowe. Automatic panoramic image stitching using invariant features. *Int. J. Comput. Vision*, 74(1):59–73, Aug. 2007. 2

[8] M. Cerman. Structurally correct image segmentation using local binary patterns and the combinatorial pyramid, 2015. Wien, Techn. Univ., Dipl.-Arb., 2015, Technical Report 133. 4

[9] B. Deen. In-Situ Mosaic Production at JPL/MIPL. Pasadena, CA : Jet Propulsion Laboratory, National Aeronautics and Space Administration, 2012. Planetary Data: A Workshop for Users and Software Developers 2012, JPL TRS 1992+. 3

[10] A. Fitzgibbon. Simultaneous linear estimation of multiple view geometry and lens distortion. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2001, Vol.1, pp.I-I.* 3

[11] B. Flinchbaugh. Smart cameras systems technology roadmap. In B. Kisacanin, V. Pavlovic, and T. Huang, editors, *Real-Time Vision for Human-Computer Interaction*, pages 285–297, 2005. 2

[12] M. A. Flores, L. Á. León, L. G. Déniz, and D. E. S. Cedrés. Automatic Lens Distortion Correction Using One-Parameter Division Models. *IPOL Image Processing OnLine (Special Issue on Lens Distortion Models)*, 4:327–343, 2014. 3

[13] B. Froehlich and M. P. Caballo-Perucha. Evaluation of image compression algorithms version 1.0, issue D1, 2015. Joanneum Research. 4

[14] D. M. German, P. d'Angelo, M. Gross, and B. Postle. New Methods to Project Panoramas for Practical and Aesthetic Purposes. In D. W. Cunningham, G. Meyer, and L. Neumann, editors, *Computational Aesthetics in Graphics, Visualization, and Imaging*. The Eurographics Association, 2007. 3, 6

[15] C. Hughes, M. Glavin, E. Jones, and P. Denny. Review of geometric distortion compensation in fish-eye cameras. In *IET Irish Signals and Systems Conference, 208. (ISSC 2008)*, 2008. 1, 3

[16] J. Kannala and S. S. Brandt. A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE TRANS. PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 28:1335–1340, 2006. 1, 3

[17] A. Kawamura, Y. Yoshimitsu, K. Kajitani, T. Naito, K. Fujimura, and S. Kamijo. Smart camera network system for use in railway stations. In *SMC*, pages 85–90. IEEE, 2011. 2

[18] Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR'04, pages 506–513, Washington, DC, USA, 2004. IEEE Computer Society. 2

[19] M. Kedzierski and A. Fryskowska. Precise method of fisheye lens calibration. In *Proceedings of the ISPRS-Congress*, pages 765–768. International Society for Photogrammetry and Remote Sensing, 2008. 3

[20] B. Kisacanin, S. Bhattacharyya, and S. Chai. *Embedded Computer Vision*. Springer, 2007. 2

[21] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004. 2, 6

[22] T. Luhmann, H. Hastedt, and W. Tecklenburg. Modelling of chromatic aberration for high precision photogrammetry. *Remote Sensing and Spatial Information Sciences*, 36 (Part 5):173–178. 3

[23] Y. Luo, M. Jiang, Y. Wong, and Q. Zhao. Multi-camera saliency. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(10):2057–2070, 2015. 3

[24] R. Lyon. The optical mouse, and architectural methodology for smart digital sensors. In H.T.Kung, B.Sproull, and G.Steele, editors, *Computer Science Press. Invited Paper, CMU Conference on VLSI structures and Computations*, 1981. 2

[25] R. Lyon. Apparatus for controlling movement of a curser in computer display system, 1983. European Patent. 2

[26] J. Mallon and P. Whelan. Precise radial un-distortion of images. *Proceedings of the 17th International Conference on Pattern Recognition, 2004, Vol.1, pp.18-21*. 3

[27] F. Perazzi, A. Sorkine-Hornung, H. Zimmer, P. Kaufmann, O. Wang, S. Watson, and M. Gross. Panoramic video from unstructured camera arrays. In *Proc. Eurographics 2015*, volume 34, 2015. 2, 3

[28] B. Rinner and W. Wolf. An introduction to distributed smart cameras. In *Proceedings of the IEEE*, volume 96, pages 1565–1575, 2008. 2

[29] R. Schneidermann. Smart cameras clicking with electronic functions. *Electronics*, 48:74–81, 1975. 2

[30] E. Schwalbe. Geometric modelling and calibration of fisheye lens camera systems. In *Proceedings 2nd Panoramic Photogrammetry Workshop, Int. Archives of Photogrammetry and Remote Sensing*, pages 5–8, 2005. 3

[31] B. Shirmohammadi and C. J. Taylor. Distributed target tracking using self localizing smart camera networks. In *Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras*, pages 17–24, New York, NY, USA, 2010. 2

[32] P. Srestasathiern and N. Soontranon. A novel camera calibration method for fish-eye lenses using line features. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pages 327–332, Aug. 2014. 3

[33] P. Sturm, S. Ramalingam, and S. Lodha. On calibration, structure-from-motion and multi-view geometry for general camera models. In R. Reulke and U. Knauer, editors, *2nd ISPRS Panoramic Photogrammetry Workshop,*, Berlin, Allemagne, 2005. ISPRS. Published in the Int. Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XXXVI-5/W8. 3

[34] S. Urban, J. Leitloff, and S. Hinz. Improved wide-angle, fisheye and omnidirectional camera calibration. *{ISPRS} Journal of Photogrammetry and Remote Sensing*, 108:72 – 79, 2015. 3

[35] X.-H. Wang, W.-P. Fu, and W. Chen. Detection of obstacle based on nocular vision. *2010 International Conference on Intelligent Computation Technology and Automation, May 2010, Vol.2, pp.71-74*. 6

[36] Z. Wang, H. Liang, X. W. andYipeng Zhao, B. Cai, C. Tao, Z. Zhang, Y. Wang, S. Li, F. Huang, S. Fu, and F. Zhang. A practical distortion correcting method from fisheye image to perspective projection image. In *Information and Automation, 2015 IEEE International Conference on*, pages 1178 – 1183, 2015. 3

[37] X. Ying, Z. Hu, and H. Zha. Fisheye lenses calibration using straight-line spherical perspective projection constraint. In P. J. Narayanan, S. K. Nayar, and H.-Y. Shum, editors, *ACCV (2)*, volume 3852 of *Lecture Notes in Computer Science*, pages 61–70. Springer, 2006. 3

[38] F. Zhang and F. Liu. Parallax-tolerant image stitching. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '14, pages 3262–3269, Washington, DC, USA, 2014. IEEE Computer Society. 2, 6

[39] B. Zitov and J. Flusser. Image registration methods: a survey. *Image and Vision Computing, 2003, Vol.21(11), pp.977-1000*. 2

21st *Computer Vision Winter Workshop*
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

A concept for shape representation with linked local coordinate systems

Manuela Kaindl and Walter G. Kropatsch
Pattern Recognition and Image Processing Group, Vienna University of Technology, Austria
http://www.prip.tuwien.ac.at

Abstract.

This paper discusses a concept for the representation of n-dimensional shapes by means of a model, based on linked local coordinate systems. Through application of the medial axis transform (MAT) and decomposition of the resulting medial axis (MA), articulated, as well as non-rigid abstract n-dimensional bodies can be described by defining corresponding local coordinate systems for each element. This should allow a distinct and invariant representation of every point of the shape, which can be used for complex composite transformations of the object in the context of robotic manipulation.

1. Introduction

For the automatic manipulation of objects and reasoning considering their attributes, a powerful model is needed. Articulated objects, like the human body, or deformable objects, like a piece of clothing, demand a model that is able to represent complex intrinsic transformations. These classes of objects can be represented by defining coordinate systems for each segment, so every point of the object is distinctly determined by a set of coordinates. One application, for both classes of objects mentioned, is automated dressing-assistance for a person. Linked local coordinate systems should allow the description of every point of the shape, so it can be exactly defined where a robotic arm needs to grasp a glove and how it needs to place it for the person to slip in comfortably, considering the person's range of motion.

A coordinate system is specified by its origin, determining the location, and a set of basis vectors, defining the orientation and scale of the element. It makes the description of an element invariant to changes. In the case of articulated movements, the specific coordinates of the parts do not need to be changed. The intrinsic movement of an articulated object's element can be described as a transformation between two linked coordinate systems. Swinging of the arm can be characterised as a transformation of the arms coordinate system in respect to the linked coordinate system of the torso for movement of the shoulder and transformation of the distal part's coordinate system in respect to the system of the upper arm for movement of the elbow. The coordinate system of the hand in respect to the system of the forearm does not change in that case (Fig. 1). In case of a smooth deformation, local interpolation between the transition of the elements may be needed.

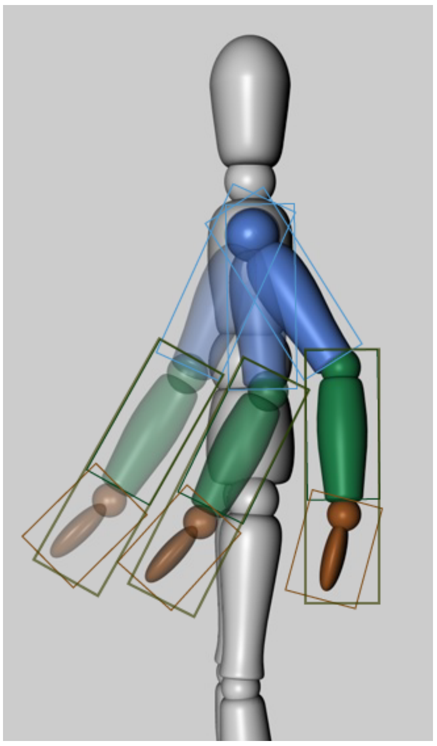


Figure 1. Linked local coordinate systems of a swinging arm. Frames indicating the area of a coordinate system. Forearm and hand do not move in respect to each other while the linked system of the distal part (parent of hand and forearm) of the arm changes in respect to the system of the upper arm.

The intrinsic movement of a non-rigid object is supported by the model's invariance to deformation originating from the axial representation. The object's axial representation provides the linked local coordinate systems. In 3D space, axial representations can be produced by sweeping spheres along the axis [16]. For 2D objects, geometric primitives, like circles or line segments, can be used as generators [14, 20]. The linked local coordinate systems are based on the resulting medial axis of the object using an end point as the origin and a branch of the medial axis as a basis vector of the coordinate system.

Several problems need to be addressed to provide a stable and invariant model that can represent an object and leads to reliable reasoning:

1. Noise
    - Noise inside the shape creating holes.
    - Noise along the boundary creating spurious branches.
2. Decomposition
    - Multiple affiliation of points in branching areas.
3. Preservation of structure
    - Ordering of axes at branching points.
4. Special shapes
    - Spheres and objects based on spheres.
    - Circular MAs.

The novelty of the method is the utilisation of linked local coordinate systems for the representation of n-dimensional objects for robotic manipulation.

The paper is organised as follows. In section 2, related work is outlined. Section 3 describes the proposed method and its open problems in detail. Section 4 concludes the paper with a discussion of the method.

2. Related Work

Most recently, research in the field of robotic dressing-assistance was done by Gao, Chang and Demiris, who utilise randomized forests for a model of the upper body [6]. Klee et al. used a skeleton tracker for a robotic dressing-application.

Handling and predicting articulated objects or non-rigid objects demands a complex model that can represent the vast amount of different possible appearances of an object. Several projects have already been dedicated to that issue. Li, Chen and Allen [11] used meshes of deformable objects to simulate the movement and its results to identify grasping points of garments. With a system of dictionary learning via spatial pyramid matching and sparse coding, a robotic grasper is enabled to grasp, flatten and fold garments. Felzenszwalb, Mc Allester and Ramanan [5] published an algorithm for the recognition of deformable objects in images by means of a discriminatively trained, multiscale, deformable part model in 2008. Godec, Roth and Bischof [7] described hough-based tracking of non-rigid objects in 2013. Their approach utilises the generalised Hough-transform to handle articulated and non-rigid objects. Pouch et al. [13] resort to the MAT to segment the deformable aortic valve apparatus in 3D echocardiographic images.

To provide a stable basis for the concept, a MAT algorithm must be used which can provide a geometrically accurate and compact MA. In recent years, several groups have been dedicated to improve prior efforts in that field. Li et al. published an approach for MAT by Quadratic Error Minimization to compute a stable and compact MA [10] The groups of Zhu et al. published a paper on the constructive generation of the medial axis for solid models [18] and also an approach for calculation of the medial axis of a CAD model by parallel computation [19]. Aichholzer, Aigner, Aurenhammer and Juettler showed a technique for the MAT by means of a polyhedral unit ball instead of the standard Euclidean unit ball [2]

3. Method

MAT has the property of producing a MA of one dimension less than the object in many cases. A 3D object creates a 2D MA and a MAT of a 2D MA generates a 1D manifold (Fig. 2.a) that can be decomposed at its branching points (Fig. 2.b, Fig. 2.c). As branching points we denote locations where more than 2 branches of the MA meet. These points represent the basis of convexities of the shape. Points within the largest inscribing circle around these branching points, the branching area, have an unclear affiliation to a MA branch, which poses a problem when the connected MA branches move in respect to each other. The decomposed

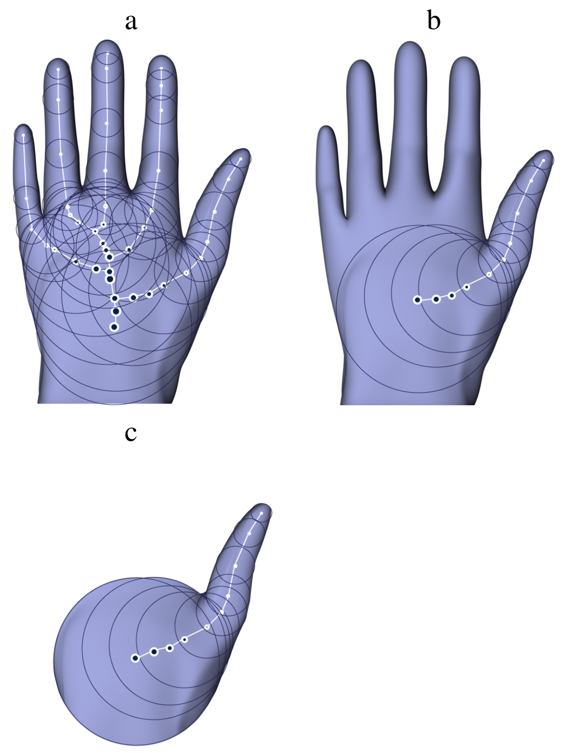


Figure. 2. a) MAT of the image of a hand. Largest inscribing circles form the MA. b) Decomposed branch of the MA. c) Area to be described in respect to this branch.

branch of the MA is straightened to form the x-axis of a new coordinate system by replacing the geodesic distances by Euclidean coordinates. The distances along the MA stay identical, while the curvature is removed (Fig. 3).

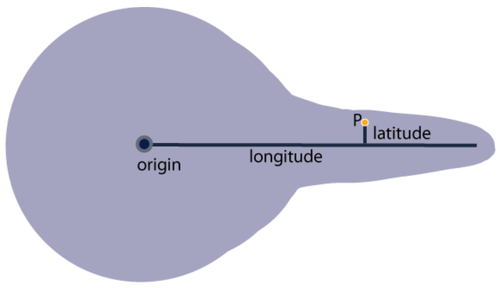


Figure 4. Coordinate system based on a MA branch. A point is defined by longitude and latitude.

This makes the representation invariant to deformation of the object, except stretching and compression, where the geodesic distance may change with movement. One end point of the axis can be chosen as the origin. All points within the silhouette of the object can be described as a tuple of longitude along the axis and latitude as the distance of the point along the normal to the axis (Fig. 4). This procedure

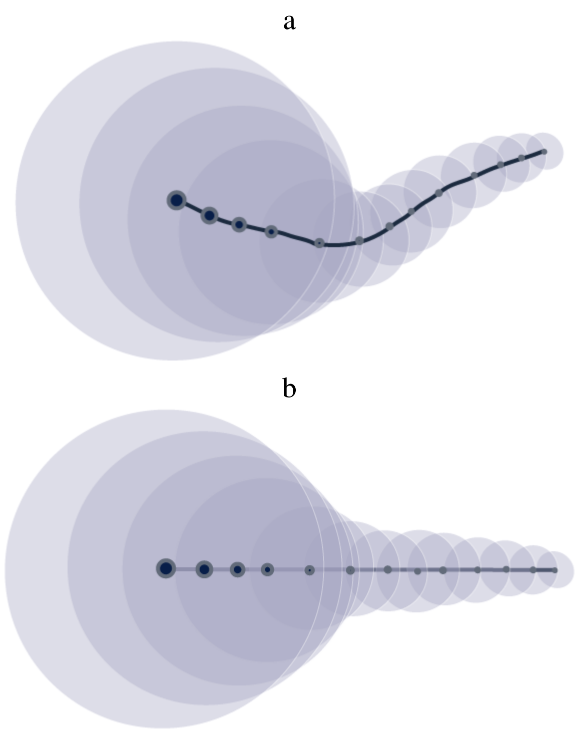


Figure. 3 a) Elongated shape with MA and its straightened representation (b). The representation is invariant to deformation.

of MAT, decomposition and straightening creates a graph with end points and branching points as nodes and axis branches as edges (Fig. 5).

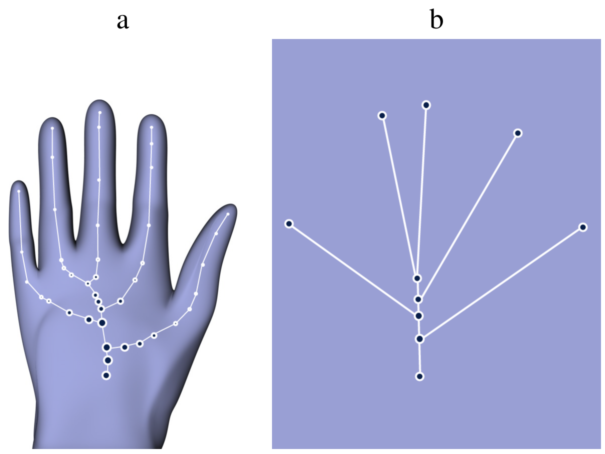


Figure. 5 a) MAT of the image of a hand. b) Graph created by straightening the MA branches of the hand.

By means of the graph, the structure of the object can be identified. The graph concept is based on the notion of cellular complexes, described by Kovalevsky [9], which states that an n-dimensional object is confined by an (n-1)-dimensional object. The

1D MA is confined by 0D points, the 2D MA is confined by 1D curves and so forth. Based on this principle of cellular complexes and the attribute of MAT to produce a MA of the objects dimensionality minus 1 in many cases, it is assumed that the proposed method holds for many n-dimensional objects by recursive application until 1-dimensionality is reached.

To communicate the principle of MA, we show how to build an abstract object from its MA. A shape can be created by sweeping a circle along a 1D Axis as can be seen in Fig. 6. The MA is synonymous

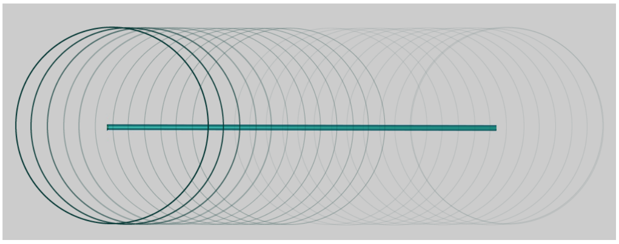


Figure 6. Circles swept along a 1D MA. Transparency indicates the sweeping movement.

with the x axis of a coordinate system we use to define all points of the shape. The radius of every circle at position x along the axis has to be stored to create the intended object. This assures the preservation of shape. Given that the circles have to touch the outline of the shape at at least 2 points at all times and no circle is completely contained in another, the silhouette of all the circles combined describes the shape that is to be produced [3]. Noise on the boundary of the object can cause spurious branches, meaning branches of the MA that do not hold valuable information about the appearance of the shape. Noise within the object may cause holes and therefore circular MAs. In Fig.7, we compose several branches to

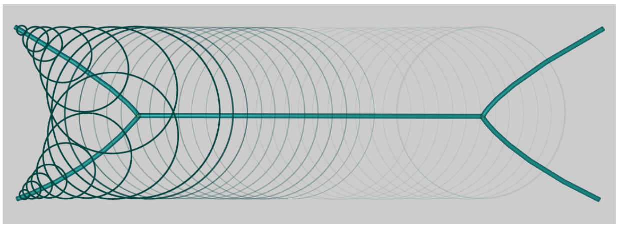


Figure 7. Circles swept along a composed 1D MA. Transparency indicates the sweeping movement.

one MA. The constellation of branches determines the structure of the object. The structure can have different constraints in its movement, depending on the intrinsic mobility of the object. This topic is discussed further in chapter 3.3 Preservation of structure. When creating the 2D object, the Euclidean coordinates of the MA are replaced by the geodesic coordinates the axis shall have within the shape. Fig. 8 shows the 2D object that emerges from the composed MA. This 2D object itself can be used as the MA for a 3D object. This concept can be be continued due to the MAT's attribute to create an object with the dimensionality of the object minus 1. So its reversal leads to an object with the dimensionality of the MA plus 1.

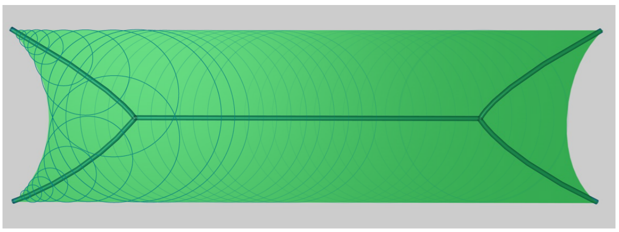


Figure 8. Covered area as 2D MA of a 3D object.

3.1. Noise

Noise on the boundary of the shape can cause spurious branches. Noise within the shape may cause holes, which can lead to circular MAs. Several projects are dedicated to the reduction of the influence of noise on the MAT. Most recently Spitzner and Gonzalez [17] published a method called Shape Peeling to improve the stability of image skeletons. Abiva and Larsson [1] proposed a method to utilise the Scale Axis Transform to prune the MA of spurious branches. Montero and Lang [12] published an algorithm for skeleton pruning by means of contour approximation and the integer MAT in 2012.

3.2. Decomposition

Decomposition is performed in branching areas to obtain less complex axes. Serino, Arcelli and Sanniti di Baja [15] recently described the decomposition of 3D objects at branching points to obtain meaningful object parts. In 2D, the branching area lies within the largest inscribing circle where 2 or more branches of the axis meet in the centre (Fig. 9). While the points of the shape lying in a circle that only belongs to one axis, are uniquely defined, points within the branching area can be described in relation to several branches of the axis (Fig. 10). If branches move in respect to each other, these points shall each be affiliated with only one branch to preserve a unique representation. While Serino, Arcelli and Sanniti di Baja [15] can already demonstrate impressive experimental results of the decomposition of the composed

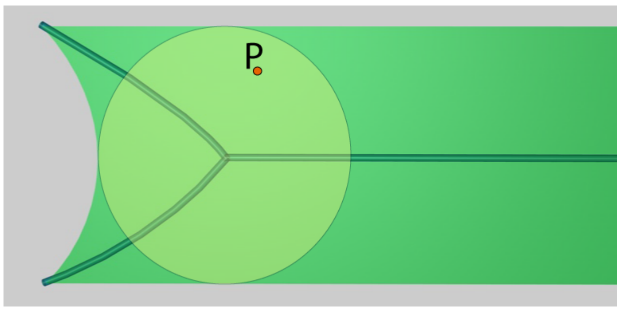


Figure 9. A point within a branching area can be described in relation to several branches of the axis.

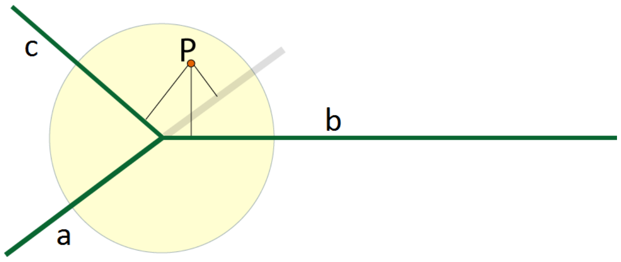


Figure 10. A point within a branching area can be described in relation to several branches of the axis. Axis a is extended across the centre, illustrating its negative domain.

1D MA of 3D objects, MAs of higher dimensions require further research.

In 3D, there can be branching points or branching curves where the branches of the MA meet as can be seen in Fig. 11. In a first idea we approach the branching area as if it is an object itself. The branching area of a curve we define by the largest inscribing sphere that is swept along the branching curve (Fig. 12).

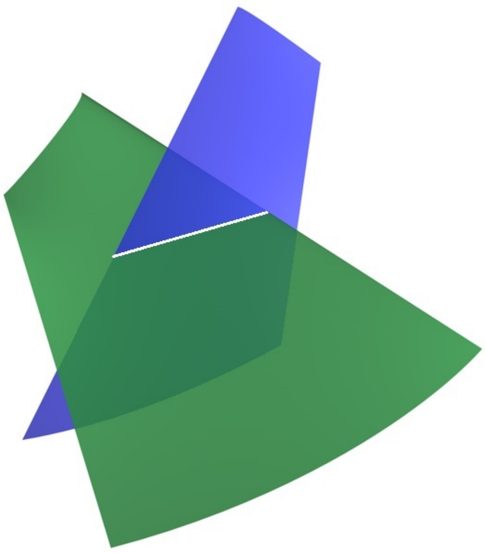


Figure 11. Two 2D MA branches of a 3D object forming a branching curve where they intersect.

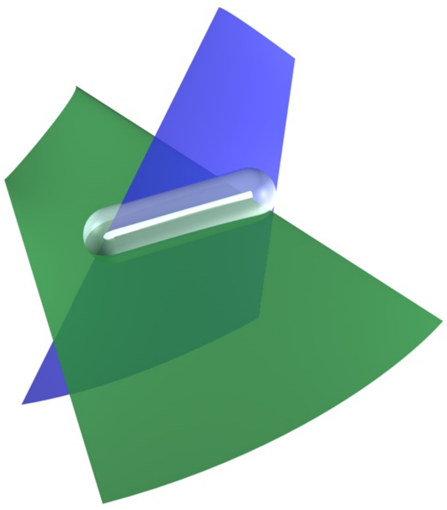


Figure 12. A sphere swept along the branching curve creating a new 3D object based on a sphere.

The branching area itself can be seen as a 3D rod-like object or as a 4D object created by sweeping a 3D sphere along an axis. This implies a leap of at least 2 dimensions to reach the 1D MA, which violates the assumption that the MAT reduces the dimensionality of an object by 1. A problem that is yet to be solved and is explained further in the chapter 3.4.1 Spheres and objects based on spheres.

A different approach is to apply the MAT recursively to every branch of the MA until 1D is reached. This way, joints will not necessarily imply a connection of the MA branches (Fig. 13) and the MA branches of an object might not intersect. If the MA breaks into several pieces, it arises the question of how the structure can be maintained. Further work on this matter is required.

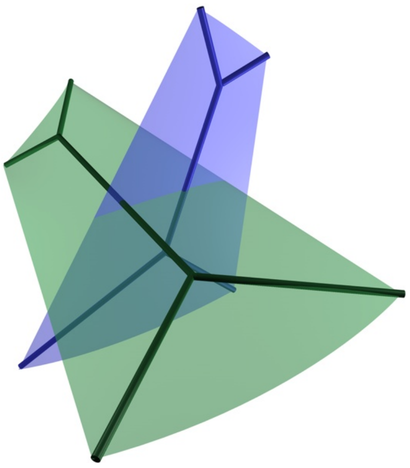


Figure 13. 1D MA branches of the 2D MA branches do not intersect.

3.3. Preservation of structure

Articulated objects with a specific range of motion require constraints at joints, so the human forearm can not rotate around the elbow, but can only flex in one direction to a certain degree. Non-rigid objects, like cloth, require different constraints since they do not have joints, but feature a certain thickness, stiffness, weight and other properties. A basic ordering has to be maintained regardless of these characteristics. As shown in Fig. 14, all MA branches might be

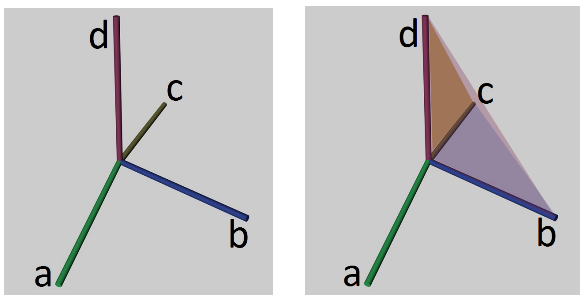


Figure 14. 3D branching point of MA branches. Branch a can move freely except across the triangles spanned by the other branches b, c and d.

able to move freely, provided they do not cross planes spanned by two different axes to sustain the objects organisation. The structure can be preserved by considering the branches of the MA as edges and the end points and branching points as nodes of a combinatorial map as described by Damiand and Lienhardt [4].

3.4. Special shapes

There are several open problems regarding special shapes in the method that require further research. Thoughts of the community on the matter are highly appreciated.

3.4.1 Spheres and objects based on spheres

The concept of MAT is mostly built on the usage of circles and spheres. If an object, or a part of it, itself is one of these primitives or based on the primitive in a higher dimension, the MAT will not create an object of its dimension minus 1, but it may create a MA with a dimensionality even lower. This violates our basic assumption that this is the case. This means that the MA can not be used to determine the location of points of the shape uniquely. One approach to solve this problem is to utilise spherical coordinate systems. Fig. 15.a shows a 3D sphere that creates a 0D MA. Fig. 15.b illustrates the sphere after application of an equator to orient the spherical coordinate system. With these systems, all points of a sphere can be distinctly determined. Objects based on spheres imply that the shape can be created by moving a sphere along a path (Fig. 15.c, Fig. 15.d). It follows, therefore, that every point of the object based on a sphere can be uniquely determined when the spherical coordinate system is moved along the MA.

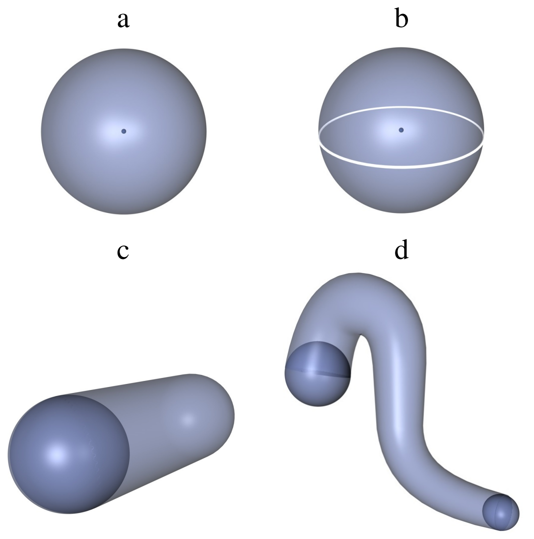


Figure 15. a) 3D sphere producing a 0D MA. b) Equator applied to a sphere to provide orientation for the spherical coordinate system. c,d) Shape described by sweeping a spherical coordinate system along a path.

3.4.2 Circular medial axes

Circular medial axes occur when an object element has genus higher than 0 (Fig. 16.a) and at concavities of the object (Fig. 16.b). If a circular MA branch is connected to 1 or more other branches of the MA, the branching points can be used to decompose the circular MA and therefore create non-circular sections that can be treated regularly. This is the case if the object features a tail. Elements with genus higher than 1 also feature connected MA branches because of the bridge between the holes whose MA branch connects the sides. This leaves an issue for objects with genus 1 and no tail (Fig. 16.a) and objects with convex elements (Fig. 16.b). The n-dimensional MA is not confined by a (n-1)-dimensional object, which violates one of the basic assumptions of this method, namely the concept of cellular complexes. If an object produces a circular MA without connected

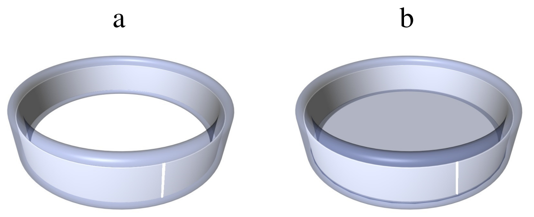


Figure 16. a) 2D circular MA within a tube-like object with an arbitrarily set reference (white). b) 2D circular MA branch as part of an object's MA with an arbitrarily set reference (white).

branches, there is no reference point that can be used as the origin of the coordinate system. A first attempt to solve this problem, based on the findings of Illetschko [8], is to place an arbitrary reference point. This point can be used as the origin of the coordinate system based on the MA. Depending on the dimensionality of the object, also a cut can be necessary. Points within the area of the new origin can then be defined in relation to both end points of the MA.

A special case is shown in Fig. 17. The torus is a shape based on a sphere, meaning that it can be described as a sphere moved along a circular path. As explained earlier, this enforces the use of a spherical coordinate system. Also the torus has a circular MA, which requires an arbitrarily set reference point.

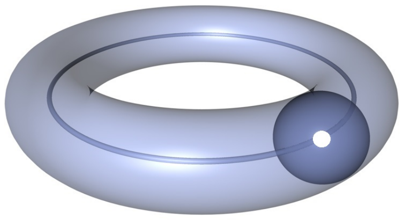


Figure 17. Special case: Torus is a shape based on a sphere and creates a circular MA. From arbitrarily set reference point on the MA (white), a spherical coordinate system is swept along the MA.

4. Conclusion

This paper proposes an novel concept for the representation of n-dimensional shapes through a model, based on linked local coordinate-systems. Through recursive application of the MAT and decomposition of the resulting MA, some n-dimensional objects can be reduced to multiple 1-dimensional sub-elements that are used as the axis for coordinate-systems. The 1D elements as edges and their end points as nodes, form a graph that represents the object. Articulated, as well as non-rigid objects can be described by defining corresponding coordinate systems of each element. This should allow complex composite transformations of the object. Intrinsic movement does not imply the transformation of point-clouds or meshes, but of linked local coordinate systems.

Further work to be done on the project is to provide a proof of concept, especially concerning the feasibility of the method for n-dimensions and resolution of the open problems described in this paper.

Acknowledgements

We would like to thank the reviewers for constructive feedback and the PRIP Club, the organization of friends and promoters of Pattern Recognition and Image Processing activities Vienna, Austria, for support.

References

[1] J. Abiva and L. J. Larsson. Towards automated filtering of the medial axis using the scale axis transform. In *Research in Shape Modeling*, pages 115–127. Springer, 2015. 4

[2] O. Aichholzer, W. Aigner, F. Aurenhammer, and B. Juettler. Exact medial axis computation for triangulated solids with respect to piecewise linear metrics. In J. Boissonnat, P. Chenin, A. Cohen, C. Gout, T. Lyche, M. Mazure, and L. Schumaker, editors, *Curves and Surfaces*, volume 6920 of *Lecture Notes in Computer Science*, pages 1–27. Springer Berlin Heidelberg, 2012. 2

[3] H. Blum. *A Transformation for extracting new descriptors of shape*. MIT Press, 1967. 4

[4] G. Damiand and P. Lienhardt. *Combinatorial Maps: Efficient Data Structures for Computer Graphics and Image Processing*, volume 129. A. K. Peters, Ltd. Natick, MA, USA, 2014. 6

[5] P. Felzenszwalb, D. Mc Allester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on. IEEE*, 2008. 2

[6] Y. Gao, H. Chang, and Y. Demiris. User modelling for personalised dressing assistance by humanoid robots. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 1840–1845, Sept 2015. 2

[7] M. Godec, P. Roth, and H. Bischof. *Hough-based tracking of non-rigid objects*, volume 117. Elsevier, 2011. 2

[8] T. Illetschko. *Minimal combinatorial maps for analyzing 3D data*. Diploma Thesis, TU Wien, 2006. 7

[9] V. A. Kovalevsky. *Finite Topology as Applied to Image Analysis*, volume 2. Academic Press, 1989. 3

[10] P. Li, B. Wang, F. Sun, X. Guo, C. Zhang, and W. Wang. Q-mat: Computing medial axis transform by quadratic error minimization. *ACM Trans. Graph.*, 35(1):8:1–8:16, December 2015. 2

[11] Y. Li, C. Chen, and P. Allen. Recognition of deformable object category and pose. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014. 2

[12] A. Montero and J. Lang. Skeleton pruning by contour approximation and the integer medial axis transform. *Computers & Graphics*, 36(5):477–487, 2012. 4

[13] A. Pouch, S. Tian, M. Takabe, H. Wang, J. Yuan, A. Cheung, B. Jackson, J. Gorman, R. Gorman, and P. Yushkevich. Segmentation of the aortic valve apparatus in 3d echocardiographic images: Deformable modeling of a branching medial structure. In *Statistical Atlases and Computational Models of the Heart - Imaging and Modelling Challenges*, volume 8896 of *Lecture Notes in Computer Science*, pages 196–203. Springer International Publishing, 2015. 2

[14] A. Rosenfeld. *Axial representations of shape*, volume 33. Academic Press Professional, 1986. 2

[15] L. Serino, C. Arcelli, and G. Sanniti di Baja. *From skeleton branches to object parts*, volume 129. Elsevier, 2014. 4

[16] E. Sherbrooke, N. Patrikalakis, and E. Brisson. *An Algorithm for the Medial Axis Transform of 3D Polyhedral Solids*, volume 2. IEEE Educational Activities Department Piscataway, 1996. 2

[17] M. Spitzner and R. Gonzalez. Shape peeling for improved image skeleton stability. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015, South Brisbane, Queensland, Australia, April 19-24, 2015*, pages 1508–1512, 2015. 4

[18] H. Zhu, Y. Liu, J. Bai, and X. Ye. Constructive generation of the medial axis for solid models. *Computer-Aided Design*, 62:98 – 111, 2015. 2

[19] H. Zhu, Y. Liu, J. Zhao, and H. Wang. Calculating the medial axis of a {CAD} model by multi-cpu based parallel computation. *Advances in Engineering Software*, 85:96 – 107, 2015. 2

[20] Y. Zhu, F. Sun, Y. Choi, B. Juettler, and W. Wang. *Computing a compact spline representation of the medial axis transform of a 2D shape*, volume 76. Elsevier, 2014. 2

21st *Computer Vision Winter Workshop*
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

A Computer Vision System for Chess Game Tracking

Can Koray
Department of Computer Engineering
Başkent University
Ankara, TURKEY
cannkorayy@gmail.com

Emre Sümer
Department of Computer Engineering
Başkent University
Ankara, TURKEY
esumer@baskent.edu.tr

Abstract. *In this paper, we present a real-time system that allows the detection of the moves of a chess game. In the proposed approach, each captured video frame, from a RGB webcam positioned over the chessboard, is processed through the following steps; the detection of the corner points of the chessboard grids, geometric rectification, chessboard position adjustment, automatic camera exposure adjustment, intensity adjustment, move detection and chessboard drawing. All steps were implemented in MATLAB programming environment without using any chess engine. The proposed approach correctly identified 162 of 164 moves in 3 games played under different illumination conditions.*

1. Introduction

There are many systems of computer vision, which require algorithms to be able to recognize different objects and scenes. Since, chess game has become an interesting issue in terms of human-computer interaction systems, a computer vision system is needed for chess playing and chessboard recognition system.

There are various published techniques related to chess-playing systems. Sokic and Ahic-Djokic [11] proposed a computer vision system for chess playing robot manipulator as a project-based learning system. The proposed algorithm detects chess moves by comparing frames captured before, during and after a move, and finds the difference between them. In a similar study, Ataş et al. [1] developed a chess playing robotic arm system composed of various modules such as main controller, image processing, machine learning, game engine and motion engine of robot arm. In their study, the top of the pieces are uniquely designed to be different from each other in order to track by the system.

On the other hand, in a study conducted by Bennet and Lasenby [2], the recognition of chessboards under deformation was carried out. Their method determined a grid structure to detected vertices of a chessboard projection. Further, the same authors developed a feature detector named 'Chess-board Extraction by Subtraction and Summation (ChESS)' to respond to chessboard vertices [3]. In a different study, a chessboard recognition system was proposed [8]. The proposed system was applied to chessboard in order to identify the name, location and the color of the pieces. Piskorec et al. [10] presented a computer vision system for chess game reconstruction. The system reconstructs a chessboard state based on video sequences obtained from two cameras.

The tracking of the chess moves can be regarded as the preliminary task before designing a robotic chess playing system. In the literature, there are several efforts that perform the chess move tracking. The studies conducted by Matuszek et al. [9], Urting and Berbers [12], Cour et al. [4] and Gonçvales et al. [7] use unique algorithms to identify the chessboard grids along with the classification of squares. These methods are not only based on corner detection but also rely on having a clean background.

In this paper, we propose a real-time chess game tracking system using a RGB webcam positioned over the chessboard. In general, the move is detected by comparing the occupancy grids based on average color information of the pieces and the squares. Before that, several pre-processing steps are employed including geometric rectification, intensity adjustment and chessboard position adjustment. The system also works successfully under different illumination conditions by means of automatic camera exposure adjustment. Besides been a tracking system, the proposed system can also perform 2D reconstruction of the chessboard states and generate movement logs.

2. Equipment and Setup

In this work, a setup is prepared to detect the moves of the pieces during the game. The setup has the Logitech c310 webcam for the capturing footage. The camera which has 5 megapixels resolution is capable of HD 720p recording. The camera has no autofocus functionality. Only the exposure mode from the camera settings is changed to 'manual mode' for move detection process. The webcam was used on a mid-range notebook. The chessboard and pieces are selected to meet World Chess Federation (FIDE) requirements in terms of color and size [6]. The board and the pieces have different colors from each other. The colors of the pieces are black and white, while the board has dark and light brown colored squares. The camera is positioned over the chessboard by a long and flexible holder as shown in Figure 1.

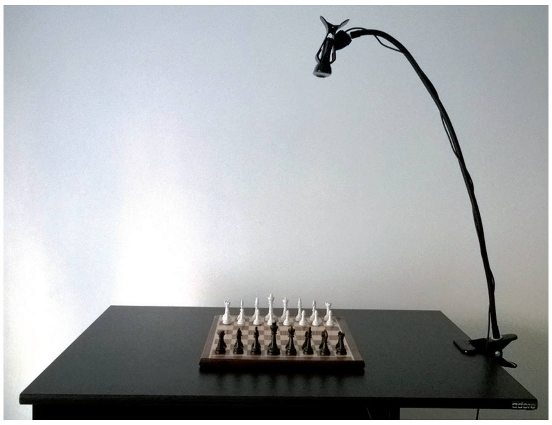


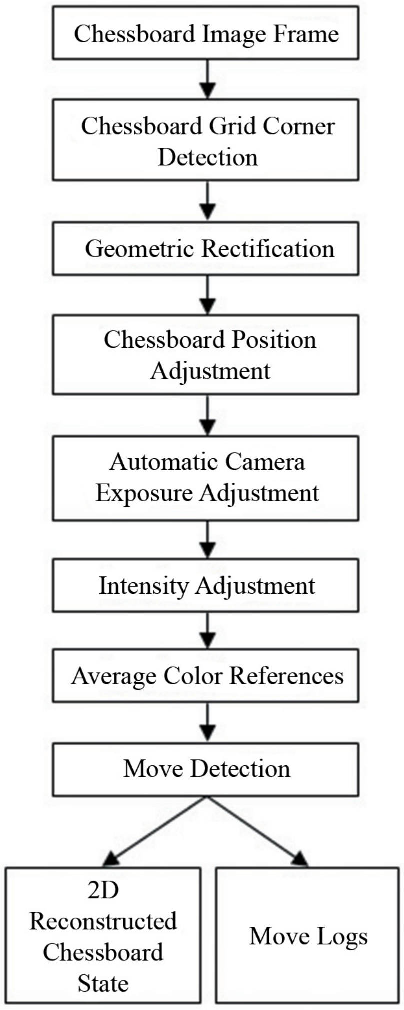
Figure 1. The image of the setup

3. The Overall Framework

The general block diagram of the proposed system is given in Figure 2. The details of the steps of the proposed framework are given in the further subsections.

3.1. Chessboard Grid Corner Detection

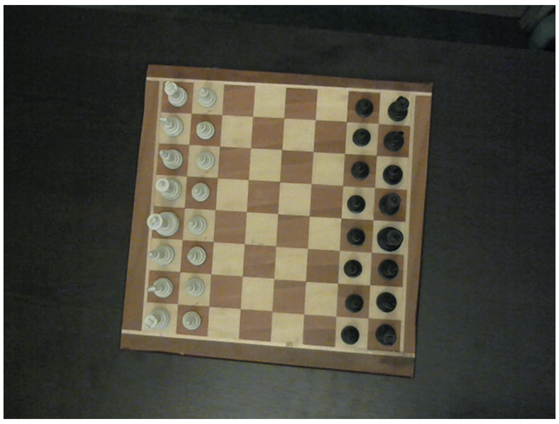
In this process, the first step is to find all grid corner points of the chessboard (Figure 3(a)) by using the snapshot of the camera. To find grid corners (Figure 3(b)), we used detectCheckerboardPoints function of MATLAB. The function that is particularly used in camera calibration gets an RGB image as an input and returns the located grid corners and the size of the board as an output. Until all grid corners are

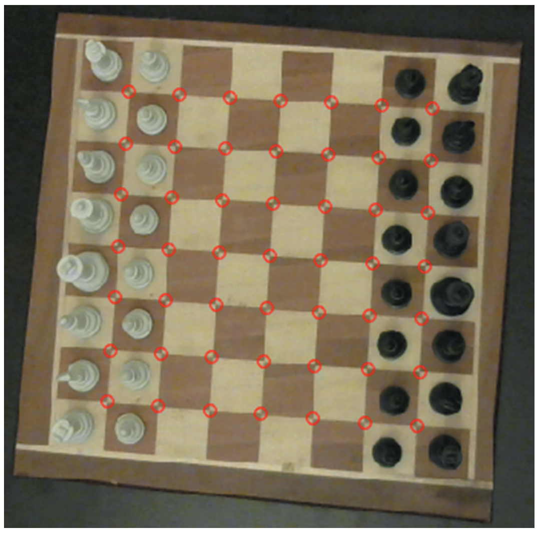

Chessboard Image Frame
Chessboard Grid Corner Detection
Geometric Rectification
Chessboard Position Adjustment
Automatic Camera Exposure Adjustment
Intensity Adjustment
Average Color References
Move Detection
2D Reconstructed Chessboard State
Move Logs
Figure 2. The overall framework

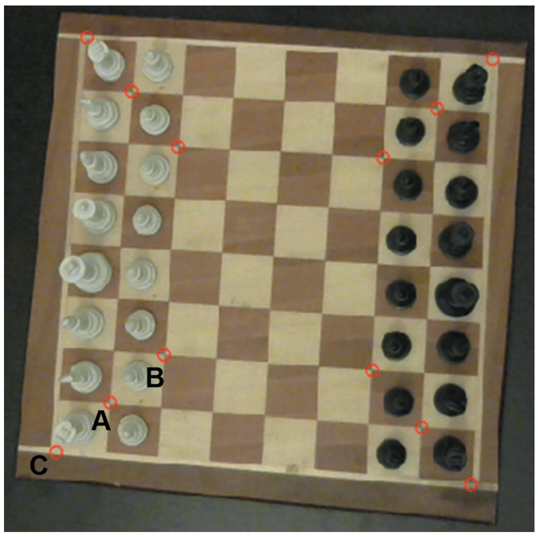
located, the saturation value of the captured image is increased gradually as a pre-process step. Once all grid corners are located, the second step is to locate the chessboard corners (point-C in Figure 3(c)). The grid corner points which are closest to the corners of the image are selected as pivot points. Point-A in Figure 3(c) is one of the pivot points. The diagonal closest inner point to the point-A is point-B, which is shown in Figure 3(c). The reflection of the point-B over the point-A is the point-C, which is the one of the chessboard corners as shown in Figure 3(c). This procedure is applied for all remaining corner points.

3.2. Geometric Rectification

The geometric rectification is an essential step to isolate the chessboard from the environment and correct the perspective distortion of the chessboard to pave the way for the other processes. The chessboard


(a)


(b)



(c)

Figure 3. (a) The original chessboard image, (b) the detected chessboard grid corners and (c) the related points to chessboard grid corner detection

is warped from its corner points which are located in the previous section to coincide with our predetermined size square corners (480x480px) (Figure 4).

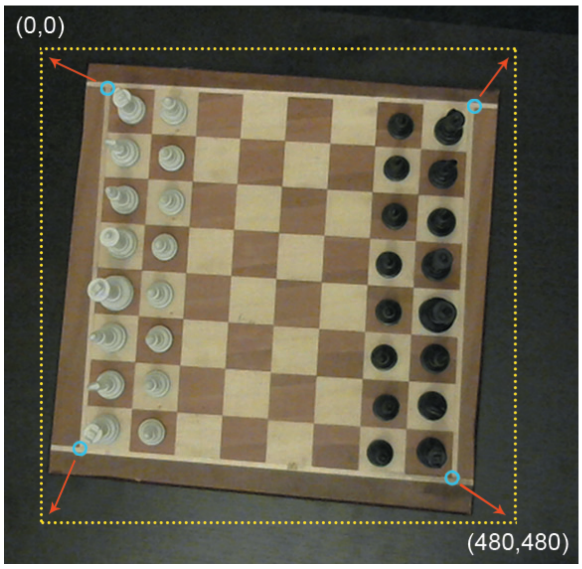


Figure 4. The chessboard before geometric rectification step

This process is applied only once before the game starts therefore, either the camera or the board should not be moved during the game. The geometrically corrected chessboard is presented in Figure 5.

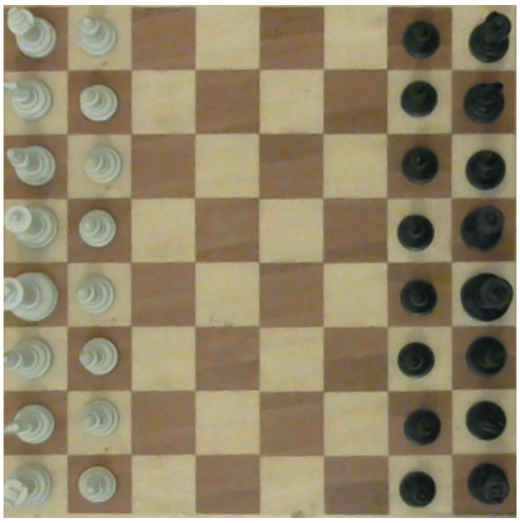


Figure 5. The chessboard after geometric rectification step

3.3. Chessboard Position Adjustment

To ease the calculations of the future processes, the white pieces are needed to be positioned at the bottom of the view. Thanks to camera position, we know that the positions of the pieces have to be on the left and right side of the camera. The comparison of the average colors of the both side's king square gives us the position of the white pieces. According to the white side position, a new transformation matrix is computed to be used in the future warping processes. In Figure 6, the white pieces are located at the bottom while the black ones are at the top.

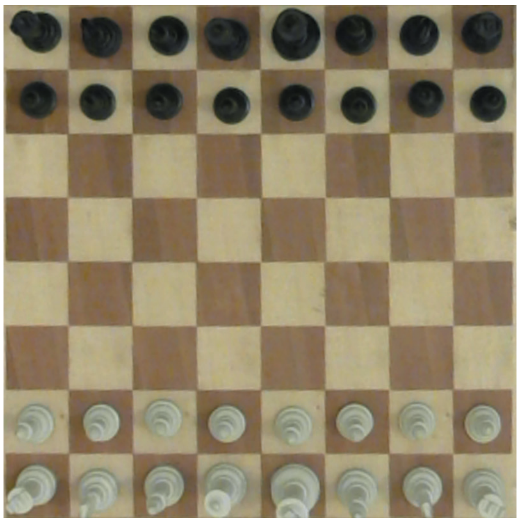


Figure 6. The chessboard after chessboard position adjustment

3.4. Automatic Camera Exposure Adjustment

The built-in automatic exposure mode of the camera may cause undesirable image acquisition for the move detection. In this mode, the camera continuously adjusts the exposure level according to the captured footage. Especially, whenever the player makes a move, the camera changes its exposure level due to the player hand on the captured image. In addition, the exposure level which is adjusted by built-in automatic exposure mode of the camera can be under or overexposure. In order to find optimum exposure level of the camera, it needs to be adjusted manually at the beginning of the game. The aim of this process is to get correct color values as much as possible by preventing under and overexposure situations. We proposed our automatic camera exposure algorithm that aims to find the optimum exposure level which maximizes the average of the color differences between light/dark piece and square (Figure 7). The calculated optimum exposure level is set to camera as a new exposure level for the following processes. In the present case the computed exposure level was computed to be -6 where the full range is between -9 (the darkest) and 0 (the lightest). The snapshot of the chessboard after applying the computed exposure level is given in Figure 8.

```
0  Begin Procedure: FindOptimumExposureLevel
1  FOREACH Exposure Level
     2  CALCULATE the average color of dark squares, dark pieces, light squares and light pieces
     3  CALCULATE the color distance between the average color of dark squares and dark pieces
     4  CALCULATE the color distance between the average color of light squares and light pieces
     5  CALCULATE the average of the color distances
     6  IF the exposure level is the first exposure level THEN
          7  SET the average to the maximum average
          8  SET the exposure level to the optimum exposure level
     9  ELSE
          10  IF the average is greater than the max average THEN
                11  SET the average to the maximum average
                12  SET the exposure level to the optimum exposure level
          13  ENDIF
     14  ENDIF
15  ENDFOR
16  RETURN the optimum exposure level
17  End Procedure: FindOptimumExposureLevel
```

Figure 7. The pseudo-code of the automatic camera exposure adjustment

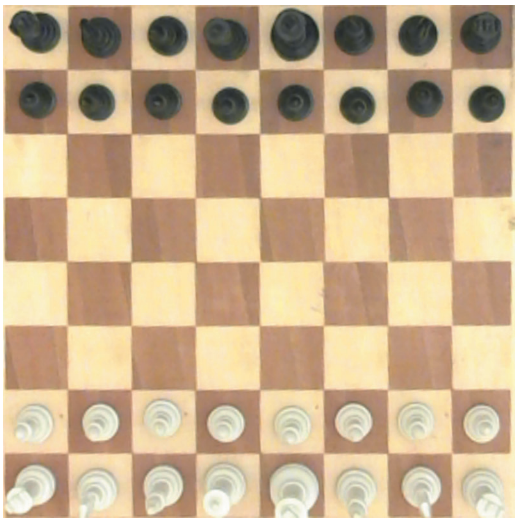


Figure 8. The chessboard after automatic camera exposure adjustment

3.5. Intensity Adjustment

To improve the image quality, a set of enhancements is applied to the snapshots of the camera. The first one is to reduce the noise problem. We used a 5x5 median filter to minimize the noise level of the images. The second one is to increase the saturation of the image to enhance colors. After this process, the average colors of pieces and squares are calculated to be used in further processes. The image of the chessboard after the intensity adjustment step is illustrated in Figure 9.

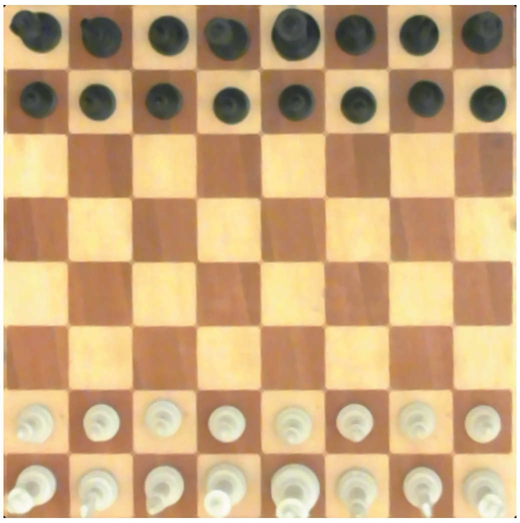


Figure 9. The chessboard after intensity adjustment

3.6. Average Color References

After all enhancements, in order to get color values of each square of the chessboard, the image of the chessboard (Figure 9) divided into 64 identical pieces each in correspondence to a square of the board. Therefore occupancy grids are created for the chessboard. After that, it is defined a region of interest (ROI) for each square (grid) of the chessboard. The primary aim of using ROIs is to get color information of the piece. ROI is defined as a 25x25px rectangle from the center of each square as shown in Figure 10.

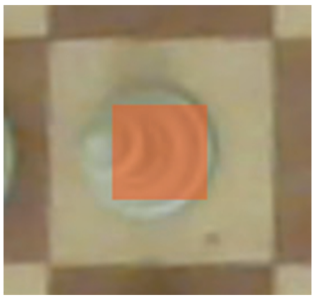


Figure 10. The orange colored region of interest superimposed on the pawn

At the beginning of the game, before the move detection, the average color values of the light/dark pieces and squares are received and recorded as reference values.

The reference colors of each type of piece and square calculated as follows:

- Reference color of the light pieces is calculated by taking the average of the 16 squares that are occupied by the light pieces.
- Reference color of the dark pieces is calculated by taking the average of the 16 squares that are occupied by the dark pieces.
- Reference color of the light squares is calculated by taking the average of the 16 light squares that are not occupied by any pieces.
- Reference color of the dark squares is calculated by taking the average of the 16 dark squares that are not occupied by any pieces.

3.7. Move Detection

The implementation of the move detection is based on a comparison between the reference image and the snapshot of the camera. For this process, the reference image is used as the first snapshot which is taken after each valid move. The first reference image is regarded as the first snapshot of the footage. During the game, the average color difference is calculated between the reference image and snapshots. Whenever the result of the calculation exceeds a predetermined threshold, we conclude that the player makes a move. After the result goes down below the threshold, we assume that the player finished the move.

At this point, the last snapshot is interpreted to determine the color and position of the pieces. Before this process, the last snapshot is warped and the enhancements are applied to the warped image of the chessboard. The ROI within each square of the image is compared with the four reference colors which are determined in section 3.6. In this comparison, the color differences are calculated in Lab color space by computing the deltaE value that represents the Euclidean distance of the related items. As a result of the comparisons, the reference color that gives the minimum deltaE value determines if a grid cell is a square or a piece with light or dark color. By applying this process to all squares of the chessboard, the chessboard state of the last snapshot is revealed. The state of the last snapshot and the previous chessboard state are compared to detect the move of the piece. The previous chessboard state represents the chessboard state of the last valid move. At the beginning of the game, the first state of the game is stored as the previous chessboard state.

When the state of the snapshot and the previous chessboard state are compared, six different outcomes can be obtained:

1. If there is no difference between previous and last states, this means there is no change in the game. For this reason, the color difference over the board is not a move.

2. If there are only one occupied and only one unoccupied squares difference with the same piece color then this is a move.

3. If there are two occupied and two unoccupied squares difference with the same piece color, then this is a special move called 'castling'.

4. If there are one occupied and one unoccupied squares difference with the same piece color and one unoccupied square difference with the other piece color, then this is another special move called 'en passant'.

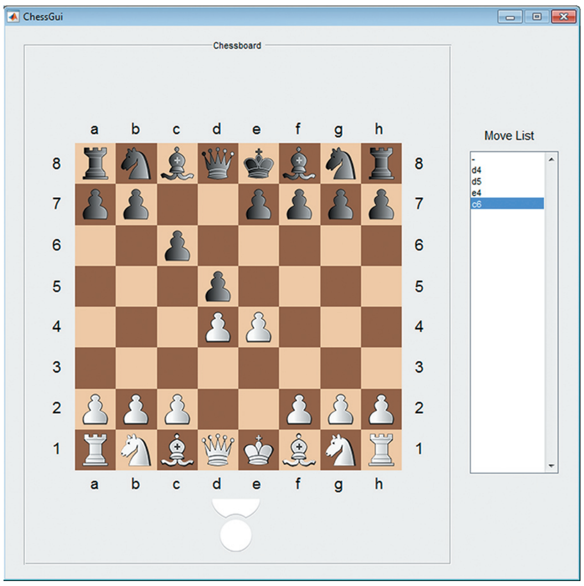
5. If there is only one unoccupied square difference and if there is a piece color change to the previous piece color of the unoccupied cell in any other occupied square, then this is a capturing move.

6. For all other conditions, the result of the comparison is not a move.

If the result is a move then the state of the chessboard is updated as the last chessboard state. The move is added to the move list and the last state of the chessboard is reconstructed in 2D. An example 2D state reconstructed from a test game and the move list are presented in Figure 11. The moves are logged as standard algebraic notation which is the notation standardized by World Chess Federation (FIDE) [5]. Note that all the steps of the proposed methodology including the graphical user interface were implemented in MATLAB.

4. Experimental Evaluation and Discussion

In order to test the system, three chess games are played at different times having different illumination conditions. In these tests, 162 moves of all 164 moves are successfully detected by the system. The corner points of the chessboard are successfully located in all games. The system performance was found to be satisfactory to detect moves in real-time.



-
d4
d5
e4
c6

Figure 11. The reconstructed chessboard state with move list

The saturation enhancement which is applied to the images taken from capturing footage helped to increase the accuracy of the average color differences.

The combination of the lighting, camera settings and chess set are playing a big role in the success of detecting moves in a chess game. Although the proposed system works well under different illumination conditions, lighting environments (having a single light source) that cast strong shadows over the board are unsuitable for tracking.

On the other hand, shadows over the light pieces are another important problem. This makes difficult to separate the light pieces from the light squares, as in 2 of 164 undetected moves during the experimental evaluation. In addition, this problem may cause to get incorrect results from the automatic camera exposure adjustment method.

Shadows and specular reflections over a particular area of the chessboard can break the uniformity of the colors. In these conditions, a chess game cannot be tracked by the proposed system. Besides, due to the reference colors of pieces and squares are determined at the beginning of the game, the overall illumination of the environment should not change dramatically during the game. Otherwise, the move detection cannot be possible by the system.

5. Conclusion

In this paper, we have presented a real-time system that performs the detection of the chess moves. The preprocessing steps are found to be quite useful.

In particular, automatic camera exposure adjustment highly reduces the color ambiguities. The environments which are heavily under the influence of directional lights are not recommended because of casting strong shadows. The results of the played games indicate that the proposed system can be an affordable and efficient option among chess game tracking systems.

As an addition to the current system, a chess move validation system is under progress to interpret the player moves. By this way, the system not only tracks the position of the pieces but also validates the movements according to the type of the piece. Therefore, the future system can be used to help decision making and monitoring by referees and anti-cheat committee.

References

[1] M. Ataş, Y. Doğan, and İ. Ataş. Chess playing robotic arm. *Proceedings of the IEEE 22nd Signal Processing and Communications Applications Conference*, pages 1171–1174, 2014. 1

[2] S. Bennet and J. Lasenby. Robust recognition of chess-boards under deformation. *Proceedings of the 20th IEEE International Conference on Image Processing*, pages 2650–2654, 2013. 1

[3] S. Bennet and J. Lasenby. Chess – quick and robust detection of chess-board features. *Computer Vision and Image Understanding*, 118:197–210, 2014. 1

[4] T. Cour, R. Lauranson, and M. Vachette. Autonomous chess-playing robot, 2006. 1

[5] FIDE. Handbook, 2015. Laws Of Chess. 6

[6] FIDE. Handbook, 2015. Standards of Chess Equipment and Tournament Venue. 2

[7] J. Gonçalves, J. Lima, and P. Leitao. Chess robot system: A multi-disciplinary experience in automation. *Proceedings of the 9th Spanish-Portuguese Congress on Electrical Engineering*, 2005. 1

[8] I. M. Khater, A. S. Ghorab, and I. A. Aljarrah. Chessboard recognition system using signature, principle component analysis and color information. *Proceedings of the Second International Conference on Digital Information Processing and Communications*, pages 141–145, 2012. 1

[9] C. Matuszek, B. Mayton, R. Aimi, M. P. Deisenroth, L. Bo, R. Chu, M. Kung, L. LeGrand, J. R. Smith, and D. Fox. Gambit: A robust chess-playing robotic system. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4291–4297, 2011. 1

[10] M. Piskorec, N. Antulov-Fantulin, J. Curic, O. Dragoljevic, V. Ivanac, and L. Karlovic. Computer vision system for the chess game reconstruction. *Proceedings of the 34th International Convention*, pages 870–876, 2011. 1

[11] E. Sokic and M. Ahic-Dokic. Simple computer vision system for chess playing robot manipulator as a project-based learning example. *Proceedings of the IEEE International Symposium on Signal Processing and Information Technology*, pages 75–79, 2008. 1

[12] D. Urting and Y. Berbers. Marineblue: A low-cost chess robot. *Proceedings of the International Conference Robotics and Applications*, pages 76–81, 2003. 1

21st Computer Vision Winter Workshop
Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.)
Rimske Toplice, Slovenia, February 3–5, 2016

Fast L_1-based RANSAC for homography estimation

Jonáš Šerých, Jiří Matas, Ondřej Drbohlav
Czech Technical University in Prague, Faculty of Electrical Engineering,
Department of Cybernetics, Center for Machine Perception,
Technická 2, 166 27 Praha 6, Czech Republic
{serycjon,matas,drbohlav}@fel.cvut.cz

Abstract. *We revisit the problem of local optimization (LO) in RANSAC for homography estimation. The standard state-of-the-art LO-RANSAC improves the plain version's accuracy and stability, but it may be computationally demanding, it is complex to implement and requires setting multiple parameters. We show that employing L_1 minimization instead of the standard LO step of LO-RANSAC leads to results with similar precision. At the same time, the proposed L_1 minimization is significantly faster than the standard LO step of [8], it is easy to implement and it has only a few of parameters which all have intuitive interpretation. On the negative side, the L_1 minimization does not achieve the robustness of the standard LO step, its probability of failure is higher.*

1. Introduction

RANSAC [3] is a robust model fitting algorithm that is the standard method used for two-view geometry estimation [5]. The plain version of RANSAC proceeds as follows: (i) randomly sample the minimum number of points required to calculate model parameters, (ii) compute the cardinality of the set consistent with that model, i.e. the number of inliers, and (iii) terminate if the probability that a better model than the one best so far will be found falls under a predefined threshold. The precision of the model returned by the algorithm is typically improved by least square fitting of the inliers of the best mode.

It has been observed [11] that the termination criterion (iii) stops the process later than expected given the recovered percentage of inliners. The discrepancy is due to a generally incorrect (overoptimistic) assumption that every minimal sample of inliers generates a "good" model, i.e. a model that will be consistent with all correct correspondences.

The problem was first addressed in a paper by Chum et al [2] who proposed an additional RANSAC step, the so called local optimization (LO). The LO step is employed whenever a new candidate model M is the best one so far found in the RANSAC loop, i.e. it has more inliers than any of the models estimated from the random minimal samples evaluated so far. Chum et al [2] proved that with the strategy, the LO step is run only $\log(k)$ times, where k is the number of random models tested.

The local optimization step[2] performs various heuristic procedures with the objective of increasing the accuracy of M, such as generating hypotheses by resampling the inliers of M and performing iterative least square estimation combined with scheduled inlier threshold changes. The standard implementation of RANSAC with the local optimization step, found in the commonly used publicly available code [10], is a combination of the above-mentioned heuristic procedures.

The choice and parameter settings of local optimization methods influence the speed and accuracy of the algorithm. In the state-of-the-art version [8], the LO step executes a complex procedure which involves repeated sampling from inliers of M and repeated iterative least squares minimisation. As the sampling is involved, it is stochastic[1]. Due to both repeated sampling and iterative least squares, it is so computationally demanding, in comparison with RANSAC steps (i) and (ii), that despite being executed only rarely, the LO step significantly influences the overall running time.

In this paper, we propose to replace the complex LO procedure of Lebeda et al. by minimization of the

[1]Since the outer loop of RANSAC is stochastic, the inner sampling does not change the character of the algorithm.

```
1: procedure STANDARD LO
2:     Input: $M$ (model estimated by LSq),
           $I$ (inliers)
3:     for $r = 1 \rightarrow reps$ do
4:         sample $S$ drawn from inliers
5:         model $M$ is estimated from $S$
6:         iterative least squares on $M$
7:     end for
8:     return best model
9: end procedure
```

```
1: procedure $L_1$-BASED LO
2:     Input: $M$ (minimum sample model),
           $I$ (inliers)
3:     while stopping condition not met do
4:         $M \leftarrow$ model estimated from inliers by IRLS optimization
5:         $I \leftarrow$ inliers to $M$
6:     end while
7:     return $M$
8: end procedure
```

Table 1: Comparison of the standard and proposed local optimization procedures in RANSAC – left and right columns, respectively. IRLS stands for interated re-weighted least squares. Note that the standard LO method includes several rounds of IRLS s which are themselves computationally demanding (for details, see [8]).The iteration stops if either the change in the cost function is below 10^{-3} or the maximum number of iterations is reached (set to 5).

sum of L_1 norms of the residuals, ie. the algebraic errors of the model on individual points. The minimizer of the L_1 norm, also known as geometric median, is robust to a modest contamination by outliers. This means that RANSAC becomes less sensitive to the inlier-outlier threshold. The threshold, a critical parameter of RANSAC, can be set more loosely and thus cover a wide range of problems. Moreover, the L_1-based procedure need not include least square estimation with multiple thresholds, thus saving time.

In practice we replace the L_1 norm by the Huber robust kernel response to the inlier algebraic errors. The Huber cost function is defined in Eq. 6. The Huber kernel is differentiable and convex and the global minimum of the cost function can be found by gradient descent. The gradient minimization alternates with the inlier-outlier selection process. The alternating minimisation can be seen as *local optimization* of the truncated Huber kernel. The procedure has only a small number of parameters that have intuitive meaning, it is simple, and deterministic.

We show that the minimization produces errors which are comparable to the standard LO-RANSAC, while being computationally much less expensive – of an order of magnitude in our experiments compared to the standard local optimization.

2. Method

The difference of the standard and proposed LO method is presented in Table 1. The L_1 minimization is carried out by iterated reweighted least squares (IRLS). The particular instantiation of IRLS is knows as the generalized Weiszfeld algorithm [1]. Weiszfeld proved that the geometric mean minimiation by IRLS requires solving repeated least squares problems where each data point is weighted by the reciprocal of its residual to the current estimate of the model. The algorithm has to be modified to avoid singularities when some point is exactly consistent with the model, i.e. it has a zero residual. To avoid the problem, we replace L_1 minimization with Huber kernel minimization. In the implementation, it only means that points with small residuals are not scaled.

First, the necessary notation is introduced. The L_2^2 norm (for a vector $\mathbf{r} \in \mathbb{R}^D$) is defined as:

$$\|\mathbf{r}\|_2^2 = \sum_{j=1}^{D} |r_j|^2, \tag{1}$$

the L_2^1 norm (for $\mathbf{r} \in \mathbb{R}^D$) as

$$\|\mathbf{r}\|_2^1 = \sqrt{\sum_{j=1}^{D} |r_j|^2} \tag{2}$$

2.1. Homography estimation by algebraic error minimization in L_2^2 and L_2^1 norms

Let the number of correspondences be M. The data matrix $\mathbf{Z}$ is computed from correspondences by a standard procedure ([5]): Let (x, y) and (x', y') be the correspondence pair. It generates two rows into the data matrix $\mathbf{Z}$:

$$\begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -x'x & -x'y & -x' \\ 0 & 0 & 0 & x & y & 1 & -y'x & -y'y & -y' \end{bmatrix}. \tag{3}$$

Let $\mathbf{z}^{(i)}$ denote the two rows generated by i-th correspondence. The homography $\mathbf{h}$ is estimated from $\mathbf{Z}$ by one of the following optimizations:

The L_2^2 optimization

$$\mathbf{h} = \operatorname*{argmin}_{\hat{\mathbf{h}}} \sum_{i=1}^{M} \|\mathbf{z}^{(i)}\hat{\mathbf{h}}\|_2^2 , \quad (\text{subj. to } \hat{\mathbf{h}}^\top \hat{\mathbf{h}} = 1) \tag{4}$$

The minimization is solved by computing the spectral decomposition of $\mathbf{Z}^\top\mathbf{Z}$ and taking the eigenvector corresponding to the smallest eigenvalue. The algorithm has the following properties: it is fast, but not robust with a breakdown point of zero [7] – in general a single outlier can make $\mathbf{h}$ arbitrarily wrong[2].

The L^1 **optimization**, defined as

$$\mathbf{h} = \operatorname*{argmin}_{\hat{\mathbf{h}}} \sum_{i=1}^{M} \|\mathbf{z}^{(i)}\hat{\mathbf{h}}\|^1 , \quad (\text{subj. to } \hat{\mathbf{h}}^\top \hat{\mathbf{h}} = 1) \tag{5}$$

is robust and can be solved by the generalized Weiszfeld algorithm, an instance of IRLS. Instead of modifying the algorithm to take of the technical problems associated with the Weiszfeld algorithm arising if one of the residuals vanishes, we instead optimize the response to the Huber kernel.

Huber optimization is defined as

$$\mathbf{h} = \operatorname*{argmin}_{\hat{\mathbf{h}}} \sum_{i=1}^{M} \begin{cases} \frac{1}{2}\|\mathbf{r}^{(i)}\|_2^2 & : \|\mathbf{r}^{(i)}\|_2^1 \leq k \\ k(\|\mathbf{r}^{(i)}\|_2^1 - \frac{k}{2}) & : \|\mathbf{r}^{(i)}\|_2^1 \geq k \end{cases}$$
$$(\text{subj. to } \hat{\mathbf{h}}^\top \hat{\mathbf{h}} = 1 \text{ and } \mathbf{r}^{(i)} = \mathbf{z}^{(i)}\hat{\mathbf{h}}) \tag{6}$$

The minimization is carried out by a slightly modified Weiszfeld algorithm ([12]), an iterative reweighted least squares method:

```
1: procedure IRLS OPTIMIZATION
2:     Initialize $\mathbf{h}$ as the estimate obtained from the minimal sample $\mathbf{h}$
3:     while stopping condition not met do
4:         Compute the geometric error $\mathbf{r}^{(i)}$:
               $\mathbf{r}^{(i)} = \|\mathbf{z}^{(i)}\mathbf{h}\|_2^1 \quad (\forall i = 1, 2, ..., M)$    (7)
5:         Reweight $\mathbf{Z}$:
               $\mathbf{z}^{(i)} \leftarrow \sqrt{\mathbf{w}^{(i)}}\mathbf{z}^{(i)}$    (8)
               $(\forall i = 1, 2, ..., M)$
6:         Recompute $\mathbf{h}$ using $L_2^2$ optimization (4)
7:     end while
8: end procedure
```

[2]In RANSAC, the error on a single point is bounded by the inlier threshold. In practice, points close to the the inlier-outlier boundary make the outcome of standard RANSAC unstable.

The iteration stops when

$$\sum_i \mathbf{r}_t^{(i)} - \sum_i \mathbf{r}_{(t+1)}^{(i)} \approx 0$$

, i.e. if the value of the cost function does not change between consecutive iterations or after 5 iterations are completed. The second condition reflects the empirical observation that most of the time, the IRLS algorithm converges after 3 iterations and it is used only as a guarantee against an infinite loop.

In the case of L_2^1 optimization, the weight $\mathbf{w}^{(i)}$ is set to $1/\left(\|\mathbf{r}^{(i)}\|_2^1 + \delta\right)$. A small constant δ is used to avoid the problem of dividing by zero when the residuals vanish.

The L^1 optimization proposed above introduces additional parameter δ in order to deal with the division by zero, but its interpretation is not clear. Using the Huber cost function instead of the L^1 norm avoids the numerical issue. The weight $\mathbf{w}^{(i)}$ is set as follows ([13]).

$$\mathbf{w}^{(i)} = \begin{cases} 1 & : \|\mathbf{r}^{(i)}\|_2^1 \leq k \\ k/\|\mathbf{r}^{(i)}\|_2^1 & : \|\mathbf{r}^{(i)}\|_2^1 \geq k \end{cases}$$

The additional Huber parameter k can be intuitively seen as a smoothing factor between L_2^2 and L_2^1 norms or, alternatively, like a lower bound on the inlier threshold.

The motivation for using this optimization is its robustness. It is closely related to geometric median computation and the formulation is convex.It is a well known property of median that it is robust to outliers for up to 50% contamination of samples by the outliers. The property makes the procedure non sensitive to the choice of the inlier-outlier threshold of the "outer" RANSAC loop.

3. Experiments.

We compared the standard RANSAC, the state-of-the-art LO-RANSAC and the proposed L_1-based RANSAC on a dataset consisting of 42 image pairs, including selected images from the ZuBuD dataset [4], images from Lebeda's homog dataset [9] used for evaluation of the LO-RANSAC, and images from the symbench dataset [6]. The Hessian Affine feature detector with SIFT descriptor was used for obtaining the tentative correspondences.

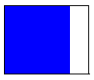
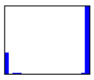
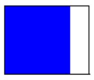
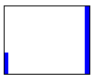
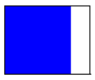
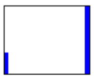
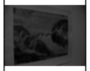
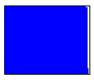
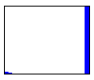
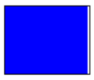
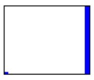
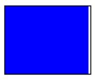
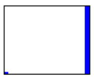
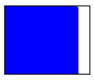
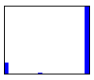
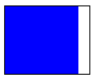
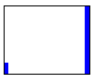
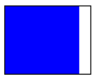
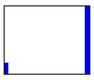
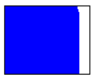
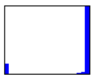
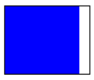
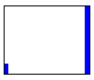
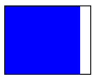
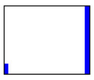
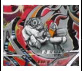
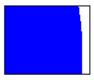
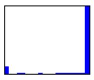
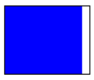
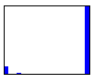
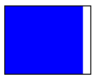
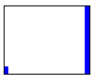
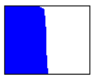
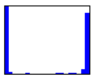
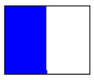
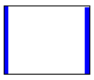
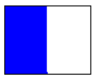
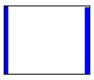
Image		Qty↓	RANSAC			LO-RANSAC			L_1-based RANSAC		
05		I	**953.2**	±0.9	(950-956)	**953.0**	±0.0	(953-953)	**953.0**	±0.1	(952-953)
		LO time (μs)	**0.0**	±0.0	(0-0)	**29158.8**	±3383.8	(27499-42497)	**3934.6**	±1035.1	(1901-6479)
		I (%)	**76.9**	±0.1	(77-77)	**76.9**	±0.0	(77-77)	**76.9**	±0.0	(77-77)
		Samp	**11.8**	±5.7	(7-35)	**11.8**	±5.7	(7-35)	**7.5**	±1.9	(7-19)
											
		Time$_{(ms)}$	**6.1**			**35.5**			**13.7**		
		Error	**0.74**	±0.05	(0.6-0.9)	**0.72**	±0.00	(0.7-0.7)	**0.73**	±0.01	(0.7-0.8)
		LO count	**0.0**	±0.0	(0-0)	**1.0**	±0.0	(1-1)	**2.2**	±0.9	(1-5)
adam		I	**250.8**	±1.1	(244-252)	**251.0**	±0.0	(251-251)	**251.0**	±0.0	(251-251)
		LO time (μs)	**0.0**	±0.0	(0-0)	**10922.9**	±1797.6	(8737-15292)	**1318.5**	±214.1	(917-1917)
		I (%)	**97.6**	±0.4	(95-98)	**97.7**	±0.0	(98-98)	**97.7**	±0.0	(98-98)
		Samp	**5.0**	±2.6	(2-14)	**5.0**	±2.6	(2-14)	**2.0**	±0.3	(2-4)
											
		Time$_{(ms)}$	**2.3**			**14.4**			**4.4**		
		Error	**1.15**	±0.45	(0.4-2.8)	**0.77**	±0.05	(0.6-0.8)	**0.79**	±0.02	(0.6-0.8)
		LO count	**0.0**	±0.0	(0-0)	**1.0**	±0.0	(1-1)	**1.4**	±0.5	(1-2)
boat		I	**328.4**	±0.5	(328-330)	**328.0**	±0.2	(328-329)	**328.0**	±0.0	(328-328)
		LO time (μs)	**0.0**	±0.0	(0-0)	**13874.9**	±2006.0	(11071-16489)	**1738.3**	±323.7	(917-2428)
		I (%)	**86.2**	±0.1	(86-87)	**86.1**	±0.1	(86-86)	**86.1**	±0.0	(86-86)
		Samp	**6.2**	±2.5	(4-15)	**6.2**	±2.5	(4-15)	**4.1**	±0.4	(4-7)
											
		Time$_{(ms)}$	**2.6**			**17.8**			**5.9**		
		Error	**1.30**	±0.14	(1.1-2.1)	**1.23**	±0.01	(1.2-1.2)	**1.24**	±0.00	(1.2-1.2)
		LO count	**0.0**	±0.0	(0-0)	**1.0**	±0.0	(1-1)	**1.7**	±0.7	(1-3)
Brussels		I	**450.0**	±3.5	(428-451)	**451.0**	±0.0	(451-451)	**451.0**	±0.0	(451-451)
		LO time (μs)	**0.0**	±0.0	(0-0)	**16342.6**	±2310.4	(13755-19648)	**2094.9**	±347.1	(1090-3084)
		I (%)	**87.2**	±0.7	(83-87)	**87.4**	±0.0	(87-87)	**87.4**	±0.0	(87-87)
		Samp	**8.3**	±4.2	(4-22)	**8.3**	±4.2	(4-22)	**4.1**	±0.3	(4-6)
											
		Time$_{(ms)}$	**3.4**			**20.6**			**6.6**		
		Error	**1.39**	±0.37	(1.1-3.3)	**1.24**	±0.00	(1.2-1.2)	**1.24**	±0.00	(1.2-1.3)
		LO count	**0.0**	±0.0	(0-0)	**1.0**	±0.0	(1-1)	**1.8**	±0.7	(1-3)
graf		I	**840.1**	±9.8	(808-848)	**846.2**	±0.4	(846-847)	**846.0**	±0.0	(846-846)
		LO time (μs)	**0.0**	±0.0	(0-0)	**24032.7**	±2219.6	(21845-29919)	**4274.4**	±834.5	(1794-6007)
		I (%)	**89.9**	±1.1	(87-91)	**90.6**	±0.0	(91-91)	**90.6**	±0.0	(91-91)
		Samp	**7.3**	±3.5	(3-20)	**7.3**	±3.5	(3-20)	**3.2**	±0.7	(3-8)
											
		Time$_{(ms)}$	**4.8**			**29.5**			**11.9**		
		Error	**1.69**	±0.22	(1.4-2.7)	**1.45**	±0.00	(1.4-1.5)	**1.45**	±0.01	(1.4-1.6)
		LO count	**0.0**	±0.0	(0-0)	**1.0**	±0.0	(1-1)	**1.7**	±0.7	(1-4)
sym_notredame13		I	**89.6**	±2.4	(77-93)	**91.0**	±0.2	(91-92)	**91.0**	±0.2	(90-92)
		LO time (μs)	**0.0**	±0.0	(0-0)	**8090.3**	±1025.8	(7196-10973)	**707.8**	±131.0	(437-1009)
		I (%)	**48.4**	±1.3	(42-50)	**49.2**	±0.1	(49-50)	**49.2**	±0.1	(49-50)
		Samp	**110.6**	±53.7	(45-257)	**54.0**	±4.0	(45-67)	**46.1**	±14.8	(37-123)
											
		Time$_{(ms)}$	**4.2**			**11.7**			**5.9**		
		Error	**1.81**	±0.62	(1.1-4.7)	**1.13**	±0.02	(1.1-1.2)	**1.15**	±0.09	(1.1-1.7)
		LO count	**0.0**	±0.0	(0-0)	**1.0**	±0.0	(1-1)	**2.9**	±1.2	(1-7)

Table 2: Results on six pairs representing well the whole dataset with the exception of cases in Tab.4. The number of inliers found (*I*), the inlier ratio *I(%)*, the LO step time (LO time), the number of RANSAC samples (*Samp*), CPU time (*time*), the mean error on ground truth correspondences (*Error*) and the number of local optimizations (*LO*). Values in bold are means over 100 runs. The ± entries are standard deviations, minimum and maximum are shown in parentheses. The blue plots represent the stability of each algorithm over 100 runs. The left one represents a probability of a tentative correspondence to be an inlier (probability on the vertical axis, correspondence index on the horizontal axis). The correspondences were sorted so that the plot is non-increasing. In the ideal case, the plot should look like a rectangle. Any other shape indicates that some of the tentative correspondences were not classified as inliers/outliers consistently over the 100 runs. The second plot is a histogram of the first plot.

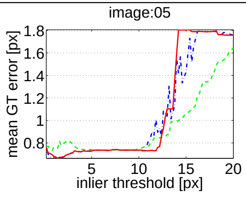
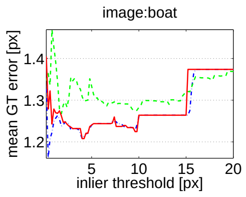
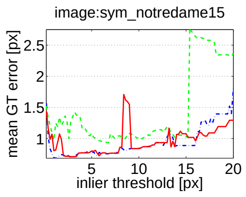
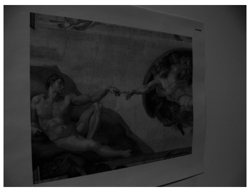
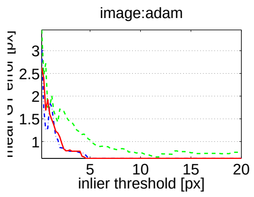
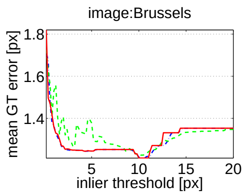
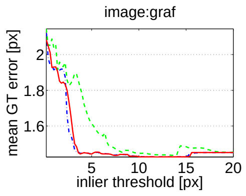
First image	Second image	Thr. sensitivity
		
		
		
		
		
		

Table 3: The dependence of the ground truth error on the inlier threshold (RANSAC green, LO-RANSAC blue, L_1-based RANSAC red). Note that the proposed L_1 algorithm yields results very similar to LO-RANSAC. The ground truth error was averaged over 10 runs for each of the methods. Experimental results demonstrated on the same image pairs as in Table 2.

The RANSAC parameters common to all three tested versions used in our experiments are summarized in table 6. The inlier threshold θ is set, following[8] given σ in the following way:

$$\theta = 5.99\,(\sigma S)^2$$

where $S = \max(w, h)/768$ is a scale factor dependent image dimensions. The 5.99 term is the 95% percentile of the χ^2 distribution with two degrees of freedom.

Additional parameters used for the standard LO-RANSAC are summarized in Table 7. The proposed

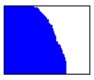
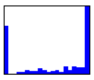
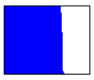
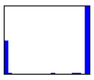
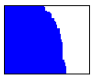
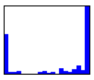
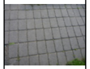
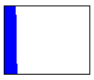
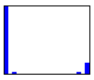
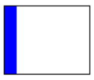
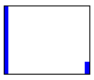
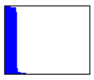
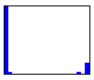
Image		Qty↓	RANSAC			LO-RANSAC			L_1-based RANSAC		
BruggeSquare		I	**201.0**	±12.9	(172-234)	**227.2**	±1.3	(224-232)	**214.9**	±11.4	(195-228)
		LO time $_{(\mu s)}$	**0.0**	±0.0	(0-0)	**12567.0**	±1944.9	(10166-16251)	**993.8**	±187.2	(598-1650)
		I (%)	**60.0**	±3.9	(51-70)	**67.8**	±0.4	(67-69)	**64.1**	±3.4	(58-68)
		Samp	**52.6**	±24.8	(15-153)	**42.5**	±10.9	(15-59)	**17.7**	±5.7	(9-41)
		Time$_{(ms)}$	**5.2**			**18.0**			**5.8**		
		Error	**3.50**	±1.25	(1.2-6.2)	**2.44**	±0.12	(2.0-2.7)	**2.93**	±0.91	(1.3-4.6)
		LO count	**0.0**	±0.0	(0-0)	**1.0**	±0.0	(1-1)	**2.7**	±1.2	(1-5)
dlazky		I	**11.0**	±0.2	(9-11)	**11.0**	±0.0	(11-11)	**10.9**	±0.6	(7-11)
		LO time $_{(\mu s)}$	**0.0**	±0.0	(0-0)	**1531.3**	±746.7	(603-4434)	**249.7**	±68.9	(109-419)
		I (%)	**14.8**	±0.3	(12-15)	**14.9**	±0.0	(15-15)	**14.7**	±0.8	(9-15)
		Samp	**10745.0**	±5429.3	(6963-27356)	**8392.7**	±3432.8	(6963-25008)	**8220.6**	±3301.0	(4820-19947)
		Time$_{(ms)}$	**87.3**			**76.0**			**71.6**		
		Error	**2.99**	±0.95	(2.6-6.4)	**2.61**	±0.00	(2.6-2.6)	**5.43**	±20.63	(2.6-204.9)
		LO count	**0.0**	±0.0	(0-0)	**4.7**	±1.6	(1-9)	**7.0**	±3.1	(2-21)

Table 4: Results on two image pairs with unusual sensitivity to the inlier threshold. See caption of Tab. 2 for description of entries.

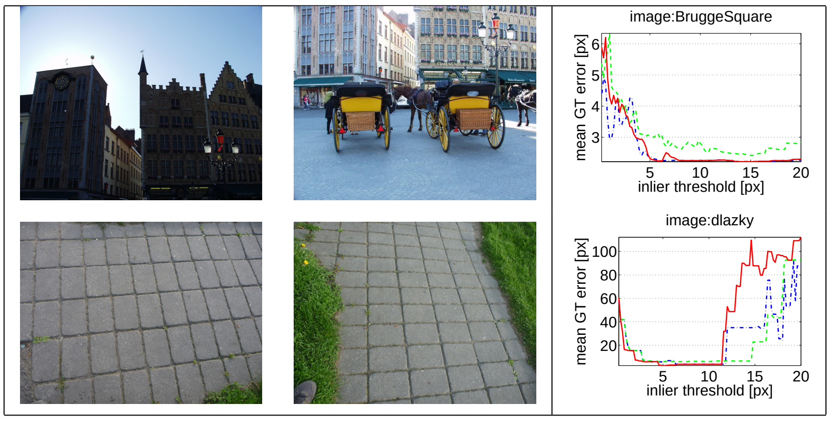


Table 5: The dependence of the ground truth error on the inlier threshold (RANSAC green, LO-RANSAC blue, L_1-based RANSAC red) for two failure cases.

confidence	0.95
σ	2.0
sample limit	500000

Table 6: RANSAC parameters

ILSQ iterations	4
ILSQ sample limit	28
threshold multiplier	4
inner RANSAC repetitions	10

Table 7: LO-RANSAC parameters

method does not introduce any new parameters.

Table 2 shows a sample of six image pairs well representing the results on the whole dataset, with the exception of a few cases described later. Note that the proposed L_1 optimization is usually about 5 times faster than the standard LO step (see 'LO time').Table 4 summarizes the performance on the few exceptional cases.

The error (see 'Error' in the table) was computed by reprojecting hand-made ground truth correspondences (about 8 of them for each image pair) by the model found by the algorithm used.

Two observations summarize the results: i) the proposed procedure yields error which is comparable to the standard LO-RANSAC, and ii) it usually runs approximately 5 times faster (see 'LO time' in the table).

Table 3 shows the comparison of the dependence of the error on the inlier threshold for standard RANSAC, standard LO-RANSAC and the proposed method. The results shown on the same subset of six image pairs which are representative of the whole dataset. The experiment confirms that the proposed procedure is able to achieve results similar to the standard LO-RANSAC.

The results for two exceptional image pairs are shown in table 5. The standard LO-RANSAC achieves good results (high stability, low error), while our proposed algorithm fails to stabilize the plain RANSAC results (the 'dlazky' pair is one of the most challenging ones from our dataset, as there are only 11 inliers).

4. Conclusions

We have shown that replacing the standard LO step of LO-RANSAC with minimization of the sum of Huber kernel responses to residuals has the following properties: it is simple, deterministic and produces similar errors as the standard LO-RANSAC and is usually approximately 5 times faster. On the negative side, in the current implementation, it has higher probability of failure than the standard LO-RANSAC.

Acknowledgements

The research was supported by CTU student grant SGS15/155/OHK3/2T/13.

References

[1] A. Beck and S. Sabach. Weiszfeld's method: Old and new results. *Journal of Optimization Theory and Applications*, 164(1):1–40, 2014. 2

[2] O. Chum, J. Matas, and J. Kittler. Locally optimized ransac. In *Pattern Recognition*, pages 236–243. Springer, 2003. 1

[3] 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(6):381–395, June 1981. 1

[4] L. V. G. H. Shao, T. Svoboda. Zubud - zurich buildings database for image based recognition. Technical report, 2003. 3

[5] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge, UK, 2000. Chapter 3 : Estimation - 2D Projective Transformations. 1, 2

[6] D. C. Hauagge and N. Snavely. Image matching using local symmetry features. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 206–213. IEEE, 2012. 3

[7] P. Huber. *Robust Statistics*. Wiley Series in Probability and Statistics - Applied Probability and Statistics Section Series. Wiley, 2004. 3

[8] K. Lebeda, J. Matas, and O. Chum. Fixing the locally optimized ransac. In R. Bowden, J. Collomosse, and K. Mikolajczyk, editors, *Proceedings of the British Machine Vision Conference*, pages 1013–1023, London, UK, September 2012. BMVA. 1, 2, 5

[9] K. Lebeda, J. Matas, and O. Chum. Fixing the locally optimized RANSAC – Full experimental evaluation. Research Report CTU–CMP–2012–17, Center for Machine Perception, Czech Technical University, Prague, Czech Republic, September 2012. 3

[10] R. Raguram, O. Chum, M. Pollefeys, J. Matas, and J. Frahm. Usac: A universal framework for random sample consensus. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):2022–2038, Aug 2013. 1

[11] B. Tordoff and D. W. Murray. Guided sampling and consensus for motion estimation. In *Computer VisionECCV 2002*, pages 82–96. Springer, 2002. 1

[12] E. Weiszfeld and F. Plastria. On the point for which the sum of the distances to n given points is minimum. *Annals of Operations Research*, 167(1):7–41, 2009. 3

[13] Z. Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. *Image and vision Computing*, 15(1):59–76, 1997. 3