Proceedings of the

18th Computer Vision Winter Workshop

Hernstein, Austria, February 4-6, 2013

Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)

Technical Report

Pattern Recognition and Image Processing Group
Institute of Computer Aided Automation
Vienna University of Technology
Favoritenstr. 9/183-2
A-1040 Vienna AUSTRIA
Phone: +43 (1) 58801-18351
Fax: +43 (1) 58801-18392
krw@prip.tuwien.ac.at
E-mail: fuensanta@prip.tuwien.ac.at
gramacha@nt.tuwien.ac.at
URL: http://www.prip.tuwien.ac.at/

PRIP-TR-129 February 4, 2013

Computer Vision Winter Workshop 2013

Walter G. Kropatsch, Fuensanta Torres and Geetha Ramachandran (Eds.)

Abstract

This technical report collects the papers presented at the Computer Vision Winter Workshop 2013 (CVWW2013).
The 18^th edition of this workshop was organized by the Pattern Recognition and Image Processing Group (PRIP), and was held in Hernstein, Austria, February 4-6, 2013. 32 Papers were submitted to the workshop this year. Each of them was reviewed by at least 2 members of the Program Committee, and 28 were selected for oral presentation at the workshop. A few authors wished that their paper is not included in these proceedings. According to the general philosophy of the CVWW series, these wishes have been accepted. 16 papers were included in the proceedings of the workshop. We would like to take the opportunity to express our sincere appreciation to the Program Committee for their careful reviews. We would also like to thank Werner Purgathofer for his excellent talk. Finally, our thanks go to all the authors for their contributions.

Walter G. Kropatsch, Fuensanta Torres and Geetha Ramachandran
Vienna, 2013

CVWW 2013

Proceedings of the 18th Computer Vision Winter Workshop

February 4-6, 2013
Hernstein, Austria

Walter G. Kropatsch, Fuensanta Torres,
Geetha Ramachandran (eds.)

Pattern Recognition and Image Processing Group
Vienna University of Technology
Vienna 2013

ISBN: 978-3-200-02943-9

Preface

The Computer Vision Winter Workshop is an annual meeting formed around research groups from the Vienna University of Technology (PRIP), Graz University of Technology (ICG), Czech Technical University in Prague (CMP), and the University of Ljubljana (CVG, MVG, ViCoS). The goal of the workshop is to communicate fresh ideas between the four groups and to provide conference experience to students. The workshop is open to students worldwide.

The 18^{th} edition of this workshop was organized by the Pattern Recognition and Image Processing Group (PRIP), and was held in Hernstein, Austria, February 4-6, 2013.

32 Papers were submitted to the workshop this year. Each of them was reviewed by at least 2 members of the Program Committee, and 28 were selected for oral presentation at the workshop. A few authors wished that their paper is not included in these proceedings. According to the general philosophy of the CVWW series, these wishes have been accepted. 16 papers were included in the proceedings of the workshop.

Besides the submitted papers, one invited talk was included. We would like to thank Werner Purgathofer (Institute of Computer Graphics and Algorithms, Vienna University of Technology) for his valuable contribution.

Last but not least, we would like to thank the members of the Program Committee for their careful reviews, and wish the participants many new contacts and fresh ideas.

Walter G. Kropatsch, Fuensanta Torres and Geetha Ramachandran
Vienna, 2013

Workshop Chairs:
Walter G. Kropatsch, Fuensanta Torres and Geetha Ramachandran

Program Committee:

TU Wien
Nicole Artner
Nicole Brosch
Margrit Gelautz
Yll Haxhimusa
Michael Hornacek
Walter G. Kropatsch

CTU Prague
Ondrej Chum
Vojtech Franc
Vaclav Hlavac
Jiri Matas
Tomas Pajdla
Radim Sara
Tomas Svoboda

Institute of Science and Technology, Austria
Stefan Jeschke
Filip Korc
Christoph Lampert
Viktoriia Sharmanska

TU Graz
Horst Bischof
Axel Pinz
Tomas Pock
Martin Urschler
Andreas Wendel

University of Ljubljana
Luka Cehovin
Matej Kristan
Bostjan Likar
Rok Mandeljc
Janez Pers
Danijel Skocaj

Austrian Institute of Technology
Csaba Beleznai
Branislav Micusik
Roman Pflugfelder

Table of Contents

Oral Session 5: Machine Learning

Oral Session 6: Tracking

Oral Session 7: Registration, Stereo and Structure

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Accurate Fast Simulation of Light

Werner Purgathofer
Vienna University of Technology
www.cg.tuwien.ac.at

Abstract.

Light distribution in a scene is a very complex issue if it shall be close to realism. For a lamp producing and light planning company such as Zumtobel it is of great value to be able to design installations interactively, providing immediate feedback to the costumers about the final result. Many aspects such as reflections and indirect lighting make this task difficult. This talk will give some ideas how a project at the research center VRVis approaches this topic, and which algorithms are useful for this. Where can we simplify without visible loss? How can we use the GPU to speed up the calculations? Why are virtual lights an efficient concept? Some example images provide evidence.

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Star Convex Object Detection by the Infinite Shape Mixture Model

Tomáš Sixta
Center for Machine Perception, Czech Technical University
Karlovo náměstí 13, 12135 Prague, Czech Republic
sixtatom@fel.cvut.cz

Abstract. *Shape is an important feature of many object categories. In this paper we propose a Bayesian framework for detection of unknown number of objects based on their shape. The task is formulated as a minimization of Bayesian risk. The loss function is designed in such a way that the number of objects need not to be known or even bounded. We introduce a probability distribution over object states (number of objects and their poses) called Infinite Shape Mixture Model which is a modification of Rasmussen's Infinite Gaussian Mixture Model [7]. Conditional posterior distributions are derived for all parameters of the model in order to make the inference feasible. Performance of the model is tested on two brief experiments.*

1. Introduction

Object detection is crucial task in many computer vision applications. Three main approaches can be found in recent literature: appearance based methods, shape-driven and combination of both. The method introduced in this paper is shape-driven but the information about appearance can be incorporated easily. The task of detecting unknown number of objects with star convex shapes [1] is formulated as a minimisation of the Bayesian risk, i.e.

$$\phi^* = \underset{d(\mathbf{x})}{\operatorname{argmin}} \int_{\Phi} L(\psi, d(\mathbf{x}))p(\psi|\mathbf{x})d\psi, \qquad (1)$$

where $\mathbf{x}$ are the observables, $d(\mathbf{x})$ is the decision and $\psi \in \Phi$ is a vector of model parameters - number of objects and their poses.

True number of objects in the image is rarely available a priori in practice. We design the loss function and the probabilistic model in such a way it need not to be known or even bounded.

In Bayesian framework all the parameters of the model are considered random variables with their own probability distributions. Due to mathematical covenience it is common to put on them conjugate priors so their posterior belongs to such family of distributions for which a specialized sampling algorithm exists. We rather choose priors with apropriate modelling properties and generate samples from corresponding posterior by Slice sampling [4].

2. Related work

There are many papers in the literature which make use of the shape for object detection. Yu et al. [10] combine local and global shape features. They first detect so called Triple-Adjacent-Segments (TAS) on an edge image. These class independent features are then matched to a codebook and prelearned spatial relationships among TAS are used to estimate potential object center. Finally, the object category is determined by probabilistic voting.

The Fan Shape Model [3] allows to describe objects jointly by their appearance and shape. The shape is represented by lengths and angles of rays emanating from a reference point to the points on its contour. A probability distribution over rays' parameters is learned for each object class and the object detection task is solved as finding the maximum a posteriori estimate.

Bai et al. [1] propose to use the skeleton instead of contours as shape representation. The topology of a skeleton is represented by a tree: the skeleton endpoints (points with only one adjacent point) and the junction points (more than two) are nodes and the segments among them are edges. A prototypical tree is constructed for each class and the actual shape of skeleton segments in the training data (associated

[1] A set S is star convex if there exists an $x_0 \in S$, such that the line segment from x_0 to any point in S is contained in S

with each edge) is modelled by an exemplar-based mixture model. Detection itself is performed by variant of the sum-and-max algorithm.

Vo et al. [9] formulate the multi-object detection problem in a Bayesian framework by modeling the collection of states of objects as a random finite set and computing its posterior distribution given the image observation. They derive closed-form expressions of this posterior for various priors under the assumptions of non-overlapping objects. Despite that, several heuristics are needed in order to make the inference feasible.

3. Loss function

We assume that the poses of all objects are fully specified by parameters ϕ of two types: a vector of "ordinary" parameters φ (e.g. shape, position, scale) and a vector of binary switches $\mathbf{s}$ which encode, whether an object occurs in the image. We denote the vector of ordinary parameters describing the j-th object φ^j and the corresponding switch s_j. The proposed loss function is then

$$L(\psi, \phi) = \sum_{j=1}^{\infty} s_j ||\vartheta^j - \varphi^j||^2 + l(t_j - s_j)^2 \quad (2)$$

where $||.||$ is L2 norm, $\phi = \{\varphi, s\}$ is the decision, $\psi = \{\vartheta, t\}$ is a vector of true (but unknown) parameters of the model, ϑ is a vector of ordinary parameters, t_j is the true value of the j-th switch and $l \in \mathbb{R}^+$ is a positive constant. Notice that the loss function (2) is additive with respect to groups of variables. Taking the partial derivative with respect to the k-th ordinary parameter of j-th object and setting it equal to zero we obtain the optimal decision rule

$$\varphi_k^{j*} = \mathbb{E}(\vartheta_k^j | \mathbf{x}). \quad (3)$$

The optimal decision for a single switch s_j is obtained by substituting φ_k^{j*} from equation (3) into the loss function:

$$s_j = \begin{cases} 0 & \mathbb{E}(s_j|x) < \dfrac{l + \sum_k \mathrm{Var}(\vartheta_k^j|x)}{2l} \\ 1 & \text{else} \end{cases} \quad (4)$$

Decision rule (4) effectively means the higher is uncertainty (measured by variance) in values of parameters the less likely would s_j be "switched on". This is a desirable property in many applications as it allows to discard objects which are too noisy.

4. Infinite Shape Mixture Model

For clarity of explanation the model will be defined for 2D case. We assume the objects are composed of points $x \in \mathbb{R}^2$ which correspond to image pixels (although this is not necessary). If a binary segmenation would be available, all the foreground pixels could be considered data points. However, this requirement might be too restrictive so we consider all the pixels of the image as data points and define the model in such a way that only prior probabilities π_f and π_b for being assigned to foreground or background has to be provided for each pixel.

Although it is rarely true in reality we assume that the joint probability of all data points given parameters factorizes, i.e.

$$p(\mathbf{x}|\phi) = \prod_{i=1}^{n} p(x_i|\phi). \quad (5)$$

Probability of a single point x_i is

$$p(x_i|\phi) = \pi_b(x_i)\mathcal{B}(x_i) + \pi_f(x_i) \sum_{j=1}^{\infty} s_j \pi_j \mathcal{S}_j(x_i|\varphi^j) \quad (6)$$

where $\pi_b(x_i) + \pi_f(x_i) = 1$, $\pi_b(x_i)$ and $\pi_f(x_i)$ are prior probabilities for pixel x_i being assigned to background or foreground respectively, $\mathcal{B}$ represents the background component, $\mathcal{S}_j$ represents the j-th shape component and π_j are shape components weights ($\sum_{j=1}^{\infty} \pi_j = 1$).

We also introduce for each data point a latent variable c_i called indicator whose role is to encode which component has generated that point. Indicators may either attain values 1, 2, 3, ... (index of a shape component) or 'b' (background).

4.1. Background component

The background component $\mathcal{B}$ models probability that pixel x_i belongs to the background. We assume it to be uniform, i.e.

$$\mathcal{B}(x_i) = \begin{cases} \frac{1}{wh} & x_i \text{ is within image bounds} \\ 0 & \text{else} \end{cases} \quad (7)$$

where w and h are width and height of the image.

4.2. Shape components

The single shape component $\mathcal{S}_j$ in the mixture (6) is a probability distribution over object poses. Its distribution function is as follows:

$$\mathcal{S}(x_i|\phi) = \frac{1}{Z(f,T)} \exp\left(-f^2(T^{-1}(x_i))\right) \quad (8)$$

where $f : \mathbb{R}^2 \to \mathbb{R}$ is a function that describes a "template shape" of the object, $T : \mathbb{R}^2 \to \mathbb{R}^2$ is a geometric transformation and $Z(f, T)$ is the partition function

$$Z(f,T) = \int_{\mathbb{R}^2} \exp\left(-f^2(T^{-1}(x))\right) dx. \quad (9)$$

The template shape is a star convex subset of $\mathbb{R}^2$. Without loss of generality we may assume it is star convex with respect to the origin of the coordinate system. Thus f can be then specified in the following manner:

$$f(x_i) = \begin{cases} 0 & x_i = 0_{\mathbb{R}^2} \\ 1 & x_i \text{ lies on template shape boundary} \\ \frac{||x_i||}{||x_\partial||} & \text{else} \end{cases} \quad (10)$$

where x_∂ is the intersection of the line segment from $0_{\mathbb{R}^2}$ to x_i and the boundary of the template shape. Roughly speaking, such f measures distance of a point from the origin of coordinates, however it also depends on direction of the vector x_i.

The only mathematical requirement on the transformation T is existence of its inverse. We further rectrict it to be a similarity transformation consisting only of scaling, rotation and translation (in this order). In homogeneous coordinates this can be efficiently expressed as matrix multiplication:

$$T(x_i) = \begin{pmatrix} v_x \cos\beta & -v_y \sin\beta & t_x \\ v_x \sin\beta & v_y \cos\beta & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{i1} \\ x_{i2} \\ 1 \end{pmatrix} \quad (11)$$

where $x_i = (x_{i1}, x_{i2})^\top$, $(t_x, t_y)^\top$ is the translation vector, β is the angle of rotation and v_x and v_y are the scales. It can be shown that for this family of transformations the density function (8) becomes

$$\mathcal{S}(x_i|\phi) = \frac{1}{v_x v_y Z(f)} \exp\left(-f^2(T^{-1}(x_i))\right) \quad (12)$$

where $Z(f)$ now only depends on the template shape:

$$Z(f) = \int_{\mathbb{R}^2} \exp\left(-f^2(x)\right) dx. \quad (13)$$

This is important, because it allows us to define closed-form conditional posteriors for all transformation parameters.

4.3. Component parameters

In this section we will derive conditional posteriors for all the parameters of the model. We assume that the priors for components of translation vector of j-th component are uniform (index j is ommited in the formulas):

$$\begin{array}{rcl} p(t_x) & \sim & \mathcal{U}(0, w) \\ p(t_y) & \sim & \mathcal{U}(0, h) \end{array} \quad (14)$$

Rotational angle β is assumed to follow uniform distribution

$$p(\beta) \sim \mathcal{U}(0, 2\pi). \quad (15)$$

In order to ensure invertibility of T neither s_x nor s_y can be equal to zero. For many applications it is also permissible to disallow reflections (i.e. require s_x and s_y to be positive). Under such constraints a reasonable prior for the scale parameters is the log-normal distribution:

$$p(v_x) = \frac{1}{\sqrt{2\pi}\sigma_{v_x} v_x} \exp\left(-\frac{(\log v_x - \mu_{v_x})^2}{2\sigma_{v_x}^2}\right). \quad (16)$$

The prior for the v_y is from the same family with parameters μ_{v_y} and σ_{v_y}. In Bayesian settings μ_{v_x}, μ_{v_y}, σ_{v_x} and σ_{v_y} would also be considered random variables with their own priors. However we argue it is more appropriate to consider them fixed and learn them from the training data. To emphasize this we write $p(v_x)$ instead of $p(v_x|\mu_{v_x}, \sigma_{v_x})$ ($p(v_y)$ correspondingly).

Strictly speaking template shapes are also random variables and should be sampled as well. In this paper, however, we assume that all objects in the image are instances of the same template shape and we postpone this question to future work.

The conditional posterior distributions for the transformation parameters of j-th component are obtained by multiplying the likelihood (5) conditioned on the indicators by their corresponding priors:

$$p(t_{x,j}|\mathbf{x}, \mathbf{c}, T_{-t_{x,j}}) \propto \begin{cases} \prod_{i:c_i=j} \mathcal{S}(x_i|f, T^j) & t_{x,j} \in [0, w] \\ 0 & \text{else} \end{cases} \quad (17)$$

where T_{-rho} denotes vector of all transformation parameters but ρ. Posteriors for $t_{y,j}$ and β_j differ only in bounds of uniform distribution and in the fact that they are functions of $t_{y,j}$ and β_j respectively. Posterior distribution for scale $v_{x,j}$ is

$$\begin{array}{rcl} p(v_{x,j}|\mathbf{x}, \mathbf{c}, T_{-v_{x,j}}) & \propto & \ln\mathcal{N}(v_{x,j}|\mu_{v_{x,j}}, \sigma_{v_{x,j}}) \cdot \\ & & \cdot \prod_{i:c_i=j} \mathcal{S}(x, i|f, T^j) \end{array} \quad (18)$$

and $p(v_{y,j}|\mathbf{x}, \mathbf{c}, T_{-v_{y,j}})$ similarly.

We choose switches $\mathbf{s}$ to deterministically depend on the components' weights:

$$\begin{aligned} p(s_j = 0|\pi_j) &= \begin{cases} 0 & \pi_j > 0 \\ 1 & \pi_j = 0 \end{cases} \\ p(s_j = 1|\pi_j) &= \begin{cases} 0 & \pi_j = 0 \\ 1 & \pi_j > 0 \end{cases} \end{aligned} \tag{19}$$

Using this prior we need not to explicitly sample the switches during the inference as they are uniquely determined by their corresponding weights. Therefore we will replace $s_j\pi_j$ by π_j in the sequel for clarity of derivations.

Following [7] the shape component weights π are given Dirichlet prior with concentration parameter $\frac{\alpha}{k}$:

$$p(\pi|\alpha) \sim \text{Dirichlet}(\frac{\alpha}{k}, ..., \frac{\alpha}{k}) \tag{20}$$

where

$$\text{Dirichlet}(\frac{\alpha}{k}, ..., \frac{\alpha}{k}) = \frac{\Gamma(\alpha)}{\Gamma(\frac{\alpha}{k})^k} \prod_{j=1}^{k} \pi_j^{\frac{\alpha}{k}-1} \tag{21}$$

and k is number of components in the mixture. For the time being we will consider k to be finite and derive desired conditional posteriors as limiting case $k \to \infty$.

Given the weights π and unary potentials $\pi_{\mathbf{b}}$ and $\pi_{\mathbf{f}}$, the indicators $\mathbf{c}$ follow multinomial distribution:

$$p(\mathbf{c}|\pi, \pi_{\mathbf{b}}, \pi_{\mathbf{f}}) = \prod_{i:c_i=b} \pi_b(x_i) \prod_{i:c_i \in 1..k} \pi_f(x_i) \prod_{\substack{j:n_j>0 \\ j \neq b}} \pi_j^{n_j} \tag{22}$$

where n_j are occupation numbers, i.e. number of indicators whose value is equal to j. By multiplying (22) by (20) and integrating over weights π we obtain prior distribution for indicators which depends only on finite number of parameters:

$$p(\mathbf{c}|\alpha, \pi_{\mathbf{b}}, \pi_{\mathbf{f}}) = \Pi_b \Pi_f \frac{\Gamma(\alpha)}{\Gamma(n+\alpha)} \prod_{j=1}^{k} \frac{\Gamma(n_j + \frac{\alpha}{k})}{\Gamma(\frac{\alpha}{k})} \tag{23}$$

where

$$\Pi_b \Pi_f = \prod_{i:c_i=b} \pi_b(x_i) \prod_{i:c_i \in 1..k} \pi_f(x_i). \tag{24}$$

In order to make the inference feasible we need the conditional probability for a single indicator. Taking the limiting case $k \to \infty$ and fixing all the indicators but c_i we obtain

$$p(c_i = b|\alpha, \pi_{\mathbf{b}}, \pi_{\mathbf{f}}) = \pi_b(x_i) \tag{25}$$

$$p(c_i = j|\alpha, \pi_{\mathbf{b}}, \pi_{\mathbf{f}}) = \pi_f(x_i) \frac{n_{-i,j}}{n-1+\alpha} \tag{26}$$

where $j \in \{1, 2, 3, ...\}$ and $n_{-i,j}$ is the number of points, excluding x_i, which are associated with component j, i.e.

$$n_{-i,j} = \begin{cases} n_j - 1 & c_i = j \\ n_j & \text{else} \end{cases} \tag{27}$$

Finally

$$p(c_i \notin \mathcal{C}|\alpha, \pi_{\mathbf{b}}, \pi_{\mathbf{f}}) = \pi_f(x_i) \frac{\alpha}{n-1+\alpha} \tag{28}$$

where $\mathcal{C}$ denotes the set of all currently represented classes (including background). Conditional posteriors for single indicators are obtained by multiplying priors (25), (26) and (28) by their corresponding likelihood functions.

Background component:

$$p(x_i|c_i = b, \mathbf{c_{-i}}) = \begin{cases} \frac{1}{wh} & x_i \text{ is within image bounds} \\ 0 & \text{else} \end{cases} \tag{29}$$

Components for which $n_{-i,j} > 0$:

$$p(x_i|T^j, \mathbf{c_{-i}}) \propto \exp(-f^2(T^{j^{-1}}(x_i))). \tag{30}$$

All other components together:

$$p(x_i|c_i \notin \mathcal{C}, \mathbf{c_{-i}}) \propto \int p(x_i|f, T^j) p(T^j) dT^j \tag{31}$$

where

$$p(T^j) = p(t_{x,j}) p(t_{y,j}) p(\beta_j) p(v_{x,j}) p(v_{y,j}). \tag{32}$$

Integral in (31) is not tractable, so we sample from the priors in order to generate a Monte Carlo estimate of the probability of generating a new class (see [5] and [7] for details).

Finally, following [7], a vague prior of inverse Gamma shape is put on concentration parameter α:

$$p(\alpha) \propto \alpha^{-\frac{3}{2}} \exp\left(-\frac{1}{2\alpha}\right). \tag{33}$$

The conditional prior for all indicators in the limiting case $k \to \infty$ can be derived by reinterpreting equations (25), (26), and (28) to draw a sequence of

indicators, each conditioned only on the previously drawn ones:

$$p(\mathbf{c}|\alpha, \pi_\mathbf{b}, \pi_\mathbf{f}) = \Pi_b \Pi_f \frac{\alpha^{k^\dagger}\Gamma(\alpha)}{\Gamma(n+\alpha)} \tag{34}$$

where $k^\dagger$ is number of currently represented components in the model (including background). Conditional posterior distribution for α is

$$p(\alpha|\mathbf{c}, \pi_\mathbf{b}, \pi_\mathbf{f}) \propto \Pi_b \Pi_f \exp\left(-\frac{1}{2\alpha}\right) \frac{\alpha^{k^\dagger - \frac{3}{2}}\Gamma(\alpha)}{\Gamma(n+\alpha)} \tag{35}$$

5. Inference

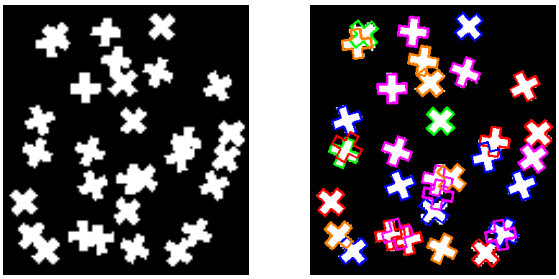
We start the inference with all datapoints assigned to one shape component. The sampling procedure is performed by standard Gibbs sampling. Due to the non-conjugate priors we sample from non-standard distributions using Slice sampling [4], for which one only has to be able to evaluate a function which is proportional to the probability density the samples are generated from.

The loss function (2) expects that the true index of each shape component is always known and it is possible to compute expected value and variances of its parameters. However, this is not true in general, it may happen during the inference that two components would switch their "positions". This issue is known as label switching problem in the literature and there has been intensive research on this topic (see [6], [8] for overview). Fortunately it turned out that in this model label switching occures rarely in practice (after burn-in period) and necessary quantities can be computed easily.

6. Experiments

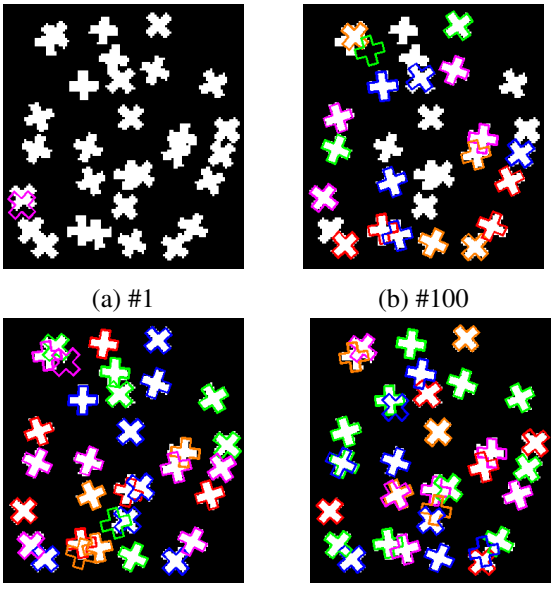
6.1. Artificial data

As a proof-of-concept we performed a simple experiment on artificial binary image 1a. The objective was to detect instances of X-like shape. There were 30 manually positioned objects in the image. Their angles were generated from the uniform distribution $\mathcal{U}(0, 2\pi)$ and scales v_x and v_y from the log-normal distribution $\ln\mathcal{N}(3.2, 0.02)$. The inference began with 200 burn-in sampling sweeps followed by 100 iterations of collecting statistics. In this experiment we did not sampled the concentration parameter α from distribution (35), but instead we set it to $\alpha = 0.1$. The concentration parameter indirectly controls number of classes in the model: the higher α the higher prior probability (28) of introducing new shape component. By setting it to a fixed value we express prior knowledge about number of components in the image.



(a) (b)

Figure 1: Artificial image (a) and detection results (b)



(a) #1 (b) #100

(c) #200 (d) #300

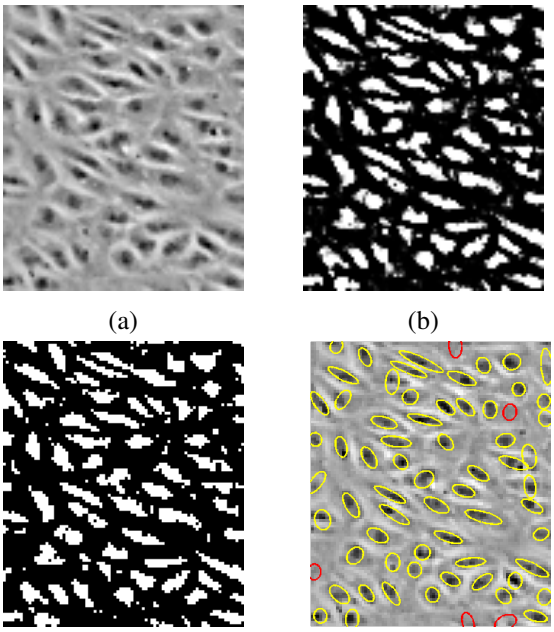
Figure 2: Samples generated during the inference (artificial image).

Evolution of the samples generated during the inference and typical detection results are shown in figures 1 and 2. All 30 objects are correctly found, however, there are also four false positives, all appearing near two or more overlapping objects. Poses of all isolated objects are estimated correctly.

6.2. Detection of cells

Second experiment is detection of endothelial cells' nuclei in a phase contrast microscopy image 3a. Nuclei are in general darker than the cytoplasm, however there are also several dark blobs which are

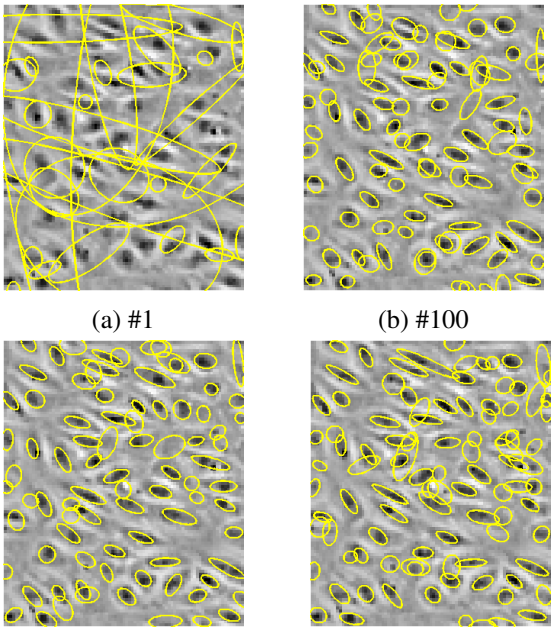
too small and they should not be detected as nuclei. In this case the binary segmentation was not available. Therefore, as a first step, we modelled the image by a mixture of two gaussians, one for background and one for foreground, and estimated their parameters by the EM algorithm. The $\mathbf{pi_b}$ and $\mathbf{pi_f}$ were values of their pdfs normalized to sum to 1 for each pixel. The template shape was unit circle. Parameters of the prior distributions for scales were estimated by maximum likelihood principle from several hand annotated nuclei: $v_x \sim \ln\mathcal{N}(2.7, 0.2)$ and $v_y \sim \ln\mathcal{N}(4.1, 0.3)$ (roughly corresponding to an ellipse with major diameter equal to 8 pixels and minor diameter 5 pixels). Translation vectors and rotation angles were assumed to a priori follow uniform distribution. Concentration parameter α was not fixed in this experiment but it was considered as random variable with posterior distribution function (35). As in the previous experiment the inference began with 200 burn-in sampling sweeps followed by 100 iterations of collecting statistics.



(a) (b) (c) (d)

Figure 3: Detection of endothelial cells. Original image (a), prior probability of foreground (black=0, white=1) (b), binary segmentation (c) and detection results (d)

Evolution of the samples generated during the inference and typical detection results are shown in figures 3 and 4. As the concentration parameter α was not fixed to a small value new detections were emerging faster during the inference when compared with



(a) #1 (b) #100 (c) #200 (d) #300

Figure 4: Samples generated during the inference (endothelial cells).

the previous experiment. Despite high variance the model was able to correctly estimate pose of most of the nuclei. However as in the previous experiment there are also several false positive detections (marked in red).

7. Conclusion

In this paper we have formulated the object detection task as a minimization of Bayesian risk. The loss function is designed in such a way that the number of objects in the image need not to be known a priori or even bounded. Further it provides a simple criterion for discarding objects whose pose is too uncertain (possibly due to the noise).

The Infinite Shape Mixture Model, a probability distribution over object states, is introduced and posterior distributions for all its parameters are derived. The inference is performed by standard Gibbs sampling. Due to mathematical convenience it is common practice to use so called conjugate priors. Instead, we focus on modelling properties of the priors and show that efficient sampling is still possible by Slice sampling, for which one only has to be able to evaluate a function which is proportional to the desired probability distribution. Performance of the model is tested on two brief experiments showing promising results.

Acknowledgements

The author was supported by the Grant Agency of the Czech Republic project P202/12/2071.

References

[1] X. Bai, X. Wang, L. J. Latecki, W. Liu, and Z. Tu. Active skeleton for non-rigid object detection. In *ICCV*, pages 575–582. IEEE, 2009. 1

[2] D. Gorur and C. E. Rasmussen. Dirichlet process gaussian mixture models: Choice of the base distribution. *Journal of Computer Science and Technology*, 25:653–664, 2010.

[3] L. J. Latecki. Fan shape model for object detection. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, pages 151–158, Washington, DC, USA, 2012. IEEE Computer Society. 1

[4] R. Neal. Slice sampling. *Annals of Statistics*, 31:705–767, 2000. 1, 5

[5] R. M. Neal. Markov chain sampling methods for dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000. 4

[6] K. Puolamaki and S. Kaski. Bayesian solutions to the label switching problem. In *Advances in Intelligent Data Analysis VIII, Proceedings of the 8th International Symposium on Intelligent Data Analysis*, pages 381–392. Springer Berlin, 2009. 5

[7] C. E. Rasmussen. The infinite gaussian mixture model. In *Advances in Neural Information Processing Systems 12*, pages 554–560. MIT Press, 2000. 1, 4

[8] M. Sperrin, T. Jaki, and E. Wit. Probabilistic relabelling strategies for the label switching problem in bayesian mixture models. *Statistics and Computing*, 20(3):357–366, July 2010. 5

[9] B.-N. Vo, B.-T. Vo, N.-T. Pham, and D. Suter. Joint detection and estimation of multiple objects from image observations. *Trans. Sig. Proc.*, 58(10):5129–5141, Oct. 2010. 2

[10] X. Yu, Y. Li, C. Fermuller, and D. Doermann. Object detection using a shape codebook. In *British Machine Vision Conference*, 2007. 1

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Detection of Curvilinear Objects in Aerial Images

Petr Zuzánek, Karel Zimmermann, Václav Hlaváč
Czech Technical University in Prague, Faculty of Electrical Engineering
Department of Cybernetics, Center for Machine Perception
Technická 2, Praha 6, Czech Republic
{zuzanp1, zimmerk, hlavac}@cmp.felk.cvut.cz

Abstract. *This paper introduces a general framework for autonomous detection of curvilinear objects in aerial images. Our contribution is two-fold. First, we designed simple yet efficient method, which sequentially prunes the space of possible curvilinear objects and thus reduces both the false negative rate detection and computational resources with respect to the exhaustive search methods. Second, our method can handle many types of curvilinear objects (e.g. roads, pipelines). We tested the method on our own dataset consisting of highway images. The produced data set is publicly available. We reached the* 93.07% *overall accuracy.*

1. Introduction

This work focuses on autonomous detection of curvilinear objects (CLO) from aerial images. We considered the scenario when the algorithm runs on the unmanned aerial vehicle (UAV). Detection of curvilinear objects, such as roads or pipelines, in aerial images is important for many UAV applications. Possible applications are, for example, monitoring of traffic situation on a highway or an autonomous navigation along a pipeline for inspection purposes.

The UAV has usually limited computational resources on board, which makes using time consuming methods, such as [12],[2],[7], [15], [13], [11] prohibitive. Computationally less consuming approaches, typically based on dynamic optimization [5], [14] usually do not achieve sufficient level of autonomy, since a user interaction is required. We propose an algorithm combining advantages of both above mentioned categories to produce a fully autonomous method for above mentioned purposes. We also provide the accuracy measure, which is not obvious for state of the art methods. Our method is designed to handle many types of curvilinear objects. We made our dataset publicly available [18].

We assume that no user interaction is allowed during the operation. This assumption becomes complicated in the situations when there are more objects in the scene and no one can decide, which is the relevant one. We assume that the relevant object is the dominant one.

The remaining part of the paper is organized as follows. In Section 2, the problem formulation is described. Section 3 provides the method overview step by step. Section 4 summarizes our experimental results. We conclude the achieved results and further work in Section 5.

2. Problem Formulation

The term CLO represents in the aerial images various number of real world objects on the Earth, such as roads, rivers, railways, forest paths, pipe lines, power lines, etc. These objects can be of various sizes, can have arbitrary orientation, position and structure.

2.1. Curvilinear Object Definition

To unify the description of several entities with different appearance only the global view has to be considered. The common properties of CLOs are said to be the following:

- The width of the object changes smoothly.
- The object curvature changes smoothly.
- The object is distinguishable from the background.
- The object is long enough, it often passes through the whole frame.
- The appearance can change non-continuously (shadows, obstacles, etc.).

2.2. Method Requirements

The requirements with respect to the UAV application are the following:

- Fully autonomous method,
- scale, rotation and position invariance,
- continuous object representation and
- real time.

3. Method Overview

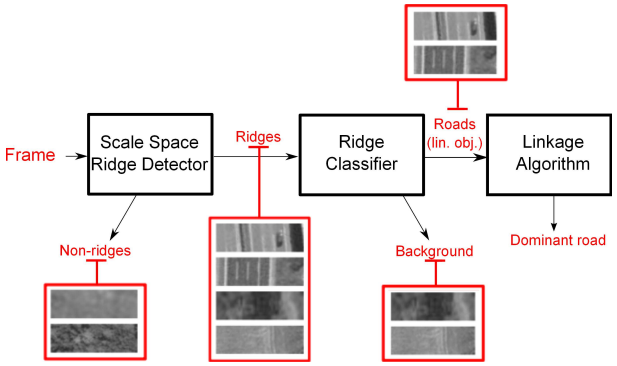


Figure 1. Overview of the proposed method. The three-stage design is depicted by black rectangles. The red rectangles demonstrate the inputs/outputs of certain stage.

The proposed method consists of three stages as depicted in Figure 1. In the first stage, the input image is preprocessed using scale-space ridge detector 3.1. The second stage performs the binary decision for ridge classification 3.2. In the third stage, the linkage algorithm 3.3 extracts the dominant curvilinear structure. The contribution of this design is that it specifies the properties of CLOs effectively and thus reduces both false negative detection rate and computational demands as compared with the exhaustive search methods.

The ridge detector 3.1 identifies pixels whose local neighbour looks curvilinear. The binary classifier 3.2 evaluates the ridge neighbourhood and propagates only those of a given object type. The linkage algorithm 3.3 considers the global properties of CLOs (defined in Section 2) and forms the dominant curvilinear structure.

3.1. Ridge Detector

The differential geometric approach is used to find both bright and dark ridges in the image intensity function (Lindeberg [9]). The detection of structures of various sizes is a crucial requirement and thus the scale-space and its notation (Lindeberg [8]) has to be introduced.

3.1.1 Scale Space Formulation

According to [9], the scale is proportional to the variance t of Gaussian kernel $g\colon \mathcal{R}^2 \times \mathcal{R}_+ \to \mathcal{R}$

$$g(\mathbf{x}; t) = \frac{1}{2\pi t} \exp^{-(\frac{\mathbf{x}^\top \mathbf{x}}{2t})} , \tag{1}$$

The original image intensity function $f\colon \mathcal{R}^2 \to \mathcal{R}$ can be transformed into scale-space with scale t using the mapping $L\colon \mathcal{R}^2 \times \mathcal{R}_+ \to \mathcal{R}$ as

$$L(\mathbf{x}; t) = g(\mathbf{x}; t) * f(\mathbf{x}) , \tag{2}$$

where $*$ denotes convolution.

Using the introduced notation, the derivation can be expressed as:

$$L_{x^\alpha y^\beta}(\cdot; t) = \partial_{x^\alpha y^\beta} L(\cdot; t) = g_{x^\alpha y^\beta}(\cdot; t) * f(.) , \tag{3}$$

where α, β denotes the order of derivatives.

3.1.2 Bright and Dark Ridge Formulation

Following the notation introduced above and formulation of bright and dark ridges in [9] [10] [6] one can express the bright ridge

$$\begin{cases} L_p = 0 , \\ L_{p^2} < 0 , \\ |L_{p^2}| \geq |L_{q^2}| , \end{cases} \tag{4}$$

and dark ridge respectively

$$\begin{cases} L_q = 0 , \\ L_{q^2} > 0 , \\ |L_{q^2}| \geq |L_{p^2}| , \end{cases} \tag{5}$$

where p and q denote the direction of the ridge and its perpendicular direction, respectively. These are obtained by the eigen analysis of the Hessian matrix of L. L_{q^2} and L_{p^2} corresponds to eigenvalues, orientations of p and q are related to orientation of eigenvectors.

We assume that any CLO can be locally described as a bright or dark ridge. Thus we can summarize that this preprocessing does not lose any object in which the user may be interested in.

The task of CLO detection has four degrees of freedom (spatial coordinates $\times$ scale $\times$ orientation). The main contribution of the scale-space ridge detector is that it uses just the 3D scale-space to determine all four parameters.

As a preprocessing, there might be relevant also another technique, such as Stroke Width Transform [3] working with parallel edges.

3.2. Ridge Classifier

The aim of the ridge classifier is to recognize the ridges belonging to a given class (type of object) as depicted in Figure 1. The ridges, whose appearance match the given class, are preserved for further processing. The rest is pruned (as depicted in Figure 1). We learnt the binary classifier from rectangular patches around the computed ridges. As depicted in Figure 2, it is sufficient to incorporate as much context as possible to be able to predict the object class successfully. We use the ridge width to determine the patch size with sufficient context.

a) b)

Figure 2. The importance of the context presented on a ridge example. An image a) consists only of the center part of an image b). In the image a) there is not enough context to determine the object class. It can be misclassified as a roof of a house, etc. On the other side, presence of context in the image b) clarifies the object class.

For the development purposes, rotationally and scale normalized samples are used as an input for the classifier both in the learning phase and the prediction process.

3.2.1 Descriptors and Features Selection

We focused at descriptors that are able to handle an object symmetry, structure and are invariant to a linear transformation of image intensity function. Its effective computation is important as well. Haar features [17] and histograms of oriented gradients (HoG) meet this requirements.

In Figure 3 there are types of Haar features used for the description.

Figure 3. Haar features type overview.

The histogram of oriented gradients uses eight equally spaced bins. The magnitude of the certain gradient is linearly interpolated into two closest bins. Let us define the ridge description

$$T_\gamma \colon \mathcal{R}^2 \times \mathcal{R}_+ \times \gamma \to \mathcal{R}^n \,, \tag{6}$$

as a mapping from ridge space to n dimensional feature space, where $\gamma \in \langle 0; \pi)$ denotes the ridge orientation.

The main contribution of these features are their effective computation (using the integral images) and scale invariance (with denoting appropriate normalization factor).

3.2.2 Classification Tool

We used Boosting algorithm Gentle Adaboost (below denoted as $H(.)$ or Gentleboost) [4] as the ridge classifier. We generated thousands of features per patch and used automatic feature selection to select the most discriminative ones.

The prediction works as follows:

$$\hat{y} = sign\left(H(\mathbf{x}) - \Theta\right) \,, \tag{7}$$

where $\Theta \in \mathcal{R}$ is a classification threshold (by default $\Theta = 0$), $\mathbf{x} \in \mathcal{R}^n$ is a feature vector and $\hat{y} \in \{\pm 1\}$ is the predicted class.

The contribution of the Gentleboost classifier are both the automatic feature selection and the cascaded prediction. It can learn any class of object and thus it allows detection of any curvilinear structure.

3.3. Linkage Algorithm

The goal of the linkage algorithm is to gather the preserved ridges into the curvilinear structure. The pseudo algorithm is described by the Algorithm 1. Linkage algorithm performs dynamic programming (DP). We choose DP because it can deal with the presence of occlusions and shadows. We slightly modified the basic DP algorithm [16] by denoting the scale change penalty and orientation change penalty. Lets compute the optimal path at position $Y = (i, j, s)$ in a scale space with predefined predecessors $\mathcal{P}_Y$ where the optimal values $F(X),\ \forall X \in \mathcal{P}_Y$ are already known. Then the $F(Y)$ is computed as:

$$\min_{X \in \mathcal{P}_Y} \left(F(X) + \alpha(X,Y)\gamma(X,Y)\mathcal{H}(Y)\right) \,, \tag{8}$$

where $\alpha(\cdot,\cdot)$ $\beta(\cdot,\cdot)$ denote scale change penalty and the ridge orientation penalty, respectively.

The linkage algorithm consists of three steps. In the first step (lines 1 - 3 of Algorithm 1), the object shape hypotheses for preserved ridges are generated. The second step (line 4 of Algorithm 1) combines these hypotheses and depicts the common parts of them. In the last step (lines 5 - 8 of Algorithm 1), these common parts are concatenated and form the dominant curvilinear structure.

Algorithm 1 Linkage Algorithm

Require: Ridges $\mathcal{Y} = \{Y\}$.
Ensure: CLO $(F^*_{L^*}, \mathbf{p}_{L^*})$.

1: Initialize $\mathcal{H}(Y)$ according to Eq. 10.
2: $(F^m, m = 1 \dots 4) \leftarrow$ Compute min-cost paths in all directions using Eq. 8.
3: $(F^*_k, \mathbf{p}_k, k = 1 \dots M) \leftarrow$ Generate hypotheses for $\mathcal{Y}$.
4: $(\mathcal{A}) \leftarrow$ Create the Accumulator of hypotheses $F^*_k, \mathbf{p}_k$ using voting schema described by Eq. 12.
5: $Y^* = \arg\min_{ijs}\{\mathcal{A}\}$.
6: Reinitialize $\mathcal{H}(Y)$ according to Eq. 13.
7: $(F^m, m = 1 \dots 4) \leftarrow$ Compute min-cost paths in all directions using Eq. 8.
8: $(F^*_{L^*}, \mathbf{p}_{L^*}) \leftarrow$ Generate hypothesis for Y^*.

3.3.1 Hypotheses Generation

We use the assumption that the CLO can pass through any preserved ridge. Using this assumption and proposed DP algorithm, we generate the min-cost path for a given amount of preserved ridges. Figure 4 depicts the recognized ridges. Generated

Figure 4. The ridges preserved by the classifier (green).

min-cost paths are said to be a hypotheses. Each hypothesis k with length N_k can be formalized as a list of coordinate triplets

$$\mathbf{p}_k = \langle\langle x_{k1}, y_{k1}, s_{k1}\rangle, \dots \langle x_{kN_k}, y_{kN_k}, s_{kN_k}\rangle\rangle\,, \quad (9)$$

i.e. path in 3D space. DP algorithm define the function $\mathcal{H}(Y)$) from Equation 8 as:

$$\mathcal{H}(Y) = \begin{cases} -\operatorname{sgn}(H(T\gamma(Y))) & Y \text{ is a ridge}\,, \\ 0 & \text{otherwise}\,, \end{cases} \quad (10)$$

The DP algorithm, starting at each image edge and terminating at the opposite edge, provides for any point Y in the scale-space the four min-cost paths $F^m(Y)\,, m = 1 \dots 4$ in four directions. For the sake of simplicity, the arguments are omitted in the rest of the paper. The hypothesis k consists of a concatenation of the two cheapest min-cost paths F^m_k, F^n_k and its support F^*_k fulfils

$$F^*_k = \min_{m,\, n=m+1}(F^m_k + F^n_k)\,,\ m = 1 \dots 3\,. \quad (11)$$

3.3.2 Hypotheses Combination

Figure 5. The accumulator of hypotheses uncovers the final dominant curvilinear structure. The weak hypotheses disappeared due to scaling factor.

We use an accumulator $\mathcal{A}$ for hypotheses combination. The voting procedure for triplet s in hypothesis $\mathbf{p}_k$ works as follows

$$\mathcal{A}(x_{ks}, y_{ks}, s_{ks}) = \mathcal{A}(x_{ks}, y_{ks}, s_{ks}) + F^*_k/N_k. \quad (12)$$

Each hypothesis $\mathbf{p}_k$ votes in particular positions where it passes through. The strength of vote reflect its support proportionally. The accumulator highlights the frequently passed positions as depicted in Figure 5.

3.3.3 Dominant Linear Structure Extraction

Figure 6. The extracted CLO (green). The width of extracted structure corresponds to the ridge scale (red).

The final dominant curvilinear structure is extracted as a hypothesis that corresponds to the position with the highest support in $\mathcal{A}$. We use DP with

function $\mathcal{H}(Y)$ from Equation 8 formulated as

$$\mathcal{H}(Y) = \mathcal{A}(i, j, s) \,. \tag{13}$$

The extracted dominant linear structure L^* is depicted in Figure 6. The $F^*_{L^*}$ indicates the quality of solution.

The main contributions of joint accumulator and DP are their high robustness to false negative output of the ridge classifier and non uniform distribution of positively classified ridges along the object. Both these scenarios can be affected by occlusions or shadows.

4. Experiments

We focused our experiments on highway extraction as a well defined concept which is easy to obtain. We created own dataset from the Google satellite images [1].

4.1. Classifier Learning

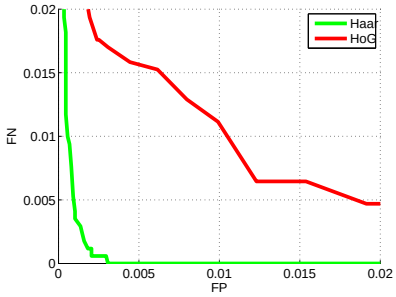


Figure 7. False detection ROC curve with parameter $\Theta \in \langle -10; 10 \rangle$ (classifier threshold) for both Haar and HoG descriptors on the test data.

We learnt the Gentleboost classifier using $6 \cdot 10^3$ positive and more than $30 \cdot 10^3$ negative samples (we used sample split 70% for learning, 30% testing). The samples were gathered approximately along 10 km of highway. Each sample was described either by 10^4 Haar features or $8.2 \cdot 10^3$ HoG features. Haar features were generated randomly, HoG features were generated hierarchically. Learning phase selected 200 features and formed the strong classifier. Figure 7 shows the ROC curve of false detection rate for different values of classification threshold Θ. The Haar descriptor achieves higher performance on test data and is used for further experiments. We believe that main limitation of the HoG discriminative power is the design of kind of hierarchical feature generation. Figure 8 shows the most discriminative features on the test samples. The formed strong classifier generated 0.00% error on training data and 0.13% on test data.

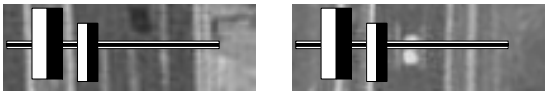


Figure 8. The most discriminative features on the positive test samples.

4.2. Highway Extraction

We tested the algorithm on approximately 500 positive images ($\mathcal{P}$) (approx. 110 km of highway) and around 400 negative images ($\mathcal{N}$) with neither highways nor other roads. The verification result considers both quality and annotation. Let the R_{L^*} depicts the ratio of extracted path which corresponds to given annotation.

We defined the quality threshold Θ_F and ratio threshold Θ_R and decision strategy $\mathcal{D}(x, \Theta)$

$$\mathcal{D}(x; \Theta) = \begin{cases} 1 & x \geq \Theta \,, \\ 0 & otherwise \,. \end{cases} \tag{14}$$

Table 1 describes the evaluation principle. Figure 9 shows an ideal result that the whole extracted structure corresponds to the annotation (i.e. lies within the annotation).

Image	$\mathcal{D}(-F_{L^*}, \Theta_F)$	$\mathcal{D}(R_{L^*}, \Theta_R)$	**decision**
$\mathcal{P}$	1	1	TP
$\mathcal{P}$	1	0	FP
$\mathcal{P}$	0	0	FN
$\mathcal{P}$	0	1	FN
$\mathcal{N}$	1	0	FP
$\mathcal{N}$	1	1	–
$\mathcal{N}$	0	0	TN
$\mathcal{N}$	0	1	–

Table 1. Evaluation schema for extracted object L^*. The decisions T(F) P denotes true (false) positive and analogically for true (false) negative. The $-$ stands for infeasible configuration.

The threshold $\Theta_F = 110$ was chosen experimentally so it minimizes the error on test data for fixed $\Theta_R = 0.1$. The Θ_R is not really strict because the algorithm can not classify the ridges near to the image boundaries. We described this unreachable area as a margin on the image boundaries. Assuming arbitrary ridge orientation, width margin is defined by the radius of outer circle of a certain patch. Since the

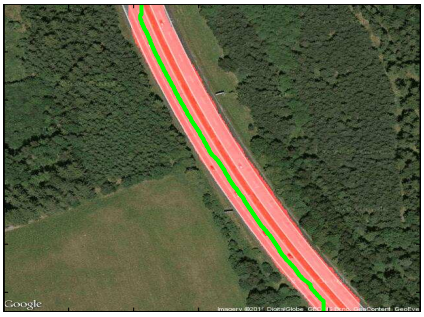


Figure 9. The extracted structure L^* (green) with annotation (red).

classification is not possible, the linkage algorithm can behave abruptly at the same place. The example of proposed behaviour demonstrates Figure 10. Established scale-space consists of 40 layers and was

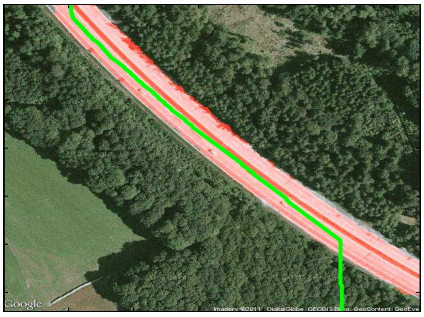


Figure 10. The extracted structure L^* (green) with annotation (red) with abrupt behaviour on the image boundaries.

able to detect the structure $4 - 40$ pixels wide. The scale change penalty $\alpha(\cdot, \cdot)$ was set to 0.0 for no change, 0.3 for a one level change and ∞ otherwise. The orientation change penalty $\beta(\cdot, \cdot)$ denotes the absolute value of cosine distance between ridge directions. The number of hypotheses for the accumulator was chosen experimentally to 100. Using the proposed methodology we achieved the 93.07% of correct classification on our test data.

We studied the influence of the ridge classifier accuracy on the global results. We simulated this by reducing the number of weak classifiers. Table 2 provides the results.

Although the accuracy for small number of weak classifiers is not precise, still the classifier presence play an important role in overall accuracy.

In our test data consisting of 640×640 px^2 images there are approximately $3 \cdot 10^5 - 5 \cdot 10^5$ detected ridges using the scale space ridge detector. Without the preprocessing step, this number increases rapidly. Assuming the discrete orientation with precision 1 degree, then the ridge classifier have to classify $\sim 2.95 \cdot 10^9$ patches. We did not study the

No. of weak cl	Trn error	Tst error	**Accuracy**
$0(+)$	$-$	$-$	34.67
2	7.93%	10.80%	80.48%
4	5.47%	4.90%	85.48%
8	3.38%	2.47%	85.95%
200	0.00%	0.13%	93.07%

Table 2. The accuracy of the method with respect to the accuracy of the classifier. Trn / Tst stands for error on the training and testing data respectively. The symbol $(+)$ means that all classifier inputs invoke a positive output. The $-$ is used for undefined numbers.

real-time implementation of scale-space ridge detector, but we believe it still outperforms the exhaustive search.

The results also approved the importance of linkage algorithm and its ability to deal with the either inaccuracy of the first two stages or presence of occlusions and shadows.

We developed a prototype of the above described concept, which is far from the real application and thus no relevant run time information is available yet. The most time consuming part of the pipeline is the ridge classification 3.2 which requires the rotationally and scale normalized samples. The most memory consuming part is the scale-space establishment 3.1.

5. Conclusion

We presented an autonomous method for the detection of CLO in aerial images. We sequentially prune the space of all curvilinear objects hypotheses and thus make the method convenient for the UAV with limited resources. The objective of this early experiments proposed in this paper was to confirm the validity of the method design. We proved the method validity by introducing the accuracy measure. Using the best configuration we achieved overall accuracy 93.07%.

The future work has to be focused both on deeper testing of proposed method (considering more types of objects in the scene, other object types, etc.) and run time optimisation. The possibilities of real-time ridge detector and ridge classification has to be studied intensively. The bottlenecks of these stages are scale-space establishment and sample normalization for classification. The most suitable way how to avoid the scale normalization is to use the scalable features. The rotated features are not suitable for this

application, since there exist no fast approaches for computation of integral image rotated by arbitrary angle. The better way seems to be a learning of several classifiers, each one used for fixed orientation.

6. Acknowledgement

The first author is supported by the Technology Agency of the Czech Republic under the project TA01020197. The second and third author are supported by the European Commission under the project FP7-ICT-247870.

References

[1] An online Google API tutorial. `https://developers.google.com/maps/`. 5

[2] D. B. L. Bong, K. C. Lai, and A. Joseph. Automatic road network recognition and extraction for urban planning. In *World Academy of Science, Engineering and Technology 53*, pages 209–215, 2009. 1

[3] B. Epshtein, E. Ofek, and Y. Wexler. Detecting text in natural scenes with stroke width transform. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2963 –2970, june 2010. 2

[4] J. Friedman, T. Hastie, and R. Tibshirani. Additive Logistic Regression: a Statistical View of Boosting. *The Annals of Statistics*, 38(2), 2000. 3

[5] A. Gruen and H. Li. Semi-Automatic Linear Feature Extraction by Dynamic Programming and LSB-Snakes. *Photogrammetric Engineering and Remote Sensing*, 63(8):985–995, August 1997. 1

[6] R. M. Haralick. Ridges and valleys on digital images. *Computer Vision, Graphics, and Image Processing*, 22(1):28 – 38, 1983. 2

[7] I. Laptev. Road extraction based on snakes and sophisticated line extraction. In *Master's thesis, Computational Vision and Active Perception Lab (CVAP), Royal Institute of Technology*, 1997. 1

[8] T. Lindeberg. *Scale-Space Theory in Computer Vision*. Kluwer Academic Publishers, Norwell, MA, USA, 1994. 2

[9] T. Lindeberg. Edge detection and ridge detection with automatic scale selection. In *Proceedings of the 1996 Conference on Computer Vision and Pattern Recognition*, CVPR '96, pages 465–, Washington, DC, USA, 1996. IEEE Computer Society. 2

[10] T. Lindeberg. *Wiley Encyclopedia of Computer Science and Engineering*, chapter Scale-Space, pages 2495–2504. 2008. 2

[11] H. Mayer, I. Laptev, and A. Baumgartner. Multi-Scale and Snakes for Automatic Road Extraction. In S.-V. B. H. 1998, editor, *European Conference on Computer Vision*, volume 1, pages 720–733, 1998. 1

[12] V. Mnih and G. E. Hinton. Learning to detect roads in high-resolution aerial images. In *Proceedings of the 11th European conference on Computer vision: Part VI*, ECCV'10, pages 210–223, Berlin, Heidelberg, 2010. Springer-Verlag. 1

[13] X. Niu. A semi-automatic framework for highway extraction and vehicle detection based on a geometric deformable model. *Journal of Photogrammetry and Remote Sensing*, pages 170–186, March 2006. 1

[14] S.-R. Park and T. Kim. Semi-Automatic Road Extraction Algorithm from IKONOS Images Using Template Matching. *In Proceedings of 22nd Asian Conference on Remote Sensing, Singapore*, pages 1209–1213, November 2001. 1

[15] V. Shukla, R. Chandrakanth, and R. Ramachandran. Semi-Automatic Road Extraction Algorithm for High Resolution Images Using Path following Approach. In *Indian Conference on Computer Vision, Graphics and Image Processing*, pages 231–236, December 2002. 1

[16] T. Svoboda, J. Kybic, and V. Hlavac. Image processing, analysis and machine vision: A matlab companion. 2008. 3

[17] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, 1:I–511–I–518 vol.1, 2001. 3

[18] P. Zuzánek, K. Zimmermann, and V. Hlaváč. The dataset of Google sattelite images for curvilinear objects detection. `http://cmp.felk.cvut.cz/~zuzanp1/`, December 2012. 1

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Simultaneous Segmentation and Recognition of Graphical Symbols using a Composite Descriptor

Martin Bresler, Daniel Průša, Václav Hlaváč
Czech Technical University in Prague, Faculty of Electrical Engineering
Department of Cybernetics, Center for Machine Perception
166 27, Praha 6, Technická 2, Czech Republic
{breslmar, prusapa1, hlavac}@cmp.felk.cvut.cz

Abstract. *This work deals with recognition of hand-drawn graphical symbols in diagrams. We present two contributions. First, we designed a new composite descriptor expressing overall appearance of symbols. We achieved rather favorable accuracy in classification of segmented symbols on benchmark databases, which is 98.93% for a database of flow charts, 98.33% for a database of crisis management icons, and 92.94% for a database of digits. Second, we used the descriptor in the task of simultaneous segmentation and recognition of graphical symbols. Our method creates symbol candidates by grouping spatially close strokes. Symbol candidates are classified by a multiclass SVM classifier learned on a dataset with negative examples. Thus, some portion of the candidates is filtered out. The joint segmentation and classification was tested on diagrams from the flowchart database. We were able to find 91.85% of symbols while generating 8.8 times more symbol candidates than is the number of true symbols per diagram in average.*

1. Introduction

The proliferation of tablets or tablet PCs implies the demand for algorithm allowing interface by handwriting or hand-drawing. The attention of researches is recently given on graphical representation of human thoughts, e.g. diagrams. One of the most common diagram for various branches is a flowchart. It is used to describe a general algorithm or a process. The flowchart is composed of boxes connected by arrows and text, which can be inside the boxes or can label the arrows. See the example of flowchart in Figure 1. To recognize a flowchart fully, we have to recognize all symbols correctly, find relations between them, and also recognize a text.

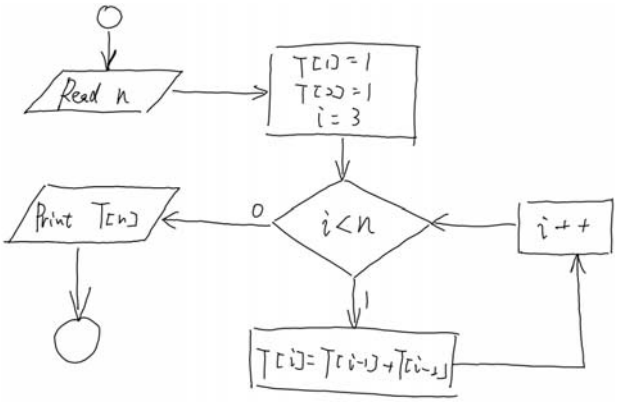


Figure 1. Example of a flowchart from the FC database.

Each diagram recognizer must perform following six stages [4]: (1) early processing - noise reduction, de-skewing, etc., (2) segmentation of strokes into isolated symbols, (3) symbol recognition, (4) identification of spatial relationships among symbol, (5) identification of logical relationships among symbols, (6) semantic interpretation. This paper describes our approach to perform steps (2) and (3) of the diagram recognition pipeline. We allow multiple candidates for the symbols to be created. The segmentation phase thus does not make a final decision. This is left at the next stages performing structural analysis. We follow the structural construction paradigm proposed by Schlesinger and Hlaváč [13]. We chose flowcharts as diagrams of our interest and we used FC database [1] for training and verification. It contains 327 diagrams drawn by 35 users and there are 4 780 symbols. The database is divided into a training dataset (200 diagrams, 2 919 symbols) and a test dataset (127 diagrams, 1 861 symbols). The diagrams are stored in inkml file format, where are in-

dividual strokes and symbols defined. We recognize just graphical symbols and text must be processed in different way. Therefore, the text is filtered out from the diagrams. In practice, this can be done using various text / non-text stroke classifiers. The state-of-the-art methods work with a high accuracy [3], [17], [7], [11].

Symbols segmentation was researched by many people. For example, Kara and Stahovich [8] presented an approach where arrows are detected first and they are stated immediately as a ground truth. Then the rest represents separated symbols. Peterson et al. [12] came with two-step stroke grouping based on a single stroke classification followed by clustering of strokes within the classes. Although the results are promising, there is still much work to do to solve the problem robustly. There exist algorithms for graphical symbols recognition, which classify already segmented symbols. These algorithms are often based on HMM or MRF [2], [18]. Some algorithms work with more complex graphs representations [9]. In some cases, it is beneficial to combine more approaches together to achieve higher accuracy [14]. The results vary with respect to the difficulty of the problem, i.e. how complex the symbols are, what is the quality of databases etc. We present an approach, in which the segmentation and classification are performed simultaneously. The segmentation is not final and we rather generate symbol candidates for the next stage of the recognition pipeline. Therefore, we call it pre-segmentation. The approach is simultaneous in the sense that the pre-segmentation is a result of classification of selected groups of strokes. This is the main advantage in comparison with the state-of-the-art methods. They do hard decisions in symbol segmentation step or rather focus on recognition of already segmented symbols.

The rest of the paper is organized as follows. The method of segmentation by classification is described in Section 2. Our symbol descriptor is defined in Section 3. An evaluation of experiments on the FC database and experiments on additional databases are given in Section 4 to show the generality of the descriptor. Finally, our conclusions are presented in Section 5.

2. Pre-segmentation by Classification

This section describes our approach how to group strokes of a flowchart into isolated symbols and how to classify them. The main idea is that we create symbol candidates by grouping single strokes together followed by classification of all the candidates. We assume there is no stroke which is common to two or more symbols. This assumption holds in most of the cases. Our goal is to generate as few candidates as possible and miss as few true symbols as possible at the same time. The generated symbol candidates are supposed to be filtered in the next stages of the pipeline using information about relations between them.

2.1. Strokes Grouping

First, we take all single strokes as a symbol candidates of size 1. Then, we create iteratively new symbol candidates of size n by adding a single, spatially close, stroke to symbol candidate of size $n-1$. It can be seen in the histogram of true symbol sizes in the FC database (see Figure 2), that it is sufficient to create symbol candidates of maximal size 5. Two strokes (or a stroke and a group of strokes) are spatially close if the distance between two closest points is below a threshold, which is defined as $distThresh = k \cdot D_{med}$, where D_{med} is a median of lengths of diagonals of bounding boxes of all single strokes in a diagram. The usage of the D_{med} makes the $distThresh$ independent of the overall size of the diagram, which differs due to the different writer's conventions or different resolution of used devices. We chose the value of k to be 0.35 empirically (see Figure 3).

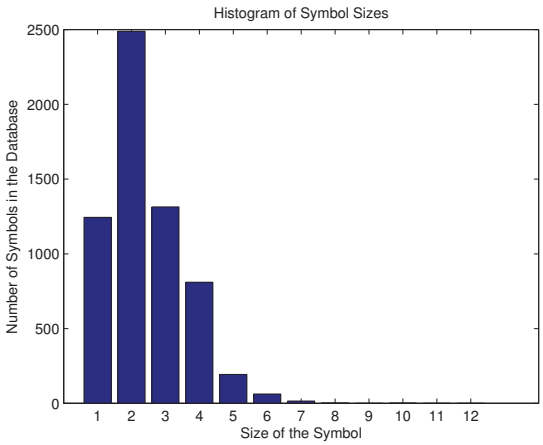
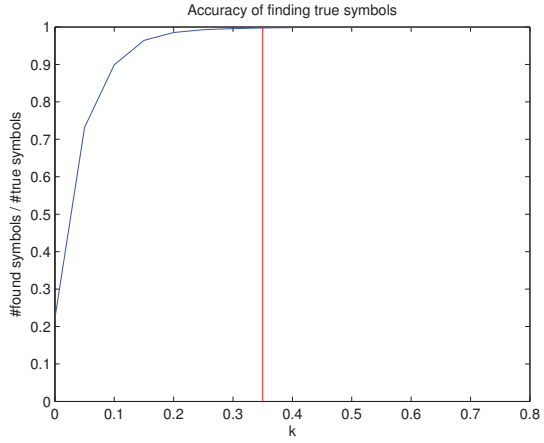


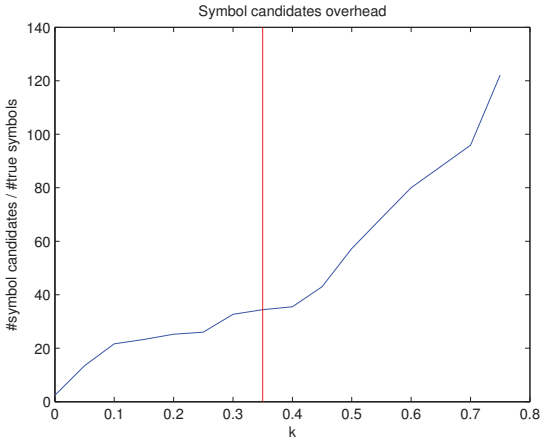
Figure 2. Histogram of Symbol Sizes.

2.2. Classification

The classification of symbol candidates is based on a descriptor which is described in Section 3. We use a multiclass classifier implemented as an instance



(a)



(b)

Figure 3. Result of the experiments on the whole FC database to obtain optimal value of the k coefficient. (a) shows the growth of the accuracy with growing k and (b) shows the growth of the number of symbol candidates. The choice of optimal k is a tradeoff.

of a structured output SVM learned by BMRM algorithm [16]. A logistic regression is fitted on the classifier response to obtain a posterior probability that a symbol candidate belongs to the class. We used Statistical Pattern Recognition Toolbox for Matlab (STPRtool) [5] for this.

We have seven basic classes in our training data: *arrow*, *connection*, *data*, *decision*, *process*, *terminator*, and *no_match*. Examples of objects from these classes are shown in Figure 4. The last class represents symbol candidates with no meaning. It is very important to prevent false positive detections. The training data is extracted from the training diagrams of the FC database. We find all symbol candidates of maximal size 3 in a way described above. The labels are taken from the annotated database for symbol candidates which represent true symbols. The rest of the candidates has no meaning and thus gets the label *no_match*. In total, we have 32 064 symbols in the training dataset, where are 1 474 examples of class *arrow*, 143 of *connection*, 337 of *data*, 248 of *decision*, 477 of *process*, 240 of *terminator*, and 29 145 *no_match*.

Symbols of the same class may look very differently yielding different descriptors. It is the most common for classes *arrow* and *no_match* (see Figure 4). Therefore, we cluster symbols of these classes into several clusters according to their descriptors. We use 10 clusters for arrows, 30 clusters for the class *no_match* and two clusters for the rest. We define a loss function which gives higher penalty when a symbol is classified as *no_match*. Specific values of the loss function are shown in Table 1. Classification into a different cluster of the same class does not count as an error. Thus, the table defining the loss function is simplified.

	arr.	conn.	data	dec.	proc.	term.	no_m.
arr.	0	2	2	2	2	2	1
conn.	2	0	2	2	2	2	1
data	2	2	0	2	2	2	1
dec.	2	2	2	0	2	2	1
proc.	2	2	2	2	0	2	1
term.	2	2	2	2	2	0	1
no_m.	100	100	100	100	100	100	0

Table 1. Definition of the loss function. Each column represents a true class.

We used the cross-validation with 5 folders to obtain the optimal value of the regularization constant for the training. The data was divided into five folders w.r.t. their classes and four folders was used for training and one for testing. We chose the constant to be 10^{-7} according to the cross-validation test error. Then we used the constant and learned the classifier on the whole train dataset. Results of the classifier on test diagrams of the FC database are shown in Section 4.

3. Descriptor

As we have shown in Section 1, there are many ways how to describe a hand-drawn graphical symbol. Flowcharts are composed of very simple and geometrically describable symbols such as rectangles, circles, etc. See Figure 1. Those symbols, of course, may change the size arbitrarily. Moreover, they can be drawn with a different number of strokes in arbitrary order. Therefore, a descriptor which requires an

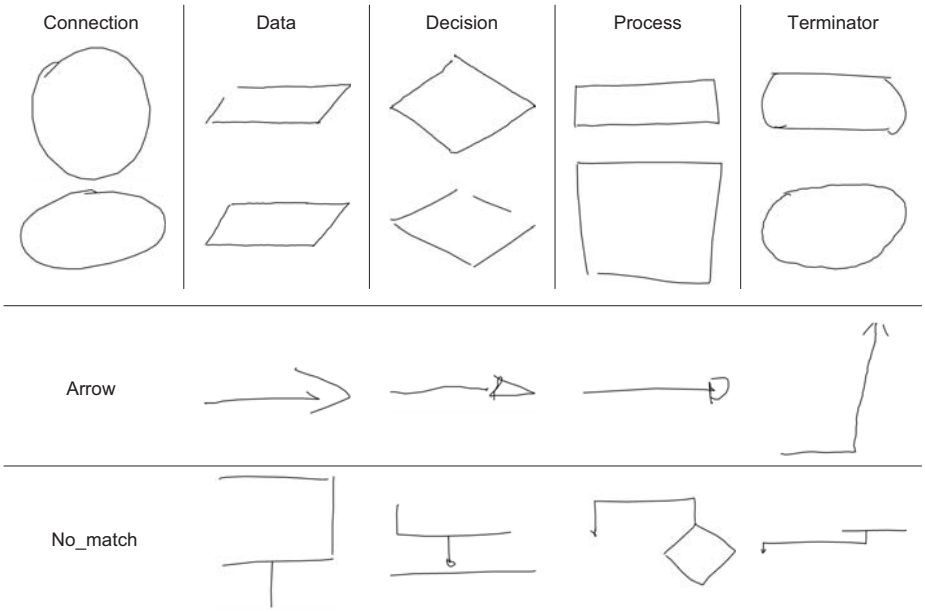


Figure 4. Examples of symbols from each of the classes taken from the database.

exact order of strokes would imply unnatural requirements on the user. For the purpose of recognizing graphical symbols contained in a flowchart, we proposed a new descriptor. Although, it is designed for flowcharts it can be used to describe arbitrary graphical symbol.

3.1. Descriptor Components

Our descriptor is composed of three components. The first one is the normalized histogram of distances between points. The second one is the normalized histogram of angles given by three points, and the last component is the histogram of small substrokes (compositions). Each of those components does not have power to fully describe the overall appearance of a given symbol. However, their combination showed to be discriminative enough. The dimension of the whole descriptor is 90 which is a concatenation of its three components with dimension 32, 16, and 42, respectively.

As we mentioned before, a symbol may be composed of many strokes. Therefore, we take points of all strokes and ignore gaps between endpoints. Let us denote the sequence of points representing a symbol $P = \{p_1, p_2, \ldots, p_n\}$.

3.1.1 Histogram of Distances Component

The distance between each possible pair of points on x-axis (Δx) and y-axis (Δy) is computed and these values are assigned to corresponding histograms. This descriptor component is composed of four histograms with 8 bins each, where two histograms are allocated for Δx and two histograms for Δy. The Algorithm 1 shows how to assign Δx to its two corresponding histograms. The approach is analogous for Δy. All histograms are finally normalized by the number of pairs of points $\binom{n}{2}$.

Data: P, Δx, $lowHist$, $highHist$
Result: Increment certain bin in either $lowHist$ or $highHist$
$lowThresh = \max_{i,j,i\neq j,j-i=1} |p_i.x - p_j.x|$;
$highThresh = BoundingBox(P).width$;
if $\Delta x < lowThresh$ **then**
  $binSize = lowThresh/8$;
  $bin = \Delta x / binSize$;
  $lowHist[bin] = lowHist[bin] + 1$;
else
  $binSize = highThresh/8$;
  $bin = \Delta x / binSize$;
  if $bin > 7$ **then**
    $bin = 7$;
  end
  $highHist[bin] = highHist[bin] + 1$;
end

Algorithm 1: Assignment of a Δx value to the certain histogram.

3.1.2 Histogram of Angles Component

The second component expresses the curvature of the symbol. Triplets of points p_i, p_j, p_k are taken such that $j - i > 1, k = \lfloor (i + j)/2 \rfloor$. For all these triplets we compute the angle

$$\alpha = \arccos \frac{\overrightarrow{p_j p_k} \cdot \overrightarrow{p_k p_i}}{\|\overrightarrow{p_j p_k}\| \cdot \|\overrightarrow{p_k p_i}\|}. \quad (1)$$

Values are mapped to 16-bin histogram the standard way. The size of the bins is $\frac{\pi}{16}$. The histogram is normalized by the number of angles $\binom{n-1}{3}$.

3.1.3 Histogram of Compositions Component

The last component is inspired by the work of Tabernik et al. [15], in which the histogram of compositions (HoC) was presented as a low-level image descriptor. We used the similar principle in building a low-level stroke descriptor. First, angles between vectors given by two consecutive points and x-axis are computed. These angles are components of the first layer. All pairs of consecutive components of the previous layer are combined to create new compositions of the next layer. The compositions carry only information about the angles. The first two layers are visualized in Figure 5. Empirically, we chose to quantize the first level components to six values in the first layer and to use two layers. Higher number of both, components of the first layer and number of layers, leads to the lower accuracy because the descriptor would be too specific and less tolerant to different styles of drawing. Therefore, this component of the descriptor has dimension 42 since it contains the 6-bin histogram of first layer components and the 36-bin histogram of second layer components. When the histograms are computed, each angle is linearly interpolated into two neighbouring bins. Therefore, each first layer component contributes to two bins of the corresponding histogram and each second layer composition contributes to four bins. Both histograms are normalized to sum into 1.

It might be computationally expensive to compute the descriptor since all possible pairs of points are considered in the first two components. In the situation, in which the symbols are more complex (consist of higher number of points), it is possible to skip some pairs. We were able to use only 50% of possible pairs (leads to double speedup) without loss of the accuracy in our experiments.

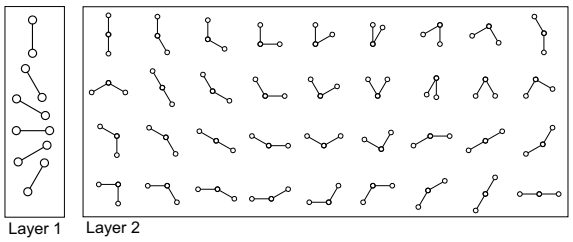


Figure 5. Visualization of compositions in two layers.

4. Experiments

We present our results achieved with the descriptor and the SVM classifier. First, the results of experiments with segmentation and classification of entire diagrams in FC database are presented. Later, we show results of experiments with the classification of segmented symbols from various databases, which demonstrate the universality of our descriptor.

We also measured how fast the computation of our descriptor is for the segmented symbols from the FC database. The symbols consist of 76.32 points in average and the computation time was in average 1.83 ms per symbol. The code is implemented in C#. We ran it on a standard tablet PC Lenovo X61 (Intel Core 2 Duo 1.6 GHz, 2 GB RAM) with 64-bit Windows 7 operating system.

4.1. Recognizing Diagrams

We tested our learned classifier on the testing diagrams of the FC database. The goal was to find as many true symbols as possible while keeping the number of symbol candidates as low as possible. The database contains 127 testing diagrams with 14.7 symbols in average. The strokes grouping algorithm is able to find 99.7% of the symbols in average while generating 368.9 symbol candidates in average. All symbols candidates were classified and three possibilities with the highest posterior probability were taken for each symbol candidate forming the set of classification results. Classification results with label *no_match* were removed. Results with posterior probability lower than 0.001 were removed as well. A possible post-processing step is to set a maximal number of classification results for each stroke and remove excessive ones with the lowest probability. The results can be found in the Table 2. Without removing excessive classification results, we can obtain accuracy 91.9% with 129.2 classification results in average. How many symbols from each class were

not recognized shows Table 3. The class arrow has the highest number of unrecognised symbols. It is because an arrow can have different shapes (different head, direction, etc.) and thus the recognition is more difficult. Achieved number of classification results is acceptable for the next step of the pipeline.

max. #candidates / stroke	#candidates	accuracy
5	48.31	86.20%
8	66.25	88.65%
10	75.54	89.45%
12	83.30	89.89%
15	92.97	90.40%
∞	129.17	91.85%

Table 2. Dependence of the average number of generated symbol candidates and the accuracy on the maximal number of symbol candidates per stroke.

arrow	connection	data	descision	process	terminator
72	0	14	18	24	32

Table 3. Numbers of unrecognised symbols from each class (out of all 1 861 symbols).

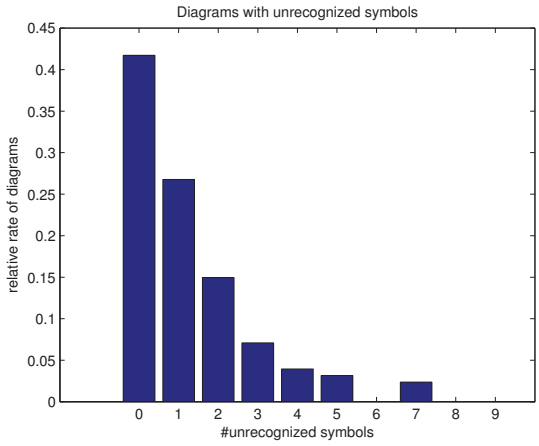


Figure 6. Histogram of numbers of unrecognised symbols in the diagrams.

Although the accuracy is not very high, it can be seen in Figure 6 that there are many diagrams (around 45%) where the accuracy 100% was reached. Those diagrams were drawn by users with nice drawing style, which is easily readable. On the other hand, diagrams with very low accuracy are sometimes very hardly readable even by human. In some cases we are facing bad annotation too. Therefore, we consider the results encouraging. Moreover, we expect that the recognition system will be providing a tool for a quick correction of misclassifications by the user. While the number of misclassifications is low (one or two) the system should be still effectively usable.

4.2. Classification of Segmented Symbols

We tested our descriptor on different databases of hand-drawn graphical symbols to get an idea how discriminative it is and for what kind of symbols it can be used. The symbols were segmented. In all cases, we used SVM classifier trained by BMRM on training dataset of the database and tested the classifier on the test dataset. We used cross-validation with 5 folders on each training dataset to obtain the optimal value of the regularization constant. We used zero-one loss function in all cases.

First, we evaluated the descriptor on the segmented symbols of the FC database. There are 6 classes and there are 2 919 symbols in the train dataset and 1 861 in the test dataset. Since the symbols are already segmented, we do not have to define the class *no_match*. The accuracy was 98.9%.

The second database, on which we tested the descriptor, was NicIcon database of handwritten icons for crisis management by Niels et al. [10], which consists of 15 372 symbols (9 212 in the train dataset and 6 160 in the test dataset) of 14 classes (see Figure 7). We performed the writer independent experiment and obtained the accuracy 98.3%. It is way better than the result of Niels et al., which is 96.5%.

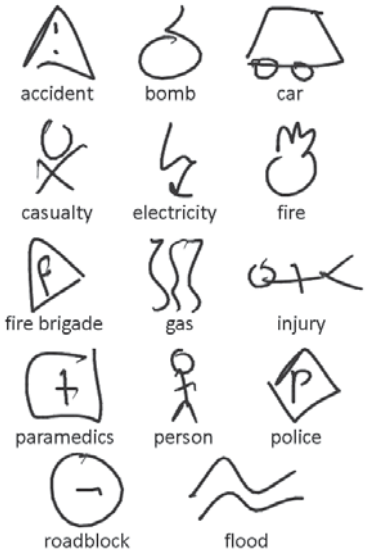


Figure 7. Symbol classes in the NicIcon database.

The last database was Unipen database of handwritten digits [6] which contains 4 990 digits (3 489 in the train dataset and 1 501 in the test dataset). In this experiment we obtained the accuracy 92.9%

which is significantly lower than in previous cases. The main reason probably is the fact that our descriptor is based on histograms and digits are very simple symbols consisting of only few points. Therefore, the histograms in our descriptor are rather dense.

5. Conclusion

We presented a new composition descriptor for describing hand-drawn graphical symbols. We trained the multiclass SVM classifier on various databases of graphical symbols, where the symbols were described with our descriptor. The result is outstanding for databases of more complex symbols. We achieved the accuracy 98.9% and 98.3% for FC database and NicIcon database, respectively. Experiments also showed that a poorer result is achieved on symbols consisting of just few points, which is the case of Unipen database of digits, where we achieved the accuracy 92.9%.

We used the descriptor to recognize symbols of flowcharts among symbol candidates, which are generated by strokes grouping. We generated the symbol candidates by grouping spatially close strokes together. This led to a segmentation, which contains a lot of candidates representing no symbol. Therefore, we learned the multiclass SVM classifier with negative examples. Moreover, we showed how the clustering of descriptors from the same class leads to better results. We were able to find 91.9% of the symbols in the FC database while we generate in average 8.8 times more symbol candidates than the number of symbols in a diagram.

In the future work, we plan to use the output of the segmentation as an input for the next stage of the pipeline, where relations between symbol candidates will be found. The goal will be to keep only the symbol candidates, which form a correct flowchart. Preliminary results based on modelling of relations and following optimization are promising and show that the number of candidates is not too large. We also plan to test the descriptor on more databases and possibly improve it to be more specialized on flowcharts, especially on filtering of the negative examples.

Acknowledgements

The authors were supported by the Grant Agency of the Czech Republic under the project P103/10/0783. The authors would like to thank V. Franc for his valuable advices in the field of machine learning.

References

[1] A.-M. Awal, G. Feng, H. Mouchere, and C. Viard-Gaudin. First experiments on a new online handwritten flowchart database. In *DRR'11*, pages 1–10, 2011. 1

[2] V. Babu, L. Prasanth, R. Sharma, and A. Bharath. Hmm-based online handwriting recognition system for telugu symbols. In *Ninth International Conference on Document Analysis and Recognition, 2007. ICDAR 2007.*, volume 1, pages 63 –67, sept. 2007. 2

[3] C. Bishop, M. Svensen, and G. Hinton. Distinguishing text from graphics in on-line handwritten ink. In *Ninth International Workshop on Frontiers in Handwriting Recognition, 2004. IWFHR-9 2004.*, pages 142 – 147, oct. 2004. 2

[4] D. Blostein. General diagram-recognition methodologies. In *Selected Papers from the First International Workshop on Graphics Recognition, Methods and Applications*, pages 106–122, London, UK, UK, 1996. Springer-Verlag. 1

[5] V. Franc and V. Hlaváč. Statistical pattern recognition toolbox for Matlab. Research Report CTU–CMP–2004–08, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, June 2004. 3

[6] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet. Unipen project of on-line data exchange and recognizer benchmarks. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition, 1994. Vol. 2 - Conference B: Computer Vision amp; Image Processing.*, volume 2, pages 29 –33 vol.2, oct 1994. 6

[7] E. Indermühle, V. Frinken, and H. Bunke. Mode detection in online handwritten documents using BLSTM neural networks. In *ICFHR '12: Proceedings of the 13th International Conference on Frontiers in Handwriting Recognition*, pages 302–307, 2012. 2

[8] L. B. Kara and T. F. Stahovich. Hierarchical parsing and recognition of hand-sketched diagrams. In *ACM SIGGRAPH 2007 courses*, SIGGRAPH '07, New York, NY, USA, 2007. ACM. 2

[9] M. Luqman, T. Brouard, and J.-Y. Ramel. Graphic symbol recognition using graph based signature and bayesian network classifier. In *10th International Conference on Document Analysis and Recognition, 2009. ICDAR '09.*, pages 1325 –1329, july 2009. 2

[10] R. Niels, D. Willems, and L. Vuurpijl. The NicIcon database of handwritten icons. In *11th International Conference on the Frontiers of Handwriting Recognition (ICFHR 2008)*, Montreal, Canada, August 2008. In press. 6

[11] S. Otte, D. Krechel, M. Liwicki, and A. Dengel. Local feature based online mode detection with recurrent neural networks. In *ICFHR '12: Proceedings*

of the 13th International Conference on Frontiers in Handwriting Recognition, pages 531–535, 2012. 2

[12] E. J. Peterson, T. F. Stahovich, E. Doi, and C. Alvarado. Grouping strokes into shapes in hand-drawn diagrams. In *AAAI'10*, 2010. 2

[13] M. I. Schlesinger and V. Hlaváč. *Ten Lectures on Statistical and Structural Pattern Recognition (Computational Imaging and Vision)*. Springer, 2002 edition, Mar. 2012. 1

[14] G. Shi and Y. Zhang. An improved svm-hmm based classifier for online recognition of handwritten chemical symbols. In *2010 Chinese Conference on Pattern Recognition (CCPR).*, pages 1 –5, oct. 2010. 2

[15] D. Tabernik, M. Kristan, M. Boben, and A. Leonardis. Learning statistically relevant edge structure improves low-level visual descriptors. In *International Conference on Pattern Recognition*, 2012. 5

[16] C. H. Teo, A. J. Smola, and Q. V. Le. Bundle Methods for Regularized Risk Minimization. *Journal of Machine Learning Research*, 11:311–365, 2010. 3

[17] X.-D. Zhou and C.-L. Liu. Text/non-text ink stroke classification in japanese handwriting based on markov random fields. In *Ninth International Conference on Document Analysis and Recognition, 2007. ICDAR 2007.*, volume 1, pages 377 –381, sept. 2007. 2

[18] B. Zhu and M. Nakagawa. Building a compact on-line mrf recognizer for large character set using structured dictionary representation and vector quantization technique. In J. E. Guerrero, editor, *ICFHR '12: Proceedings of the 13th International Conference on Frontiers in Handwriting Recognition*, pages 155–160, Los Alamitos, USA, 2012. IEEE Computer Society. 2

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Detection of Brain Tumors Based on Automatic Symmetry Analysis

Pavel Dvorak
Department of Telecommunications
Faculty of Electrical Engineering and Communication
Brno University of Technology, 612 00 Brno, Czech Republic
pavel.dvorak@phd.feec.vutbr.cz

Walter Kropatsch
Pattern Recognition and Image Processing Group
Institute of Computer Graphics and Algorithms
Faculty of Informatics
Vienna University of Technology, Favoritenstr. 9/186-3, A-1040 Vienna, Austria
krw@prip.tuwien.ac.at

Abstract. *This article focuses on the detection of a brain tumor location in magnetic resonance images. The aim of this work is not the precise segmentation of the tumor and its parts but only the detection of its approximate location. It will be used in future work for more accurate segmentation. For this reason, it also does not deal with detecting of the images containing the tumor. The algorithm expects a 2D T2-weighted magnetic resonance image of brain containing a tumor. The detection is based on locating the area that breaks the left-right symmetry of the brain. The created algorithm was tested on 73 images containing tumor, tumor with edema or only edema. These pathological structures had various sizes and shapes and were located in various parts of the brain.*

1. Introduction

The detection of brain tumors is generally a more complex task than the detection of any other image object. Pattern recognition usually relies on the shape of the required objects. But the tumor shape varies in each case so other properties have to be used. The general properties of healthy brain are widely used as a prior-knowledge. One of them is the probability of tissues locations using probability brain atlas, which is used e.g. in [7]. Another widely used knowledge, which is used in this article, is the approximate left-right symmetry of healthy brain. This approach is also used e.g. in [3] [4] [5]. Areas that break this symmetry are most likely parts of a tumor.

There are also many other methods used for tumor extraction, but they usually rely on machine learning algorithms such as SVM used e.g. in [6]. For this purpose, many algorithms need to have patient-specific training dataset. This makes the method more demanding for the experts. These methods usually rely on other contrast images, such as T1-weighted contrast enhanced images [10]. Fully automatic exact segmentation of the tumor is still an unsolved problem, as the accurate image segmentation itself. The method proposed in this work is less accurate than many other methods used nowadays, but it is fully automatic and it is used only for the detection of the brain tumor location for subsequent segmentation, which will be the aim of future work.

The big advantage of the symmetry approach is that the process does not need any intensity normalization, human work etc. The only step that needs to be done is the symmetry axis detection. Another advantage is its independence on the type of the tumor. It can correctly detect anomalies in images containing a tumor, a tumor with edema or only an edema, which is an abnormal accumulation of the fluid around the tumor and is present only with particular types of tumors.

2. Proposed Method

The input of the whole process is a stand-alone 2D T2-weighted magnetic resonance image containing a tumor. It means that no neighbor slices are considered. The reason for T2-weighted image is the visibility of tumors in this type of image.

The tumor detection process consists of several steps. The first step is the brain extraction followed by cutting the image. In this cut image, the asymmetric parts are detected and then the decision which half contains the tumor is made. The detection of the symmetry axis is skipped because the input data were aligned in previous processing. The only assumption of proposed method is a vertically aligned head. For the purpose of detecting the symmetry axis, the well performed algorithm works and is described in [8]. Addition of this method as a preprocessing step will be one of the aims of the future work.

2.1. Brain Extraction

The extraction of skull is based on technique mentioned in [2] and is done by the well-known method called Active contour, or Snakes [9]. At first, the smallest rectangle, whose sides are parallel to the image sides, surrounding the skull are detected. The initial mask is set to this rectangle to be sure that the whole skull is inside the mask. Then the algorithm is executed.

Assuming that the head is approximately symmetric, the symmetry axes is set to be parallel to the vertical axis and to divide the detected rectangle into two parts of the same size.

The results of the segmentation algorithm is not only the border of the skull, but also the border of the brain. This border is used to extract only the brain instead of the whole skull as in [2]. Only the segment that is located in the center is extracted. Because in some cases the brain segment can be joined to the skull segment but not symmetrically, another processing has to be done. The operation of logical conjunction is performed with this segment and its symmetric flipped image. This causes that points that are not on one side will not be considered also on the other side. The resulting mask is applied to the input image. The result of the brain extraction is shown in Figure 1.

The described process approximately extract the brain and set the symmetry axis in center of the new image. Except the brain, in cases where eyes are present, they are also inside the brain mask because there is usually not clear border between them and brain.

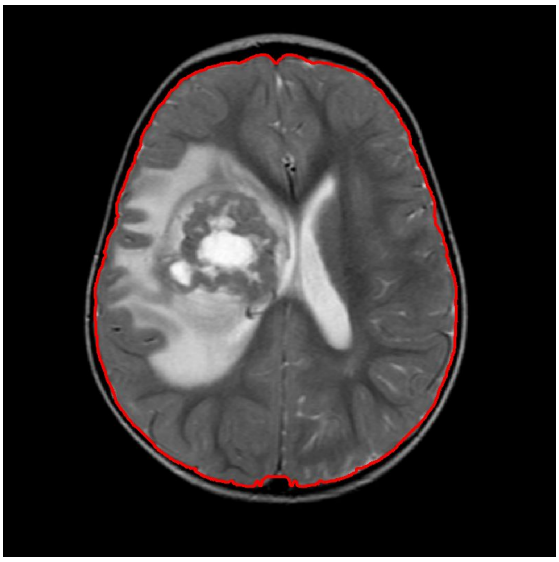


Figure 1. Extracted brain.

Even if the mask is not so precise, the future results are not so influenced because the asymmetries caused by tumor are much higher.

After the extraction of the mask, the image is filtered by a Gaussian filter of size 5x5 to make the particular parts more homogeneous. The resulting mask is then applied to this filtered image followed by cutting the image because in parts outside the mask, the symmetry does not need to be checked.

2.2. Asymmetry detection

The main part of this work is the detection of symmetric anomalies, which are usually caused by brain tumor, whose detection is the main purpose of this article. The first step of this process is dividing of the input image into two approximately symmetric halves.

Assuming that the head is not rotated and the skull is approximately symmetric, the symmetry axis is parallel to vertical axis and divide the image of detected brain into two parts of the same size.

A squared block, with the side length computed as one quarter of the longer side of the input image, is created. This size is suitable for the detection of both small and large tumors. The algorithm goes through both halves symmetrically by this block. The step size is smaller than the block size to ensure the overlapping of particular areas. These areas are compared with its opposite symmetric part. In this case, the step size of one sixteenth of the block size was set.

Comparing is done by Bhattacharya coefficient. [1] Normalized histograms with the same range are computed from both parts and the Bhattacharya coefficient is computed from these histograms as follows [1]:

$$BC = \sum_{i=1}^{N} \sqrt{l\,(i) \cdot r\,(i)}, \qquad (1)$$

where *N* denotes the number of bins in the histogram, *l* and *r* denote histograms of blocks in left and right half, respectively.

The range of values of Bhattacharya coefficient is $\langle 0, 1 \rangle$, where the smaller value, the bigger difference between histograms. For the next computation, the asymmetry is computed as:

$$A = 1 - BC. \qquad (2)$$

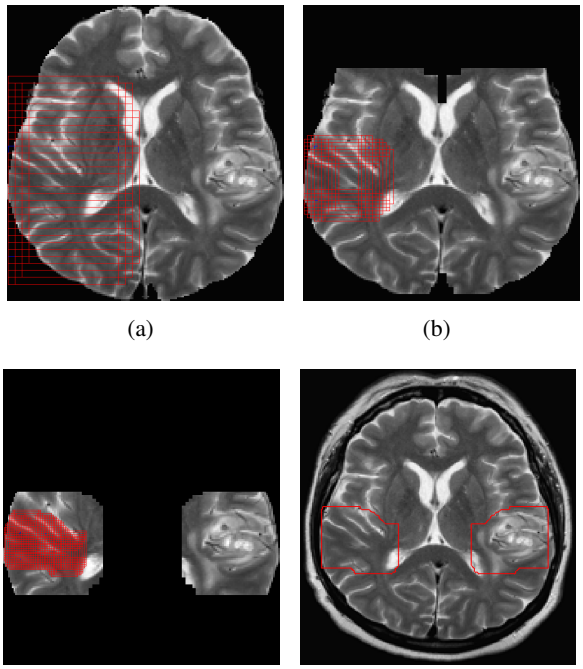
This asymmetry is computed for all blocks. The global maximum is detected. This is the most asymmetric block and most likely contains the tumor. Since the tumor can be larger, the initial size of the block, also the blocks with asymmetry bigger then *0.5· max(Asym)*, are extracted. This threshold was set experimentally as a compromise between the size of the area and the asymmetry of areas. When the threshold was decreased, the resulting areas were too large, while for higher thresholds, some parts of the tumors were located outside the area.

The output of this computation is a both-sided mask containing the most asymmetric parts. This mask is slightly enlarged by morphological operation dilation for the case that some part of the tumor could be outside the region. This mask is applied to the input image.

The whole cycle is repeated twice for this new image but with smaller block. Height and width of the block is iteratively reduced to the half of the previous value. So the new size of the block is one quarter and one sixteenth of the initial size, respectively. The purpose of smaller areas is the more precise detection of asymmetry. This approach corresponds to multi resolution image analysis described in [11].

The resulting both-sided mask is again applied to the input image and this image is sent to the output of the detection process.

The results of particular steps are shown in the Figure 2. The input image size in this example was 256x256, so the Figures 2(a), 2(b) and 2(c) demonstrate detection of the most asymmetric areas for the block size of 64x64, 32x32 and 16x16 pixels, respectively. As can be seen, searching for asymmetric parts is done only in asymmetric areas provided by previous step.



(a) (b) (c) (d)

Figure 2. Asymmetries detection: (a) the first step for block size of 64x64 pixels, (b) the second step for block size of 32x32 pixels, (c) the third step for block size of 16x16 pixels, (d) the result of the asymmetries detection,

2.3. Locating the tumor

The detection of asymmetric areas does not explicitly locate the position of the tumor. There are still two possible locations of the tumor - right or left side. Two methods, for deciding in which part the tumor is, were tested. First of them is the prior-knowledge of the physical properties of brain tissues.

In T2-weighted images, tumors and edemas appear hyperintense [13]. This means that the produced signal is stronger than the signal of the white matter, in which tumors are located in most cases. This method is based on computation of the mean of the region. Tumors located near ventriculus could cause problems, because ventriculus produces even stronger signal. This could lead to misclassification.

The second possibility how to locate the tumor is to find it in the same way as asymmetries. Normalized histograms are computed from both areas and also from the rest of the brain. Histograms of both areas are compared with the rest of the brain using Bhattacharya coefficient. Area with less similar his-

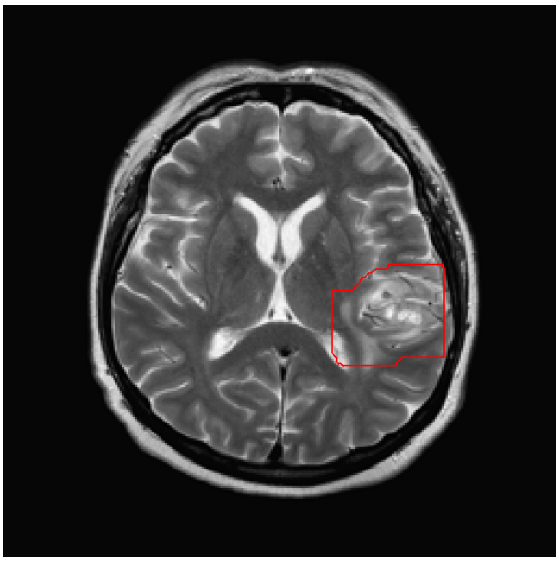


Figure 3. Located tumor.

togram is labeled as the one containing the tumor.

Both methods were tested. The first one produces slightly better results, the quantitative results are described in the next section.

The result of the tumor location for the input image from Figure 2(d) is shown in Figure 3. In this figure, the result image of the whole algorithm is demonstrated.

A problem occurs if the tumor appears in both halves of the brain. Since the tumor is not symmetric it is likely detected as asymmetric area even in this case. But the locating step relies on comparing both sides, therefore only one of them can be labeled as a pathological.

3. Results

The algorithm was tested on 73 T2-weighted images from 13 different patients. Every image contained a tumor, a tumor with an edema or only the edema. Various shapes, locations, and sizes of these pathological areas and various image resolution were tested. Results are shown in Table 1. Results are described by number of cases and by percentage of the total number of tested images.

At first, the evaluation of the detection of symmetric anomalies will be described. In 1 case, the anomaly detection failed. In this particular image, only the edema was present and it was hardly visible even for human. At least 75% of the pathological area was detected in 72 cases. In 8 cases, the pathological area was found, but the extracted area was too large compared to the tumor, or the tumor

Result	Num. of cases	Percentage
Number of images	73	
Incorrect anomaly detection	1	1.37%
Detected main part of path. area	72	98.63%
Too large area	8	10.96%
15-20% outside	9	12.33%
Correct anomaly detection	55	75.34%
Correct area extraction	52	71.23%

Table 1. Total results.

was not in the approximate center of this area. The example of this result is shown in Figure 7(b). In 9 cases, the pathological area was found, but from 15% to 25% of it was situated outside the extracted area. This includes also 3 images, where the pathological area was located in both halves. Such case is shown in Figure 7(a). In only 2 of these 9 cases, more than 20% of the pathological area was located outside the extracted region. It means that in 17 cases, the result of anomaly detection was not totally incorrect, but it was not so accurate. In 55 cases, the anomaly detection was correct.

After the anomaly detection process, the decision, on which side the pathological area is, has to be done. In this part, the case with incorrect anomaly detection result is not considered. The region mean computation failed in 6 cases, from which 3 cases belongs to the group of 55 correctly detected areas. So the total number of correctly extracted area is 52. For localizing the tumor by comparing to the rest of the brain, the computation failed in 8 cases, from which 6 cases belongs to the group of 55 correctly detected areas. So the total number of correctly extracted area is 49.

In Table 2, the results dependent on pathological area size are shown. There were 8 small, 23 medium and 42 large tumors. According to the assumption, the most of tumors, whose part was situated outside the extracted region, belongs to the group of large pathological area, and the only totally incorrect result belongs to the group of small pathological area.

A few results can be seen in the Figure 4, 5 and 6. The area of the tumor location is surrounded by a red line. One can see that the detected area is a little bit larger than the pathological area itself. One reason is the use of dilation at the end of asymmetry detection.

Result	Size of pathological area		
	Small	Medium	Large
Number of images	8	23	42
Incorrect anomaly detection	1	0	0
Detected main part of path. area	7	23	42
Too large area	2	4	2
15-20% outside	0	1	8
Correct anomaly detection	5	18	32
Correct area extraction	5	17	30

Table 2. Results dependent on tumor size.

This is done to locate the whole tumor and not only a part of it. Another reason could be explained by influence of the tumor in the neighbor parts of the brain. Because the tumor is a tissue which is growing during the time, it presses the other parts of the brain. This creates the deformation and asymmetry not only in the tumor location but also in the adjacent parts and gradually in the whole brain.

Since the method is based on asymmetry detection, the problem appears when the tumor is located in both halves or on the symmetry axis. In this case, some parts of the tumor could be outside of the extracted area even if they are located in the half in which the tumor was detected. The reason is that the tumor located in both sides causes symmetry in these parts, so for the algorithm it seems to be a healthy tissue.

The part of the tumor located in the other half of the brain is also outside the detected area. The example of that problematic type of tumors is shown in the Figure 7(a). This problem could be prevented by an additional step that consists of checking whether the border of the asymmetric area matching the symmetry axis border, in other words if the both-sided mask creates only one homogeneous region.

Compared to the approach proposed in [2], our algorithm provides a region containing the most of the tumor area, which will be necessary in the next processing that is the aim of the future work. Moreover, the results of our method are not simple rectangles, but they can better capture the structure of the tumor.

From the principle, the proposed algorithm could also detect multifocal tumors as separated regions [14]. Unfortunately, this assumption was not tested, because no images containing multifocal tumors were present.

4. Conclusion and future work

The aim of this work was not the precise segmentation of the brain tumor but only detection of approximate location of the tumor. This location could be then used for more precise tumor extraction and could make this task easier. The proposed method correctly found the pathological area in 55 from 73 images. In other 17 cases, the main part of the pathological area was detected, but the result was not so precise. Tumors ware correctly extracted in 52 cases.

The future work will consist of the automatic symmetry axis detection and the more precise extraction of the tumor based on current results.

The attention in the future work will also be paid on automatic detection of the image containing the brain tumor and searching for the relations between neighbor slices. After that, the work will continue with extending the method to 3D MR images.

5. Acknowledgments

This research is part of the project reg. no CZ.1.07/2.3.00/20.0094 "Support for incorporating R&D teams in international cooperation in the area of image and audio signal processing" and is co-financed by the European Social Fund and the state budget of the Czech Republic.

References

[1] Bhattacharyya, A., "On a measure of divergence between two statistical populations defined by their probability distribution," *Bulletin of the Calcutta Mathematical Society*, vol. 35, pp. 99–110, 1943. 3

[2] Ray, N. and Saha, B.N. and Graham Brown, M.R., "Locating Brain Tumors from MR Imagery Using Symmetry," *Signals, Systems and Computers, 2007. ACSSC 2007. Conference Record of the Forty-First Asilomar Conference on*, pp. 224–228, November 2007. 2, 5

[3] Khotanlou, H. Colliot, O. and Bloch, I., "Automatic brain tumor segmentation using symmetry analysis and deformable models,"*Conf. on Advances in Pattern Recognition ICAPR, Kolkata, India,*. Jan. 2007. 1

[4] Pedoia, V., Binaghi, E., Balbi, S., De Benedictis, A., Monti, E., et al., "Glial brain tumor detection by using symmetry analysis,"*CProc. SPIE 8314, Medical Imaging 2012: Image Processing, 831445,*. February 23, 2012. 1

[5] Somasundaram, K., Kalaiselvi, T., "Automatic detection of brain tumor from MRI scans using maxima transform,"*National Conference on Image Processing (NCIMP)* 2010. 1

[6] Zhang, J. G., Ma, K. K., Er, M. H., and Chong, V., "Tumor segmentation from magnetic resonance imaging by learning via one-class support vector machine,"*International Workshop on Advanced Image Technology (IWAIT 2004)*., pp. 207–211, 2004. 1

[7] Cuadra, M. B., Pollo, C., Bardera, A., Cuisenaire, O., Villemure, J. G. and Thiran, J. P., *Atlas-based segmentation of pathological MR brain images using a model of lesion growth.*, *IEEE Trans. Med. Imaging*., vol. 23, num. 1, pp. 1301–1314, 2004 1

[8] Ruppert, G. C. S., Teverovskiy, L., Yu, C., Falcao, A. X., Liu, Y., "A New Symmetry-based Method for Mid-sagittal Plane Extraction in Neuroimages" *International Symposium on Biomedical Imaging: From Macro to Nano*, 2011. 2

[9] Xu, C. and Prince, J. L., "Snakes, shapes, and gradient vector flow." *IEEE Transactions on Image Processing*, vol. 7, num. 3, pp. 359–369, 1998. 2

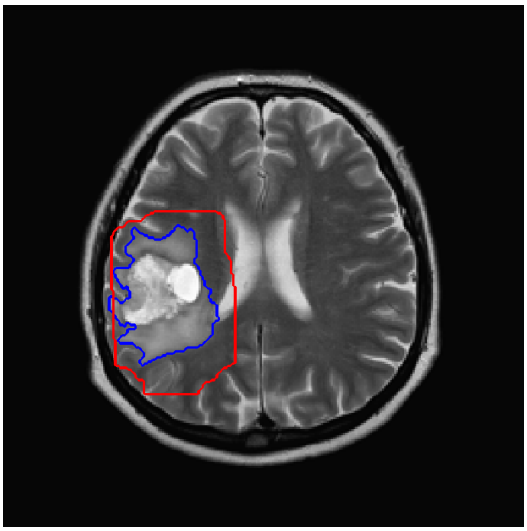
[10] Capelle, A. S., Colot, O., and Fernandez-Maloigne, C., "Evidential segmentation scheme of multi-echo MR images for the detection of brain tumors using neighborhood information.," *Information Fusion*.vol. 5, pp. 103–216, 2004. 1

[11] Kropatsch, W. G., Haxhimusa, Y., Ion, A., "Multiresolution Image Segmentations in Graph Pyramids," in *Applied Graph Theory in Computer Vision and Pattern Recognition Studies in Computational Intelligence*, vol. 52, pp. 3–41, 2007. 3

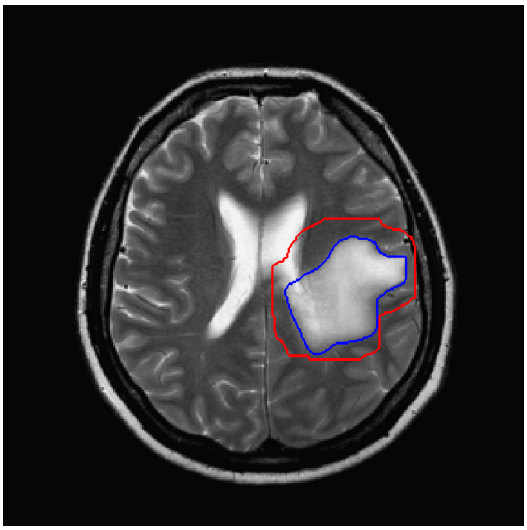
[12] Rodrguez, A. O., "Principles of magnetic resonance imaging."*Revista Mexicana de Fsica*, 2004.

[13] Shah, L. M., Salzman, K. L., "Imaging of Spinal Metastatic Disease."*International Journal of Surgical Oncology*, 2011. 3

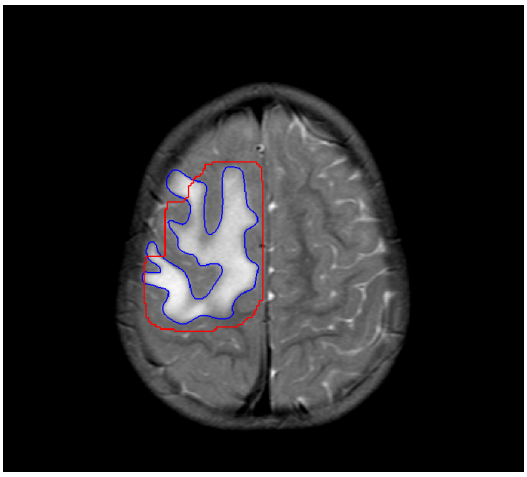
[14] Lim, D. A., Cha, S., Mayo, M. C., Chen, M.-H., Keles, E., VandenBerg, S., Berger, M. S.,"Relationship of glioblastoma multiforme to neural stem cell regions predicts invasive and multifocal tumor phenotype," in *Neuro Oncol*, vol. 9, Issue. 4, pp.424–429, October 2007. 5



(a)

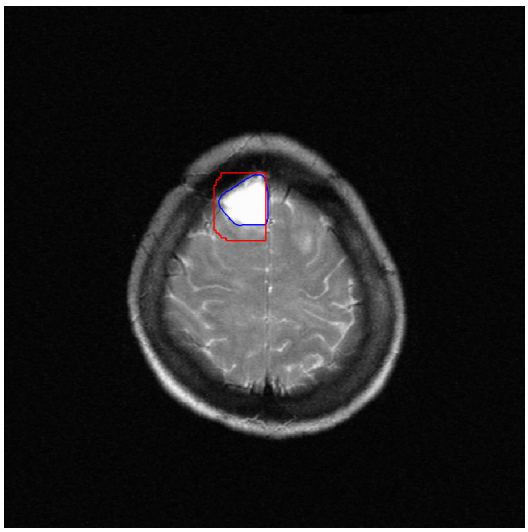


(b)

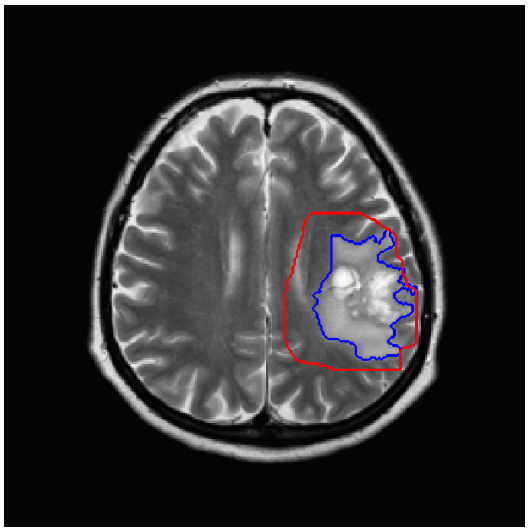


(c)

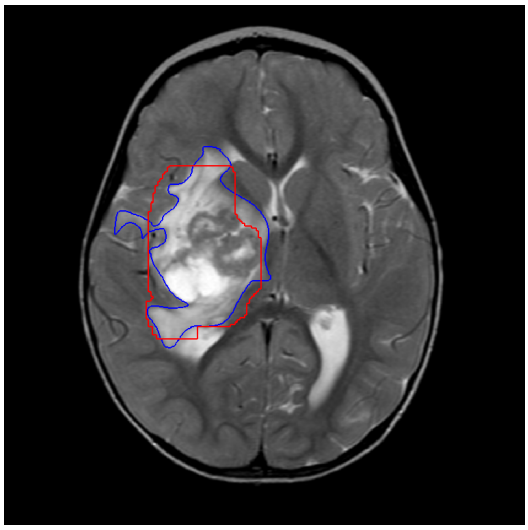
Figure 4. Examples of results (red area) compared to the ground truth (blue area).



(a)

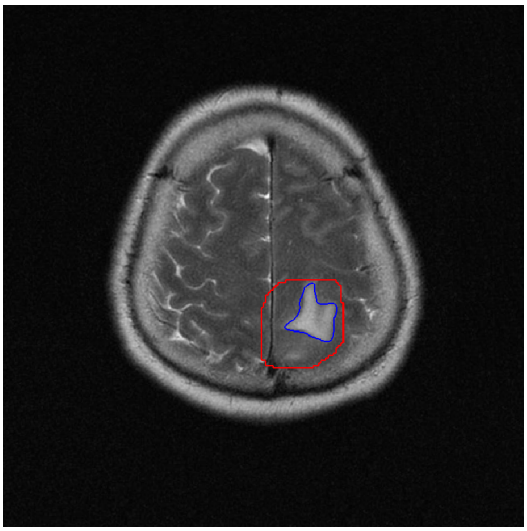


(b)

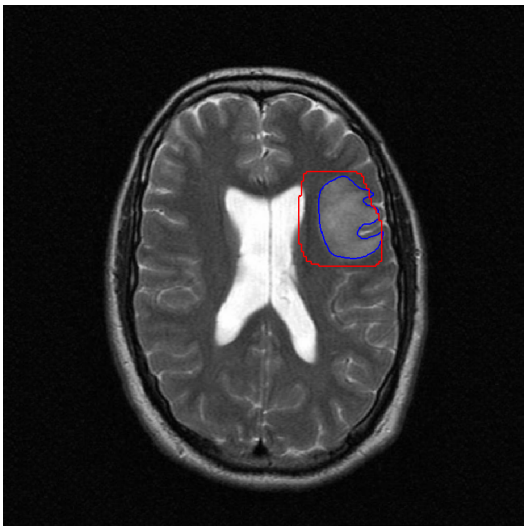


(c)

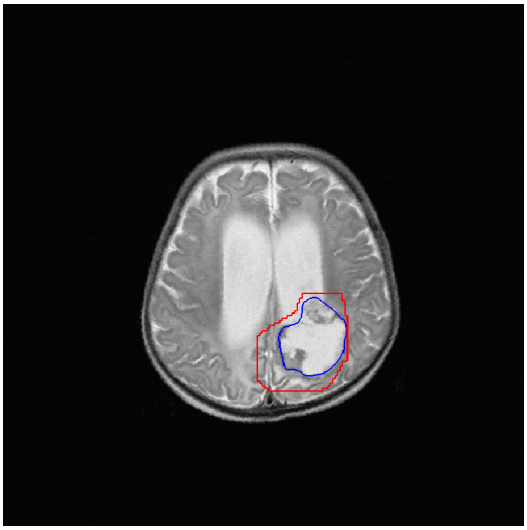
Figure 5. Examples of results (red area) compared to the ground truth (blue area).



(a)

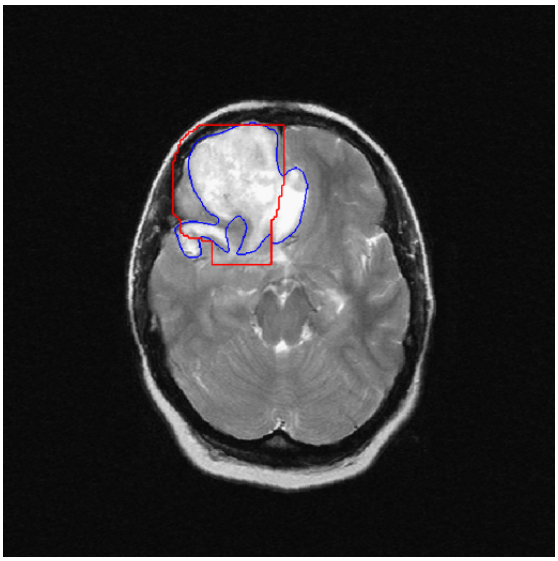


(b)

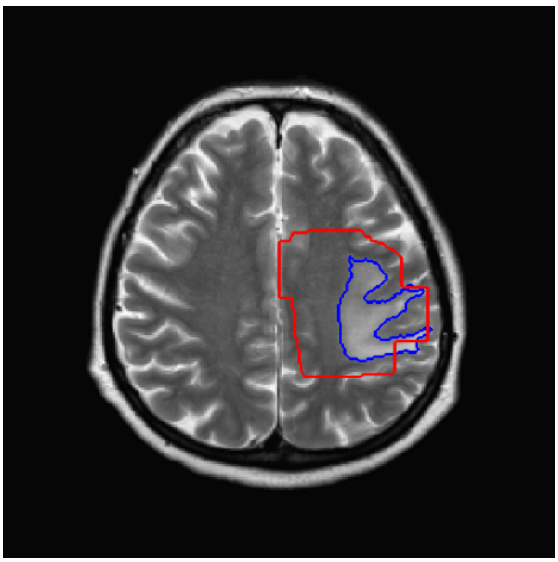


(c)

Figure 6. Examples of results (red area) compared to the ground truth (blue area).



(a)



(b)

Figure 7. Less precise results (red area) compared to the ground truth (blue area): (a) problematic type of tumor located in both halves, (b) result evaluated as a large area.

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Hierarchical Histogram for RGB Color Modeling

Lukáš Cerman and Václav Hlaváč
Czech Technical University, Faculty of Electrical Engineering
Center for Machine Perception
121 35 Prague 2, Karlovo náměstí 13, Czech Republic
{*cermal1*, *hlavac*}*@fel.cvut.cz*

Abstract. *A histogram-like model is suggested for the representation of multi-dimensional distributions such as RGB colors of subjects in tracking and segmentation tasks. Unlike the normal 3-D histogram it can be estimated from the limited amount of training data without the need to reduce the precision of the measured data. The proposed hierarchical histogram model (HHM) is compared to the well known Gaussian mixture model (GMM). The HHM is able to represent the underlaying distribution with the similar quality as the GMM but it surpasses it in the speed of parameters estimate and log-likelihood evaluation. This makes it a suitable color model for the time critical tasks like real-time tracking.*

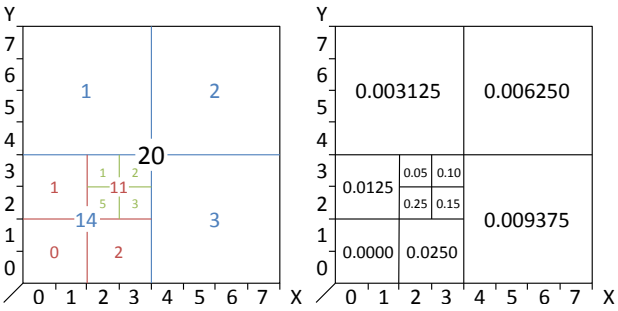


Figure 1. An example of the 2-D hierarchical histogram built up from twenty 3-bit data samples: $(1,2)$, $(2,1)$, $(2,1)$, $(2,2)$, $(2,2)$, $(2,2)$, $(2,2)$, $(2,2)$, $(2,3)$, $(3,2)$, $(3,2)$, $(3,2)$, $(3,3)$, $(3,3)$, $(3,4)$, $(4,2)$, $(4,2)$, $(4,3)$, $(4,5)$, $(5,4)$. The left image shows how the sample counts are distributed among (split) bins with the bin splitting threshold $\mathrm{K}_b = 10$. The right image shows the probability density corresponding to each leaf bin.

1. Introduction

Many computer vision tasks, such as object tracking or segmentation, involve statistical modeling of the individual subjects (which can be objects, backgrounds, etc.) appearance. If the subjects have multiple distinct colors, which is in fact a very frequent case, a multi-modal representation of their appearance is required. Another question that must be considered when deciding about the appearance model is the data dimensionality. A vast majority of the contemporary cameras provide RGB images on their output, thus a three dimensional model is requested. Some authors simplify the task by converting the original RGB measurements to one dimensional intensity values, which can be sufficient in some tasks. However, in this work, we are interested in a full 3-D representation of the measured data.

A very popular model used to represent 3-D RGB color distributions is the Gaussian mixture model

$$P(x|\psi) = \sum_{j \in \mathrm{M}} \omega_j N(x|\Sigma_j, \mu_j) \ , \qquad (1)$$

where x is a 3-D measurement, N is a multivariate normal distribution, M is a number of the mixture components and

$$\psi = (\omega_1, \Sigma_1, \mu_1, \ \ldots \ , \omega_\mathrm{M}, \Sigma_\mathrm{M}, \mu_\mathrm{M}) \qquad (2)$$

are the model parameters, the number of which is usually relatively low. Such a model is well suited for tasks with a limited amount of training data. The disadvantage of GMM lies in the computation complexity of estimating the parameter vector ψ, which is usually estimated using the K-means [4, 1] or (even slower) the EM algorithm providing the maximum

likelihood estimate of ψ [5, 3, 1, 6]. For time critical applications, even the calculation of the data likelihood $P(x|\psi)$ can be expensive, as it involves a computation of exponentials and logarithms. Another disadvantage is that the number of components M must be set a-priori or estimated together with ψ, which can slow the whole process further down.

In one dimensional cases, the histograms are often used when a quick model learning and PDF calculations are required. However, the use of histogram becomes impractical with increasing number of dimensions, as the number of bins grows exponentially with the number of dimensions. To obtain a good density estimate using a traditional histogram in the 3-D color space, a huge amount of training data is required (note that a full precision 3-D histogram of 8-bit color samples has 16.7 millions bins) or the spatial precision must be sacrificed (e.g., cutting the precision down to 2-bits reduces the amount of bins to 64).

In this work, we propose a histogram-like model, called hierarchical histogram model (HHM) that compromises on the *model size* vs. the *precision* dilemma by providing a fine density estimate in the areas largely supported by the data and a coarse estimate in the areas with sparse data support. Having this property, it can be used to model densities in higher dimensional spaces, but at the same time, it can be learned using the limited amount of training data. The proposed HHM provides similar classification performance as the GMM model, but surpasses it in the speed of the parameters estimate and the likelihood calculation.

The HHM building process is similar to an octree, which has been already used for color representation in the past, e.g., in [7]. However, in [7] the tree is built in a bottom-up fashion, starting from the leaves and pruning the tree by merging the sub-trees that collected a substantial amount of data. In our work, we do the exact opposite: we try to achieve as fine representation as possible by starting from the root node and splitting the current leaves in the sub-trees if they collected enough data.

2. The Hierarchical Histogram

Unlike a standard 3-D histogram with uniformly distributed bins, the hierarchical histogram is represented by a hierarchical structure resembling an octree, which is built recursively using the following rules:

1. At the top level, the data are sorted to eight bins by the index composed of the most significant bit of each color component of the RGB data sample.
2. If a bin collects more than K_b samples, it is split to eight sub-bins. The samples from the bin are sorted to the eight sub-bins by the index composed of the second most significant bit of each color component of the data sample.
3. If a sub-bin collects more than K_b samples, the splitting process is recursively repeated in the same way, using the less significant bit, until the least significant bit is reached.

Once all training data are sorted to the bins, the histogram is normalized to obtain a probability: a height of each leaf bin (i.e., a bin that has not been split) is calculated as the number of samples it collected divided by the total number of samples and the area of the bin. See Figure 1 for a 2-D example.

It might happen that there is not enough data points to obtain a sufficiently good estimate of the distribution even with the hierarchical histogram. For this reason, the appearance is modeled using a mixture of the uniform distribution on a color cube,

$$U\left(x\right) = \left(\frac{1}{256}\right)^3 , \tag{3}$$

and the hierarchical histogram Z:

$$P(x|\phi) = \frac{\mathrm{K}_u}{\mathrm{K}_u + \mathrm{K}_z} U\left(x\right) + \frac{\mathrm{K}_z}{\mathrm{K}_u + \mathrm{K}_z} Z\left(x\,|\phi\right) , \tag{4}$$

where K_z is the number of data samples used to learn the histogram model and K_u is a user defined constant controlling the weight of the uniform distribution in the mixture. x is a 3-D vector (three 8-bit integers) representing a measured RGB color and ϕ are parameters of the histogram (i.e., positions and heights of the bins estimated from the training data). In cases with insufficient amount of the training data ($\mathrm{K}_z \to 0$), the mixture distribution (4) converges to the uniform distribution while in cases with enough training data ($\mathrm{K}_z \gg \mathrm{K}_u$), the uniform component U becomes negligible.

2.1. The Adaptive Color Model

An adaptive version of the hierarchical histogram can be used in applications that require sequential learning of the color model. It differs from the model described in the previous section in the following aspects:

1. A weight is assigned to each data point. The new points are added with the unit weight to the model.
2. The weight of the data points that are already in the model is decreased by a constant factor each time the model is being updated using a new batch of the training data.
3. The data points with a negligible weight are removed from the model.
4. While (re-)building the histogram, the point counts, including K_z, are calculated as sums of the data points weights.

2.2. Alternative Color Models

The individual RGB color channels tend to be correlated in many situations. This is also true for our data, see the strong diagonal structure of the probability densities in Figure 2. This can be addressed by transforming the original RGB color space to a different three dimensional space with the color and intensity information separated. We have tried to transform our data to CIE $L^\star a^\star b^\star$ and $rg\mathrm{I}$ color spaces, but the improvement of the separation of the individual subject classes in the destination space was too little to justify the additional computations involved in the color space transformation.

2.3. The Model Parameters

There are several constant parameters, that control the estimate of the hierarchical histogram model and calculations of its probability density function. The following table summarizes those parameters together with the values that were estimated empirically so that the good results are obtained when applied on the labeling problem presented later in *Section 3.2*.

	Parameter Represents	Used Value
K_b	bin splitting threshold	2
K_u	weight of the uniform distribution component	200

3. Results

The first part of this section is related to the test of the hierarchical histogram. The second part covers the application of the presented model on the labeling task.

3.1. Testing the Hierarchical Histogram Model

The performance of the hierarchical histogram model (HHM) depends on the choice of its parameters K_b and K_u as well as the amount of available training data. For this test, we collected a set of 404 606 data samples that were manually labeled to three classes. The set contains 340 600 background samples, 16 751 object samples and 47 255 companion samples. The data corresponds to one out of many video sequences used for testing the tracking algorithm [2], in which the tracking is performed as a labeling of the independently tracked feature points. The *companion* class represents the object's context.

We have performed several test batches, each with a different amount (1%, 5%, 10%, 15%, 25%, 35%, 50%, 65% and 80%) of the training data that were randomly sampled from the labeled set. The remaining data served as a validation set. Every batch was repeated ten times, each time with a new random sample of the same size. The repeated measurements from the same batch were averaged.

In each batch, we learned multiple HHM's corresponding to all combinations of $\mathrm{K}_b \in \{1, 2, \ldots, 20\}$ and $\mathrm{K}_u \in \{1^3, 2^3, \ldots, 40^3\}$. We also used the same training data to learn a reference ten-component Gaussian Mixture Model (GMM). The number of components was experimentally verified to provide the best results for this task. Then we used each trained model to classify the points from the validation set using a simple strategy:

$$l_i^\star = \arg\max_{l_i \in \mathcal{L}} \left(P(x_i | \phi^{l_i}) \right) , \qquad (5)$$

where $\mathcal{L}$ is the set of labels, $l_i^\star$ is a decided label of the point i, x_i is its RGB color and ϕ^{l_i} are model parameters of each class. Having all points classified we calculated the classification accuracy and the precision and recall measures of each class. Table 1 summarizes the results. It can be seen that, in terms of accuracy, the HHM outperforms the GMM, with the exception of the smallest training set, where the HHM wins only for a certain combination of its parameters. Observing precision and recall measures, one can see that the HHM lags behind in object class recall and slightly in companion class recall too, especially when trained with small data sets and for large K_b and K_u. We took this into account when choosing $\mathrm{K}_b = 2$ a $\mathrm{K}_u = 200$ for the labeling algorithm, since we do not want to miss too many object points. Note that these measurements are based on the independent classification decisions. The complete classifier, as described in [2], involves pairwise dependencies, so the final classification will be better.

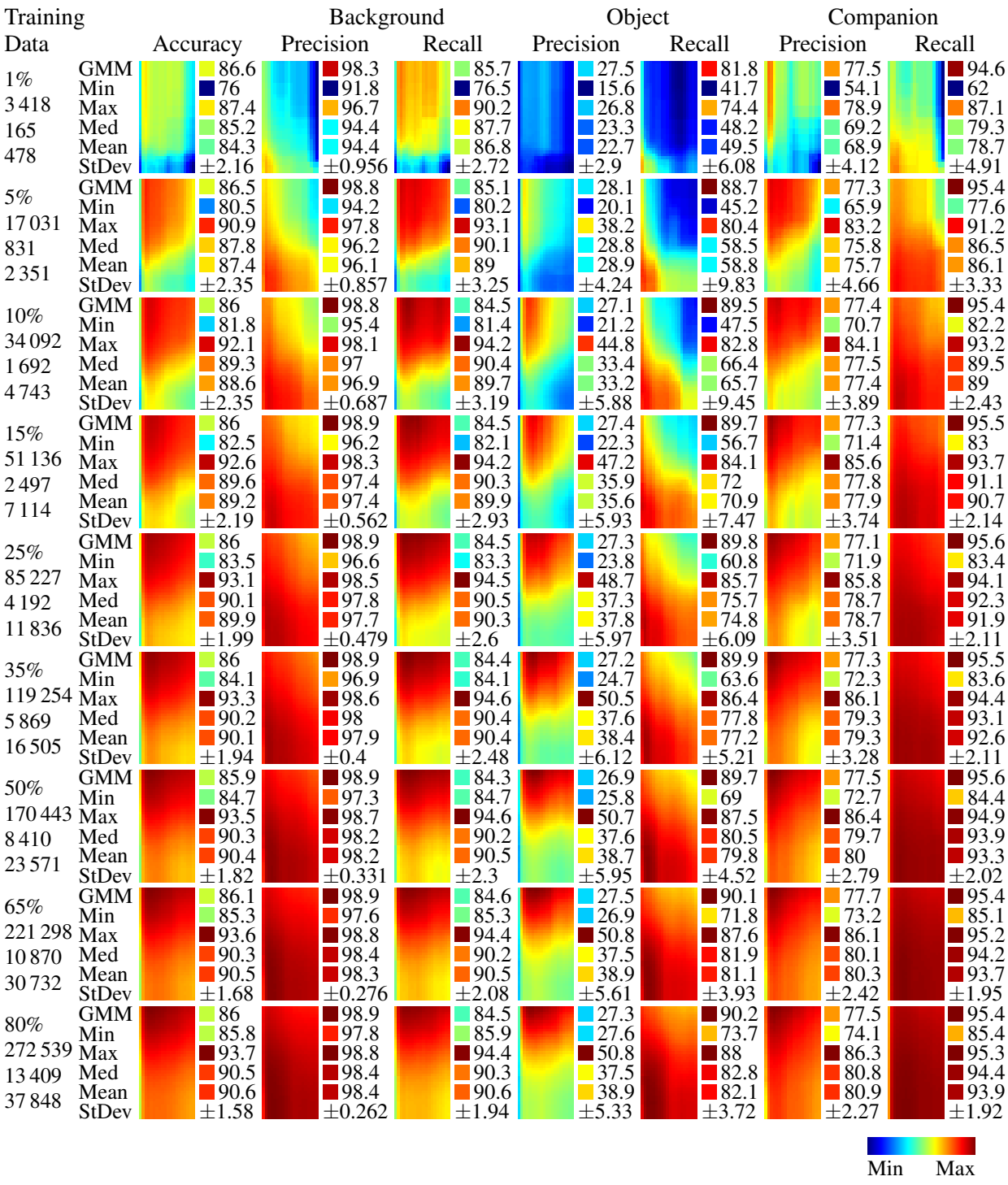


Table 1. Performance (in percent) of the hierarchical histogram model (HHM) in the various configurations compared to the Gaussian mixture model (GMM) as a reference. Each table row corresponds to a single test batch. The amount of the training data used in each batch is described in the first table column as a fraction (in percent) of the all data with the corresponding counts of the background, object and companion points displayed bellow it. Each table cell contains a small graph that represents the HHM performance measurements depending on $\mathrm{K}_b \in \{1, 2, \ldots, 20\}$ (the horizontal axis of the graph with its origin on the left) and $\mathrm{K}_u \in \{1^3, 2^3, \ldots, 40^3\}$ (the vertical axis of the graph with its origin on the bottom). The color code of the graph is different in each table column and corresponds to the minimum and the maximum value of the measurements in the whole column, see the scale to right of the table. Next to the small graph there are six numbers with color coded values in the same scale as the values in the graph. The first one is a measurement of the reference GMM performance and the others are statistics of the HHM measurements shown in the graph to the left of them – minimum, maximum, median, mean and standard deviation.

Training Samples	GMM [ms]	Hierarchical Histogram Model [ms]									
		K_b 2	4	6	8	10	12	14	16	18	20
4 061	1 639	72	70	69	67	67	65	64	64	63	63
20 213	2 837	81	78	76	75	74	73	73	73	72	72
40 527	4 840	87	83	81	80	79	78	77	77	76	76
60 747	4 271	92	87	85	84	83	82	81	80	80	79
101 255	5 021	100	95	93	91	90	89	88	88	87	86
141 627	6 464	106	102	99	97	96	95	94	93	93	92
202 424	7 807	116	110	108	106	104	103	102	101	100	99
262 900	7 864	124	117	114	112	110	109	108	107	106	105
323 796	8 091	130	123	120	118	116	114	113	111	111	110

Table 2. Execution time (in milliseconds) of estimating the hierarchical histogram model in various configurations from the training data and calculating likelihood of the testing data compared to the same operation performed with the Gaussian mixture model (GMM). The rows correspond to different amounts of data samples used for training.

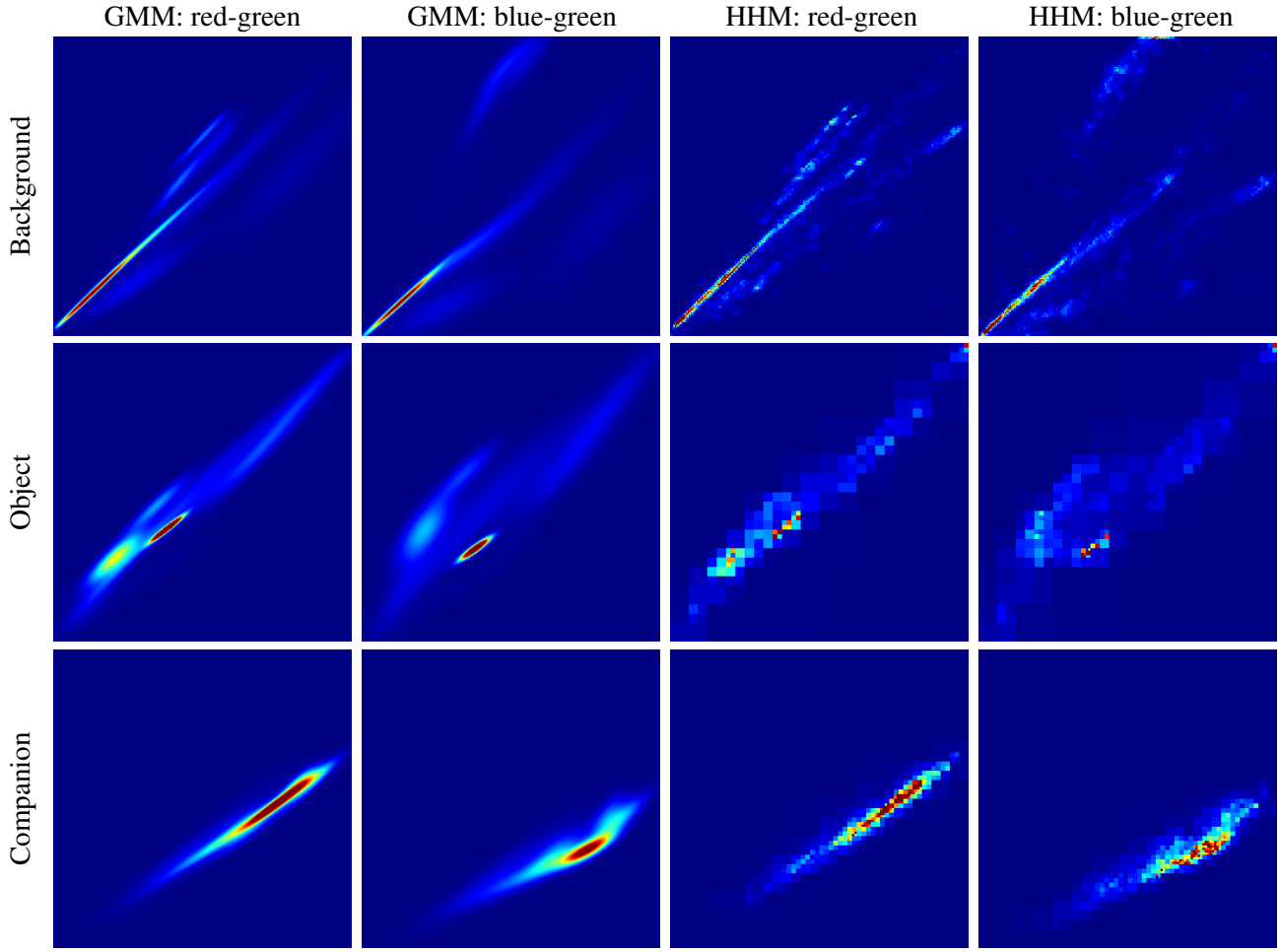
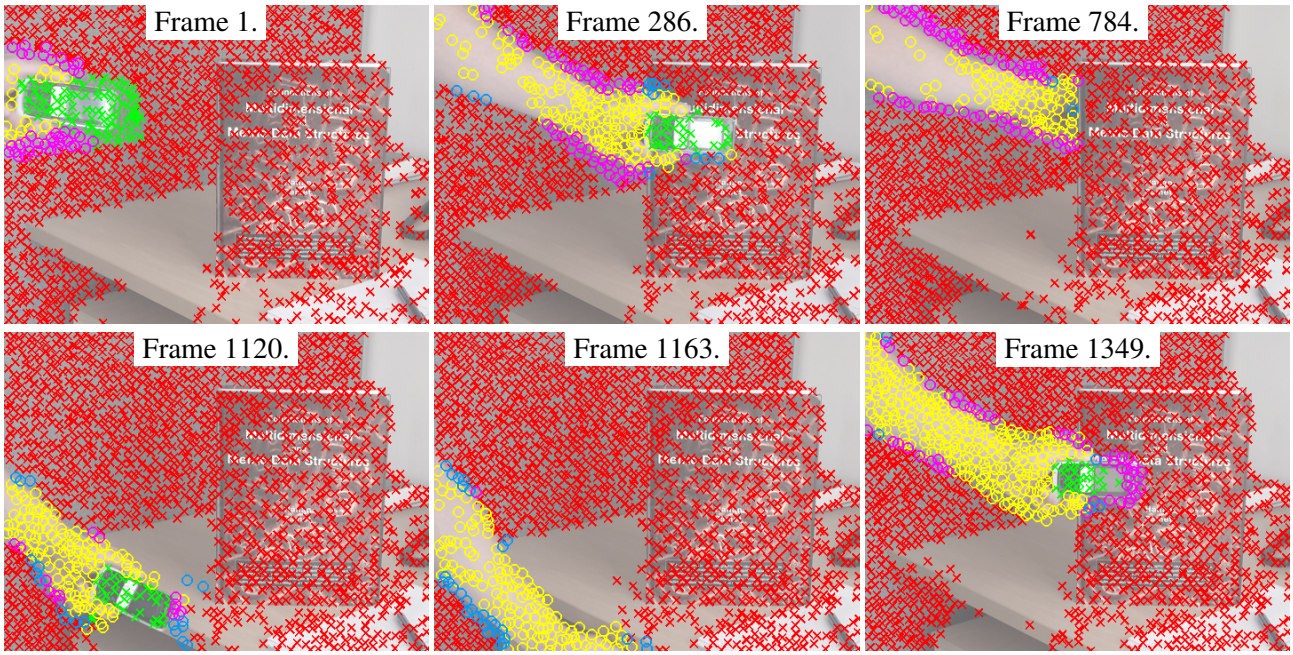


Figure 2. Comparison of the Gaussian mixture model (GMM) and the hierarchical histogram model (HHM) representing appearance of three different subject classes. The 3-D PDFs are visualised as the red-green / blue-green marginals. The horizontal axes of each graph has the origin on the left and represent the green intensity value. The vertical axis has the origin on the bottom and represent the red or blue intensity value.

Our aim when designing the hierarchical histogram model was to design a model with the similar classification performance but a much higher speed of the parameters estimate and the likelihood calculation than the Gaussian mixture model. Table 2 justifies that we reached the goal.

Tracking with the Gaussian mixture as the appearance model:



Tracking with the hierarchical histogram as the appearance model:

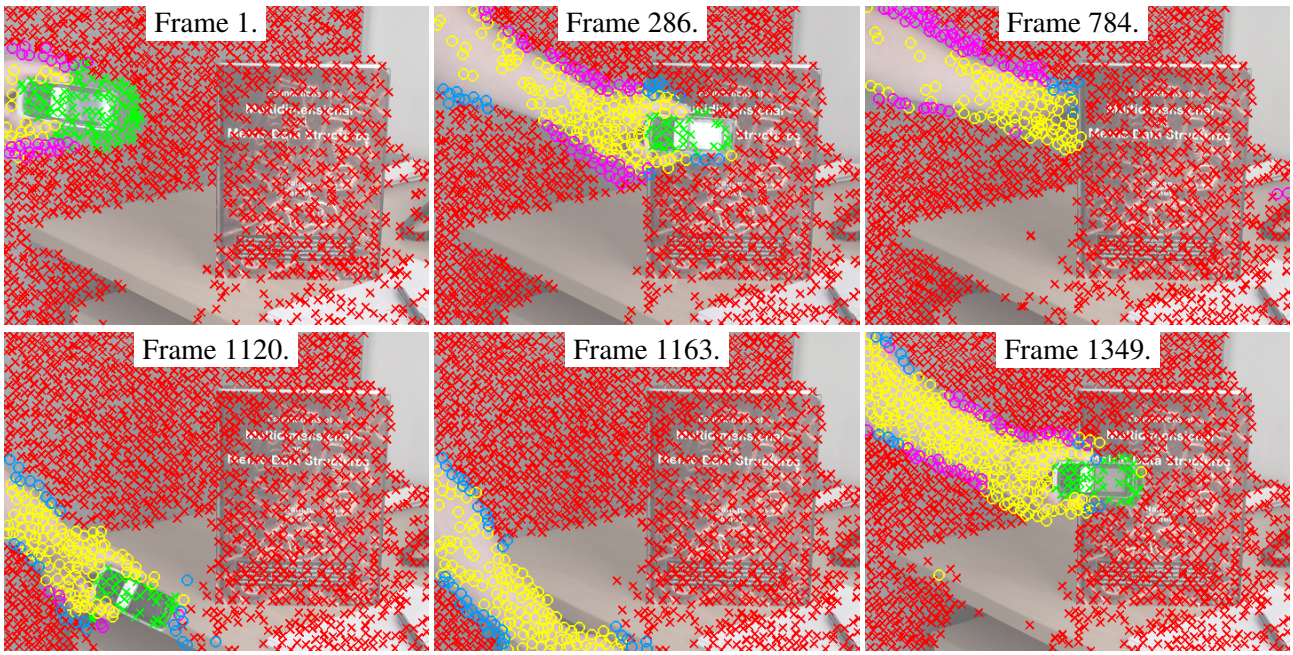


Figure 3. Application of the hierarchical histogram on the tracking task formulated as a semi-supervised learning and labeling problem.

3.2. Application to a Labeling Problem

Figure 3 demonstrates the application of the hierarchical histogram to the labeling task. The proposed model replaces the Gausian mixture in the appearance model of the algorithm described in [2]. The tracking results of the two appearance model variants are similar and in both cases the object (marked by green crosses) was successfully tracked to the end of the sequence. However, the time spent by the appearance model related calculation was much lower in the case of HHM model. From total 3 852 s of the tracking process, only 25 s were spent in the appearance model learning and 41 s in the log-likelihood calculations. In the case of GMM those figures were significantly higher: 7 873 s total running time, 3 684 s appearance model learning and 560 s for the evaluation of the appearance models log-likelihoods. The

speedup of the appearance related parts is in the factor of 63 which corresponds well to the figures in Table 2.

4. Conclusions

We have suggested a histogram-like model that can be used to represent RGB color of subjects in the tracking and segmentation tasks. The proposed model overcomes the curse of dimensionality of the traditional histogram and represents the color distributions with a similar quality as the well known and frequently used Gaussian mixture model, which was proven by experiments. In addition to a good representation of the involved distributions, the proposed HHM surpasses the GMM in the simplicity of the involved calculations, which makes it suitable for the time critical tasks like a real-time tracking.

5. Acknowledgements

The authors wish to thank the reviewers for the valuable comments on the paper. The authors were supported by the EC project FP7-ICT-247525 HUMAVIPS and by The Technology Agency of the Czech Republic project TE0102019. Any opinions expressed in this paper do not necessarily reflect the views of the European Community. The Community is not liable for any use that may be made of the information contained herein.

References

[1] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer Science+Bussiness Media, New York, 2006. 1, 2

[2] L. Cerman and V. Hlaváč. Tracking with context as a semi-supervised learning and labeling problem. In *Proceedings of the International Conference on Pattern Recongnition*, pages 2124–2127. IEEE Computer Society, 2012. 3, 6

[3] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39:1–38, 1977. 2

[4] J. B. MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, pages 281–297. University of California Press, 1967. 1

[5] M. I. Schlesinger and V. Hlaváč. *Ten Lectures on Statistical and Structural Pattern Recognition*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2002. 2

[6] M. Šonka, V. Hlaváč, and R. D. Boyle. *Image Processing, Analysis and Machine Vision*. PWS, Boston, USA, second edition, 1998. 2

[7] X. Wan and C. Kuo. Image retrieval with an octree-based color indexing scheme. In *Proceedings of International Symposium on Circuits and Systems*, volume 2, pages 1357–1360. IEEE Computer Society, 1997. 2

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

FHSR: Face Hallucination based on Sparse Reconstruction

Yongchao Li *Cheng Cai
College of Information Engineering
Northwest A&F University
Shaanxi,China
ychao_lee@nwsuaf.edu.cn
*chengcai@nwsuaf.edu.cn

Guoping Qiu
School of Computer Science
University of Nottingham ,United Kingdom
qiu@cs.nott.ac.uk

Kin-man Lam
Department of Electronic and Information Engineering
Hong Kong Polytechnic University ,Hong Kong
enkmlam@polyu.edu.hk

Abstract. *Example based methods have been commonly used to achieve good performance on image super resolution (SR). In this paper, we propose a new face hallucination method, namely face hallucination based on sparse reconstruction (FHSR). Unlike many existed face hallucination methods such as the from local pixel structure to global image super resolution method (LPS-GIS) and the super resolution through neighbor embedding (SRNE), where the prior models are learnt employing the L_2-norm methods, but our FHSR framework aims to shape the prior model using sparse representation, which is solved by minimizing the L_1-norm. Then this learnt prior model is employed to guide the reconstruction process. Based on the assumption that using local image information only is sufficient to predict the missing high resolution (HR) details, our framework firstly uses the input single frame low resolution (LR) facial image to search the similar face images from a training dataset of LR-HR face image pairs. The searched HR example faces should possess similar local pixel structures to the input LR face. These selected HR face images are then warped to the input using optical flow. The local pixel structures are learnt from the warped HR faces using sparse representation. Finally, the learnt local pixel structures are applied to the input LR face to estimate the targeted HR face. Experimental results show that our framework is very flexible, and can achieve a competitive performance in terms of both reconstruction error and visual quality.*

1. Introduction

The idea of super resolution (SR) was first presented by Tsai and Huang [25], and significant progress has been made over the last three decades. Since SR is an ill posed problem, the prior constraints are necessary to attain the good performance in SR. Based on the different approaches to attain the prior constraints, methods for SR can be broadly classified into two classes. One is the conventional approach which reconstructs a HR image from a sequence of LR images of the same scene, also widely known as multi-image SR [9, 25] or regularization based SR. These algorithms mainly employ regularization model to solve the ill posed image SR and use smooth constraints as the prior constraints which are defined artificially. The other is single frame SR [5, 11, 12, 14, 19], which is also called learning based SR or example based SR and generates a HR image from a single LR image with the information learnt from a set of LR-HR training image pairs, and these algorithms attain the prior constraints between the HR images and the corresponding LR images through learning process. Many example based or learning based algorithms [5, 11, 12, 14, 16, 19, 21, 22] have been proposed in the field of image processing. Also in SR, Qiu [22] and Baker [3] have demonstrated that the smooth prior constraints used in many regularization

based methods will provide less and less help to S-R problem as the zooming factor increases, but example based approaches are promising to overcome this problem using advances in machine learning and computer vision. In this paper, we focus on the single image SR problem.

In a general framework of example based SR, the input LR image is first interpolated using the conventional methods to the size of the targeted HR image, then the input interpolated LR is used to be the initial estimation of the targeted HR. Here it is a blurry image and lack of HR information. The input LR image also is divided into the overlapping or nonoverlapping image patches. The example based framework will use the image patches to find the most matched examples through searching a training dataset of LR-HR image pairs, then the most matched HR examples is employed to learn HR information as the prior constraints. Finally, the learnt HR information and the input interpolated image are combined to evaluate the targeted HR image.

The idea of face hallucination was first proposed by Baker and Kanade [2], and was then used for SR problems in [12, 16, 19]. Example based face hallucination is a specific and an important object category of image SR. In [16], a two-step approach was developed for face hallucination, in which a Bayesian formula and a nonparametric Markov Network were employed to deal with face hallucination. Of all the various methods for face hallucination, Hu [12] is the first that explicitly and directly learn the local pixel structures of the most matched example images as reconstruction priors, and he proposed a three-stage face hallucination framework of Local Pixel Structure to Global Image Super resolution (LPS-GIS) in [12], in Stage 1, k pairs of example faces which have similar pixel structures to the input LR face are searched from a training dataset using k-Nearest Neighbor (KNN), then they are subjected to nonrigid warping using optical flow so that the corresponding targeted HR image can be reconstructed more accurately. In Stage 2, the LPS-GIS method learns the face structures, which are represented as coefficients using a standard Gaussian function; the learnt coefficients are revised according to the warped errors. In Stage 3, LPS-GIS forces the revised face structures namely the revised coefficients to the input LR face, and reconstructs the targeted HR image using an iterative method. However, in his LPS-GIS framework, the prior models are learnt using L_2-norm and it has been demonstrated that it is less robust and more sensitive to outliers which will usually decrease the accuracy of image SR. Our proposed FHSR framework aims to shape the prior model using sparse representation, which is solved by minimizing the L_1-norm. This prior model is then used to guide the reconstruction process. As for the organization of this paper, Section 2 provides an introduction to a basic concept which is called from local pixel structure with sparsity to face hallucination. The details of our framework are presented in Section 3, including the comparison of the L_1-norm and L_2-norm used to learn prior models. Section 4 presents the experiments and an evaluation of the proposed framework. Finally, the concluding remarks and future work are given in Section 5.

2. Local Pixel Structure with Sparsity

Firstly, an observed model between a HR face and its corresponding LR counterpart is given as

$$\mathbf{I_l} = \mathbf{I_h}\mathbf{H}S(r) + \mathbf{N}, \tag{1}$$

where $\mathbf{I_l}$ and $\mathbf{I_h}$ denote the LR and HR faces respectively, $\mathbf{H}$ represents a blurring filter operator and $S(r)$ is a subsampling operator with a scaling factor of r in every dimension[1] , and $\mathbf{N}$ is a noise vector such as the Gaussian white noise. Here, we ignore the blurring filter, so (1) can be rewritten as follows:

$$\mathbf{I_l} = \mathbf{I_h}S(r) + \mathbf{N}. \tag{2}$$

Therefore, the purpose of SR is to recover as much of the information lost in the down sampling process as possible. As we know, the HR face and the corresponding LR face have a common global face structure, so we can assume that they also have similar local pixel structures, and the local image information in the LR alone should be sufficient to predict the missing HR details. In our algorithm, we use the neighboring pixels of a missing pixel to estimate the targeted HR face. This idea is similar to neighbor embedding in [5, 11]. A formula is given to describe the model that used in our method:

$$I(x,y) = \sum_{\mu,\nu \in C} \alpha_{\mu,\nu}(x,y) \times I(x+\mu, y+\nu), \tag{3}$$

where $I(x,y)$ is a pixel at location (x,y), $\alpha_{\mu,\nu}(x,y)$ denotes the weight between the pixel $I(x,y)$ and its neighboring pixel $I(x+\mu, y+\nu)$ with a relative

[1]Here we just assume the scaling factor in each dimension is equal.

displacement of (μ, ν), μ and ν cannot be zero at the same time, and C denotes a local window whose center is at the pixel $I(x, y)$. Based on the above assumption of similar pixel structures between the HR-LR face pairs, the weights are almost the same at the same position of HR face and the corresponding LR one. Our algorithm searches for k similar LR examples to the input face from a dataset of LR-HR face pairs, meanwhile the neighboring weights of the pixel structures in the corresponding k HR example faces are utilized to estimate the information lost in the input LR face. In order to learn the embedded weights of the central pixel, Chang et al. [5], Gong et al. [11], and Hu et al. [12] used the L_2-norm methods such as Gaussian functions and least square methods. We believe that a good visual quality image is very sharp due to sharp edges, high frequency information and discontinuities, the sharp image means the local pixel structure has sparse property and this can be interpreted that the pixel $I(x, y)$ in (3) can be better reconstructed with only a fraction of but not all neighboring pixels, so in this concept, our algorithm will use sparse representation [24, 29] as the prior model to learn the embedded weights.

In [32, 33, 34], Yang et al. have imported sparse representation to SR and face hallucination problems, our method has some similarities to [32, 33, 34]. We both construct the methods based on the concept in sparse signal representation which suggests that the linear relationships among HR signals can be accurately recovered from their low dimensional projections. Also we both use LR image's structure to get a sparse prior model and then use this sparse model to reconstruct the HR image or HR patches. But the difference is that Yang et al. believe image patches can be well represented as a sparse linear combination of elements from an appropriately chosen overcomplete dictionary while in our method, a pixel can be well represented as a sparse linear combination of elements from its neighboring pixels. They seek a sparse representation for each patch of the LR input from an LR overcomplete dictionary, then they use the coefficients of this representation to generate the HR output. One important process is the training of two dictionaries for the LR and HR image patches. While in our method, we only have LR dictionary constituting with neighboring pixels of input. We use the sparse local pixel structure to ensure the reconstruction of the output HR.

Sparse representation has gained a great deal of attention recently. It is based on the assumption that most or all signals can be represented as a linear combination using only a small number of elementary signals, called atoms, belonging to an overcomplete dictionary. Compared to other conventional methods, sparse representation can usually offer a better performance with its capacity for efficient signal modeling [13]. The sparse representation of signals has already been applied in many domains, such as object recognition [24, 29], text categorization [23], signal classification [13], etc.

In sparse representation, a common formula for the problem of finding the sparse representation of a signal in a given overcomplete dictionary is described as follows:

$$\hat{x}_0 = \min \|x\|_0 , \qquad s.t. \qquad y = \mathbf{A}x, \quad (4)$$

where $\mathbf{A}$ is an $M \times N$ matrix whose columns are the elements of the overcomplete dictionary with $M < N$, and $y \in R^{\mathrm{M}\times 1}$ is an observational signal. The purpose of sparse representation is to find an $N \times 1$ coefficient vector x, which is considered to be a sparse vector, i.e. most of its entries are zero, except those whose elements in the overcomplete dictionary $\mathbf{A}$ are associated with the observational signal y. $\|x\|_0$ is the L_0-norm, and essentially it is equivalent to the number of non-zero components in the vector x. In fact, if the columns of $\mathbf{A}$ are of a general case, i.e. nonorthogonal, then a random vector x is the unique sparsest solution as long as it has less than $M/2$ non-zero components [7]. However, solving the sparsest solution for (4) has been found to be NP hard, and it is even difficult to approximate [1]. In other words, there is no known process which is more efficient than an exhaustive search of all the possible x in solving the sparsest solution.

With the rapid development of the theories for compressed sensing [4, 6], it has been discovered that if the vector x in (4) is sparse enough, then the solution to the L_0-norm in (4) can be replaced by the solution to the L_1-norm problem as

$$\hat{x}_1 = \min \|x\|_1 , \qquad s.t. \qquad y = \mathbf{A}x, \quad (5)$$

In fact, as long as the number of non-zero components in x0 is a small fraction of the dimension M, then the L_1-norm can replace and recover the L_0-norm efficiently [29]. In addition, the optimization problem of the L_1-norm can be solved in polynomial time [8, 30]. However, in real applications, the data in the overcomplete dictionary $\mathbf{A}$ are noisy. This will

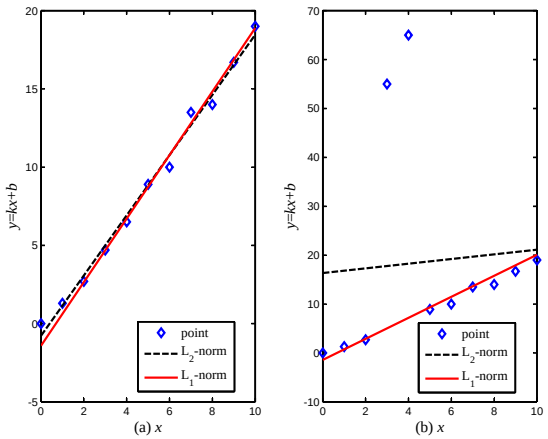


Figure 1. Fitting a line to 11 given points using the L_1-norm and L_2-norm: (a) without outliers, and (b) with two outliers.

lead to the result that the sparse representation of an observational signal in terms of the training data in $\mathbf{A}$ is not very accurate. In order to deal with the problem, formula (5) can be relaxed to a modified form as

$$\hat{x}_1 = \min \|x\|_1 , \quad s.t. \quad y = \mathbf{A}x + z, \qquad (6)$$

where $z \in R^{\mathrm{M}\times 1}$ is a noise vector with bounded energy $\|z\|^2 \leq \varepsilon$. Thus, (6) can be rewritten as follows:

$$\hat{x}_1 = \min \|x\|_1 , \quad s.t. \quad \|y - \mathbf{A}x\|_2^2 \leq \varepsilon, \qquad (7)$$

Lagrange multipliers offer an equivalent formula

$$\min_x \|\mathbf{A}x - y\|_2^2 + \lambda \|x\|_1 , \qquad (8)$$

where $\lambda \in R^+$ is a regularization parameter which balances the sparsity of the solution and the fidelity of the approximation to y. This is actually a typical convex optimization problem, and it can be efficiently solved using the method of Large Scale L1-Regularized Least Squares (L1LS) [17].

In the previous section, we have stated that many face hallucination methods use the L_2-norm to learn the prior model and we believe the local pixel structure in a good visual quality image has sparse property that means in all the neighboring pixels of pixel $I(x, y)$ in (3), many neighboring pixels can be regarded as outliers. It has been demonstrated in [15, 18, 29], compared to the L_1-norm, the L_2-norm is less robust and is more sensitive to outliers which will usually decrease the accuracy of image SR. Furthermore, in [18, 29], the L_1-norm has given more robust and better results on SR and face recognition respectively, especially when the signal is sparse and discontinuous. Here, we will give a simple example to illustrate this point that L_1-norm is more robust to outliers. Suppose 11 points $\{(x_i, y_i)|i = 0, ..., 10\}$ are given. Here we want to fit a line model $y = kx + b$ to these points, then, the L_1-norm and L_2-norm are utilized to solve this model. Fig. 1(a) shows the L_1-norm and L_2-norm both produce a similar estimation when there are not any outliers in the input data. However, when there are two outliers in the input data, the results are different. In Fig. 1(b), where two outliers are given, the L_1-norm still produces similar result, while it can be obviously seen that the outliers affect the result of L_2-norm seriously and lead to a divergent result. It demonstrates methods of L_1-norm are more robust to outliers and this toy example make us believe that in the condition of sparse local pixel structure, sparse representation can provide a better performance in face hallucination.

3. Detailed Procedure of the Framework

Fig. 2 illustrates that three primary steps are involved in our proposed FHSR framework. In Step 1, the input LR face is used to search a face dataset and identify the k pairs of LR-HR example faces having the most similar local pixel structures to the input LR face using the PCA and the KNN methods. The k pairs of example faces are composed of k LR faces and their corresponding k HR ones. Then the k HR examples are employed and warped to the input LR face using optical flow to make the targeted HR face of the input more accurate. This process will produce k HR warped examples, and the warped errors will be used in the next step. In Step 2, the local-pixel structures, which are represented by the weights of the neighboring pixels, are learnt from the k HR warped example faces using sparse representation. Then, the accuracy of the weights is improved using the warped errors produced in Step 1. In Step 3, the weights of the neighbors are employed to estimate the targeted HR face using an iterative method. Following are the details of these three steps.

3.1. Step 1: Searching and Warping

In our FHSR framework, finding the example faces of the input LR face in a large face dataset is the first step. In the k pairs of example faces, the k LR example faces have the most remarkably similar pixel structures to the input LR face. In our experiments, the dataset contains 800 LR-HR face pairs, and they have all been aligned and normalized. The LR faces are also magnified to the size of the HR faces using

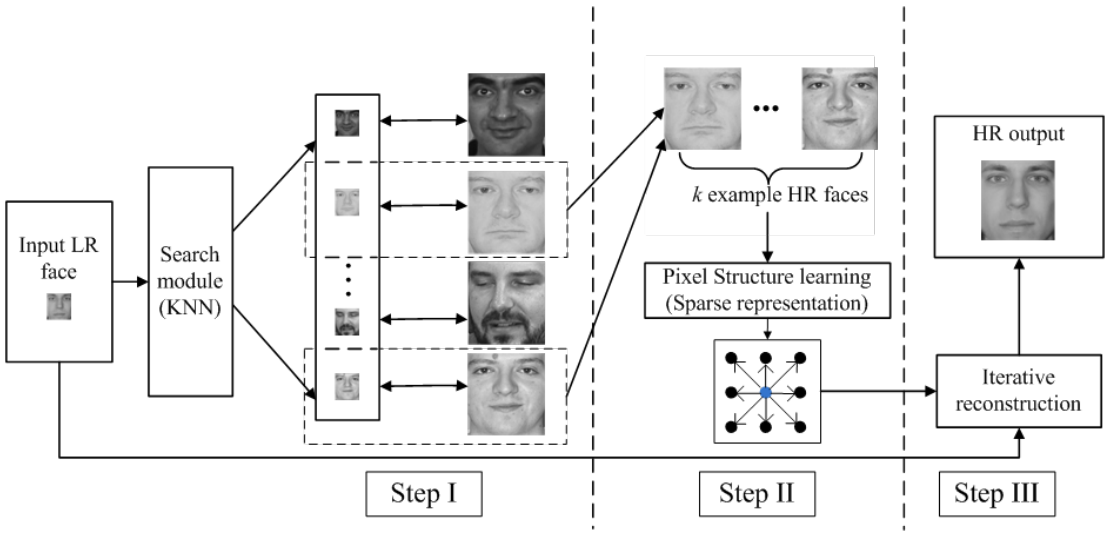


Figure 2. The implementation procedure for our proposed FHSR face hallucination framework.

bicubic interpolation. Principal component analysis (PCA) is used to represent the interpolated LR face images, and the KNN method is adopted to search the k LR example faces that are the most similar to the LR input face. Then, the selected corresponding k HR faces are used as example faces. A warping operation is employed to make the estimation more accurate. We use optical flow to warp the example faces. Optical flow has been used in SR in [12, 18]. In our paper, firstly, we derive the flow field between the input LR face and each of the k LR example faces, and then the corresponding HR example faces are warped accordingly based on the k flow fields, respectively. The purpose of the warping operation is to force the k HR example faces to have the most similar local pixel structures to the input LR face.

3.2. Step 2: Learning Neighboring Weights via Sparse Representation

Combining (3) and (4), we propose using the following formula to model the learnt local pixel structure:

$$z(x,y) = \sum_{j=1}^{p^2-1} \alpha_j(x,y)\mathbf{A}(:,j), \tag{9}$$

where $z(x,y) = [I^1(x,y), \cdots, I^k(x,y)]^T$ and k is the number of HR example faces, $\alpha_j(x,y)$ denotes the weight between the pixel $I(x,y)$ and its neighboring pixel $I(x+\mu, y+\nu)$ with a relative displacement of (μ,ν). The i^{th} row of the matrix $\mathbf{A}$ in (8) and (9) are the neighboring pixels of the central pixel $z(x,y)$ of the i^{th} example face, and the neighborhood size is $p \times p$:

$$\mathbf{A} = [I^1_{\mu,\nu}(x+\mu, y+\nu) \cdots I^k_{\mu,\nu}(x+\mu, y+\nu)] \in R^{k\times(p^2-1)}. \tag{10}$$

We also assume that $x = [0, \cdots, 0, \alpha_{\mu,\nu}(x,y), \cdots, 0]^T$is the coefficient vector used in (8), whose entries are all zero, except those associated with the central pixel vector y. We use L1LS [17] to solve (8) with the matrix $\mathbf{A}$ and the central pixel vector y. After the coefficient vector x is computed using L1LS, a refinement procedure is performed according to the warped errors produced in the first step. x can be rewritten as $x = c_{x,y}x'$,where x' denotes the x in (8), and $c_{x,y}$ represents the refinement procedure. Then (9) can be rewritten as

$$z = c_{x,y}\mathbf{A}x' = c_{x,y}\hat{z} \tag{11}$$

where x' has been calculated using L1LS. The definition of $c_{x,y}$ is the same as in [12].

3.3. Step 3: Reconstructing the Targeted HR Face

At this step, the local structures have been learnt. The major task of this step is to force the local structures of the reconstructed face similar to fit those of the input interpolated LR face. An iterative method [27] is employed in this step. The pixels of the input LR image are used as the anchor points in the iterations. The targeted HR pixel values are confined within the range of 0-255, and are reconstructed

as

$$\triangle_t(x,y) = \hat{\mathbf{I}}_{\mathbf{h}}^{\mathbf{t}}(x,y) - \sum_{\mu,\nu \in C} \alpha(x,y)\hat{\mathbf{I}}_{\mathbf{h}}^{\mathbf{t}}(x+\mu, y+\nu), \tag{12}$$

$$\hat{\mathbf{I}}_{\mathbf{h}}^{\mathbf{t+1}}(x,y) = \hat{\mathbf{I}}_{\mathbf{h}}^{\mathbf{t}}(x,y) - g \triangle_t (x,y), \tag{13}$$

where $\triangle_t(x,y)$ is the regulation parameter between iterations, and g is a scale factor and set to be 0.05 in our experiment. Furthermore, in each iteration, we have $\mathbf{I_h}(x,y) = \mathbf{I_l}(\frac{x}{r}, \frac{y}{r})$, where r is the down-sampling factor in both dimensions in (2). The input interpolated LR face is selected as the initial estimate HR image. In our experiments, we set the number of iterations to be performed t at 200.

4. Experimental Results

In our experiments, the training face dataset which contains 800 images(100 for test and others for training) is selected from the GTFD [10] and the FERET databases [20]. The parameters k and p in (10) and λ in (8) were determined empirically through experiments. In our next experiments, we set $k = 9, \lambda = 0.03, p = 3$.

To measure the performance of our proposed face hallucination method, we first reconstructed HR face images with a magnification factor of 4. The size of the input LR faces was 31×27, and the size of the original faces was 124×108. All the face images in the dataset have been aligned using Wong's method [28] and normalized using an efficient illumination-normalization technique [31]. Our proposed algorithm is compared to Hu's method [12] and Chang's method [5], as well as bicubic interpolation. The reconstructed HR faces of five randomly selected input LR faces using the different methods are shown in Fig. 3. It is obvious that the results using bicubic interpolation are the most blurry, while the others can provide much better results, especially Hu's method and our proposed method. To be specific, Hu's method and our proposed method achieve better performance in the eyes, mouth and eyebrows, and our method can achieve even better results in edge regions than Chang's and Hu's methods. This is mainly due to the fact that the L_1-norm is more robust than the L_2-norm when there are outliers in the data. Next, we measured the performance of the different methods in terms of PSNR (Peak Signal to Noise Ratio) and SSIM (Structural Similarity Index) [26] using test samples, and the results of some samples are shown in Tab. 1. Our method outperforms the other methods in terms of PSNR and SSIM. As mentioned above, Hu's method is superior to Chang's method, and the bicubic interpolation produces the worst results. The results shown in Fig. 4 and Tab. 2 demonstrate that our method can achieve the best performance in terms of both visual quality and reconstruction error.

	PSNR(dB)	SSIM
Bi-cubic interploation	25.589	0.7659
Hu's method	27.850	0.8448
Chang's method	27.317	0.8203
Our FHSR method	**28.901**	**0.8638**

Table 1. Average PSNR and SSIM of different methods, with magnification factor = 4.

	PSNR(dB)	SSIM
Bi-cubic interploation	23.372	0.7278
Hu's method	26.790	0.8086
Chang's method	23.342	0.7176
Our FHSR method	**27.734**	**0.8156**

Table 2. Average PSNR and SSIM of different methods, with magnification factor = 6.

We also measured the performances of the different methods when the magnification factor was increased to 6. The sizes of the HR and LR images were 126×108 and 21×18, respectively. Fig. 4 shows the HR faces reconstructed using the different methods, and the corresponding average PSNR and SSIM are tabulated in Tab. 2. We can see that the performance of Chang's method degrades significantly, and is even worse than bi-cubic interpolation. However, our proposed method can still retain a steady performance in terms of both visual quality and reconstruction error and it demonstrates that our proposed FHSR framework can deal with larger magnification in face hallucination problem.

5. Conclusion

In this paper, we have proposed a method, namely FHSR, for face hallucination. Our method employs neighborhood embedding, and the neighbor weights are learnt using sparse representation, i.e. L_1-norm, instead of using L_2-norm. We also apply optical flow to align images so as to make the estimation more accurate. We compare our method with some other state of the art SR methods, and the experimental results shows that our method is competitive and can achieve the best performance. The primary reason of

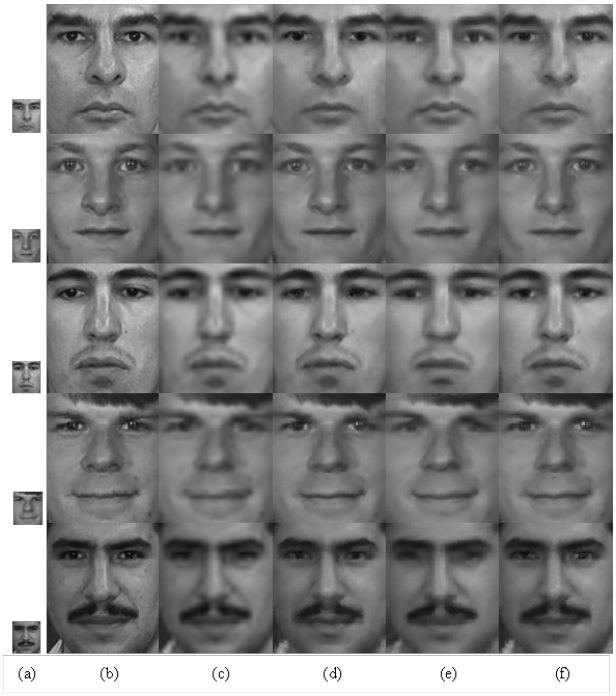


Figure 3. HR faces reconstructed using different methods with a magnification factor of 4: (a) the input LR faces; (b) the original HR faces; (c) bicubic interpolation; (d) Hu's method; (e) Chang 's method; and (f) our proposed FHSR framework.

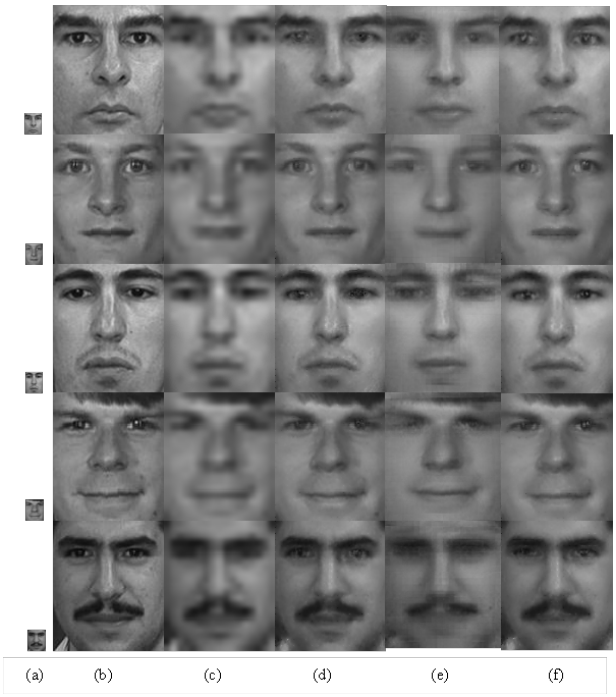


Figure 4. HR faces reconstructed using different methods with a magnification factor of 6: (a) the input LR faces; (b) the original HR faces; (c) bicubic interpolation; (d) Hu's method; (e) Chang 's method; and (f) our proposed FHSR framework.

the superior performance of our proposed method is that it can estimate the local pixel structures of targeted HR faces more accurately from the example faces using sparse representation. In future work, we would like to improve and enrich our work in the following aspects. First, find a new image alignment method to improve the performance of our framework , as it is proposed based on the face local-pixel structures, so an efficient and accurate alignment method for the face images is very important. Second, we plan to apply our framework to other SR problems.

Acknowledgements

This work is supported by project 61202188 supported by National Natural Science Foundation of China, projects 2010JQ8019 and 2010K06-15 supported by Natural Science Foundation of Shaanxi Province, China, project QN2009093 supported by the Special Foundation for Basic Scientific Research of Central Universities, China.

References

[1] E. Amaldi and V. Kann. On the approximability of minimization nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, 209:237–260, 1998. 3

[2] S. Baker and T. Kanade. Hallucinating faces. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 83–88, 2000. 2

[3] S. Baker and T. Kanade. Limits on super-resolution and how to break them. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1167–1183, 2002. 1

[4] E. Candès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, 2006. 3

[5] H. Chang, D. Yeung, and Y. Xiong. Super-resolution through neighbor embedding. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 275–282, 2004. 1, 2, 3, 6

[6] D. Donoho. For most large underdetermined systems of linear equations, the minimal l1-norm solution is also the sparsest solution. *Communication on Pure and Applied Mathematics*, 59(6):797–829, 2006. 3

[7] D. Donoho and M. Elad. Optimal sparse representation in general (non-orthogonal) dictionaries via l1 minimization. In *Proceedings of the National Academy of Sciences of the United States of America*, pages 2197–2202, 2003. 3

[8] D. Donoho, Y. Tsaig, I. Drori, and J. Starck. Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit. *IEEE Transactions on Information Thery(submitted)*, 2006. 3

[9] S. Farsiu, M. Robinson, M. Elad, and P. Milanfar. Fast and robust multiframe super resolution. *IEEE Transaction on Image Processing*, 13(10):1327–1344, 2004. 1

[10] Georgia Tech Face Database. `http://www.anefian.com/research/face_reco.htm`. 6

[11] M. Gong, K. He, J. Zhou, and J. Zhang. Single color image super-resolution through neighbor embedding. *Journal of Computational Information Systems*, 7:49–56, 2011. 1, 2, 3

[12] Y. Hu, K. Lam, G. Qiu, and T. Shen. From local pixel structure to global image super-resolution: A new face hallucination framework. *IEEE Transaction on Image Processing*, 20(2):433–445, 2011. 1, 2, 3, 5, 6

[13] K. Huang and S. Aviyente. Sparse representation for signal classification. In *presented at the Neural Information Processing Systems*, 2006. 3

[14] K. Kim and Y. Kwon. Example-based learning for single-image super-resolution. *Computer Science Pattern Recognition*, 5096:456–465, 2008. 1

[15] N. Kwak. Principle component analysis based on l1-norm maximization. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 30(9):1672–1680, 2008. 4

[16] C. Liu, H. Shum, and C. Zhang. A two-step approach to hallucinating faces:global parametric model and local nonparametric model. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 192–198, 2001. 1, 2

[17] X. Mei, H. Ling, and D. Jacobs. Sparse representation of cast shadows via l1-regularized least squares. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2009. 4, 5

[18] D. Mitzel, T. Pock, T. Schoenemann, and D. Cremers. Video super resolution using duality based tv-l1 optical flow. *Computer Science Pattern Recognition*, 5748:432–441, 2009. 4, 5

[19] J. Park and S. Lee. An example-based face hallucination method for single-frame, low-resolution facial images. *IEEE Transaction on Image Processing*, 17(10):1806–1816, 2008. 1, 2

[20] P. Phillips, H. Wechsler, J. Huang, and J. Rauss. The feret database and evaluation procedure for face-recognition algorithms. *Image and Vision Computing*, 16:295–306, 1998. 6

[21] G. Qiu. A progressively predictive image pyramid for efficient lossless coding. *IEEE Transaction on Image Processing*, 8(1):109–115, 1999. 1

[22] G. Qiu. Inter-resolution look-up table for improved spatial magnification of image. *Journal of Visual Communications and Image Representation*, 11:360–373, 2000. 1

[23] T. Sainath, S. Maskey, D. Kanevsky, B. Ramabhadran, D. Nahamoo, and J. Hirschberg. Sparse representations for text categorization. In *Proceedings of INTERSPEECH'2010*, pages 2266–2269, 2010. 3

[24] G. Sivaram, S. Ganapathy, and H. Hermansky. Sparse auto-associative neural networks: Theory and application to speech recognition. In *Proceedings of INTERSPEECH'2010*, pages 2270–2273, 2010. 3

[25] R. Tsai and T. Huang. Multiframe image restoration and registration. *Advances in Computer Vision and Image Processing*, 1(2):317–339, 1984. 1

[26] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transaction on Image Processing*, 13(4):600–612, 2004. 6

[27] P. Wesseling. *An Introduction to Multigrid Methods*. R T Edwards, 1992. 5

[28] K. Wong, K. Lam, and W. Siu. An efficient algorithem for human face detection and facial feature extraction under different conditions. *Pattern Recognition*, 34:1993–2004, 2001. 6

[29] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009. 3, 4

[30] S. Wright, R. Nowak, and M. Figueiredo. Sparse reconstruction by separable approximation. In *Proceedings of ICASSP'2008*, pages 3373–3376, 2008. 3

[31] X. Xie and K. Lam. An efficient illumination normalization method for face recognition. *Pattern Recogniton Letter*, 27:609–617, 2006. 6

[32] J. Yang, H. Tang, Y. Ma, and T. Huang. Face hallucination via sparse coding. In *IEEE International Conference on Image Processing*, pages 1264–1267, 2008. 3

[33] J. Yang, J. Wright, T. Huang, and Y. Ma. Image super-resolution as sparse representation of raw image patches. In *IEEE Computer Vision and Pattern Recognition(CVPR)*, 2006. 3

[34] J. Yang, J. Wright, Y. Ma, and T. Huang. Image super-resolution via sparse representation. *IEEE Transaction on Image Processing*, 19(11):2861–2873, 2010. 3

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Logical Layout Recovery: Approach for Graphic-based Features

Aysylu Gabdulkhakova, Tamir Hassan, Walter G. Kropatsch
Pattern Recognition and Image Processing Group
Technische Universität Wien
Favoritenstraße 9-11, 1040 Wien, Austria
{aysylu,tam,krw}@prip.tuwien.ac.at

Abstract. *In contrast to the existing approaches for document analysis and understanding this paper presents a system that determines the logical role of vector-graphic objects in predominantly textual, natively digital PDF documents. This work was inspired by the idea of recognizing structural graphic objects in order to clarify the logical layout of mostly graphical documents, even those of a complex nature. Based on visual perception, geometric features and spatial relations, the proposed statistical method distinguishes illustrative graphic objects from structural graphic objects. We evaluated the method on two document domains – newspapers and technical manuals – and found the results to be reliable. We propose using logical information about graphic objects to be a new step towards domain-independent document understanding systems.*

1. Introduction

A human reader can easily rediscover the logical structure of any document from text properties (typesetting conventions) and layout. In ambiguous cases a human can additionally follow the meaning of the text paragraphs.

Document analysis and understanding systems presented in the literature are focused on textual data in domain-specific documents (books, business letters, scientific papers, technical specifications). Such systems determine the logical structure – headings, paragraphs, reading order–based on individual properties of a given document class. In these works graphic regions are detected as non-textual and no further processing or analysis for them is performed [1, 3, 2, 6]. The problem stems from a need to create a system capable for a broad class of documents and to reuse or repurpose the document content, represented by graphic objects. The purpose of this paper is to draw attention to the utilization of graphical objects for more reliable logical structure recognition.

Natively digital PDF documents (*PDF Normal* or *Formatted Text and Graphics*) often do not contain explicit logical structure information and define vector-graphic objects by a set of instructions for rendering low-level primitives, such as lines, rectangles, curves and glyphs. In such documents vector-graphic objects can be a part of a graphic region, table, letter or ruling line (Figure 1).

Hereafter *structural element* is a line or rectangle that is used as a visual separator of logical blocks in a document. Other graphic objects are respectively called *non-structural elements*.

We class documents into three types according to their layout complexity:

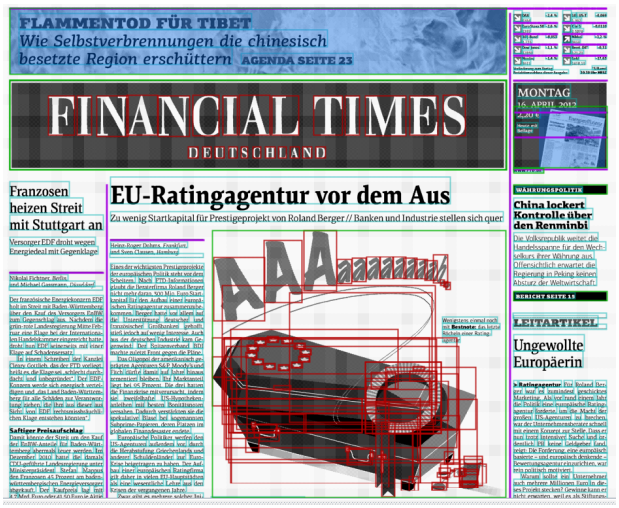
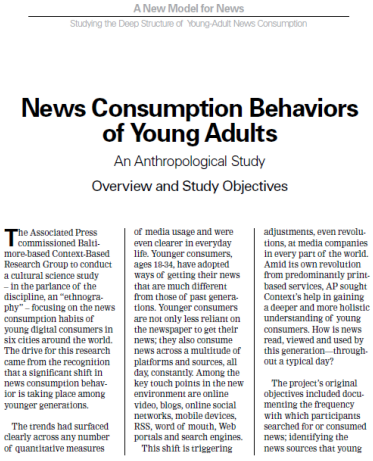


Figure 1. The result of parsing instructions of natively digital PDF page.


A New Model for News

Studying the Deep Structure of Young-Adult News Consumption

News Consumption Behaviors of Young Adults

An Anthropological Study

Overview and Study Objectives

The Associated Press commissioned Ralt-mores-based Context-Based Research Group to conduct a cultural science study – in the parlance of the discipline, an "ethnogra-phy" – focusing on the news consumption habits of young digital consumers in six cities around the world. The drive for this research came from the recognition that a significant shift in news consumption behav-ior is taking place among younger generations.

The trends had surfaced clearly across any number of quantiative measures of media usage and were even clearer in everyday life. Younger consumers, ages 18-34, have adopted ways of getting their news that are much different from those of past genera-tions. Younger consumers are not only less reliant on the newspaper to get their news; they also consume news across a multitude of platforms and sources, all day, constantly. Among the key touch points in the new environment are online video, blogs, online social networks, mobile devices, RSS, word of mouth, Web portals and search engines. This shift is triggering adjustments, even revolu-tions, as media companies in every part of the world. Amid its own revolution from predominantly print-based services, AP sought Context's help in gaining a deeper and more holistic understanding of young consumers. How is news read, viewed and used by this generation – through-out a typical day?

The project's original objectives included docu-menting the frequency with which participants searched for or consumed news; identifying the news sources that young

Figure 2. Example of a mostly textual document with trivial layout


Bresciaoggi

Dinamica

Strage senza colpevoli

Morte in campo, il calcio si ferma

Alle parti civili restano le spese

In auto contro un palo perde la vita a 29 anni

La verità è il filo sottile che non va spezzato

Figure 3. Example of a predominantly textual document with non-trivial layout

Figure 4. Example of a complex ambiguous document

- simple, mostly textual documents: determination of logical layout is relatively easy for both humans and automated algorithms, e.g., scientific papers (Figure 2);
- complex predominantly textual documents: layout analysis is more difficult for automated algorithms; however it can still be performed by understanding text-based layout conventions, e.g., newspaper pages (Figure 3);
- complex, mostly non-textual documents: layout analysis requires understanding of the graphical content, making domain-generic solutions unfeasible, e.g., creative design of magazines (Figure 4).

The paper presents an object-based approach for analysis and understanding of vector graphic objects that appear in predominantly textual natively digital PDF documents. Our heuristic rule-based method considers logical geometrical properties and mutual arrangement between the graphic and text objects. This approach enables grouping low-level primitives into higher-level logical blocks and finding structural graphic elements.

The remainder of this paper is organized as follows: in Section 2 we provide an overview to state-of-the-art research related to the problem of logical structure discovery. Section 3 describes in detail a method for analysis of vector graphic content. Section 4 shows the evaluation of the above method and discussion of the obtained results. Finally, Section 5 presents conclusions and directions for the future work.

2. Related work

State-of-the-art methods provide various solutions to the problem of logical structure discovery. Most of them take into account image-based features and deal with document images rather than electronic documents. As our system processes natively digital PDF documents, this section focuses on approaches using PostScript instructions as a starting point.

A groundbreaking method for object-based analysis was published by Lovegrove and Brailsford [9]. Their approach attempts at segmenting bottom-up text regions and determining their logical role – heading, title, main text etc. – according to the decision of a blackboard system combined with logical-relationship rules. Although this methodology is text-oriented, for the future work authors propose to take advantage of "graphic elements such as article or column separators" for document understanding purposes. Anjewierden [1] created a system, AIDAS, which builds physical blocks in a bottom-up fashion and determines their logical role top-down using shallow grammars. The system was tested on three domains of technical manuals provided by industrial partners. Despite its extendability to other domains by adding extra rules, the major drawback of the proposed solution is its domain-specific nature. Chao and Fan [3] developed a method for information extraction in scientific papers. It uses both object-based and

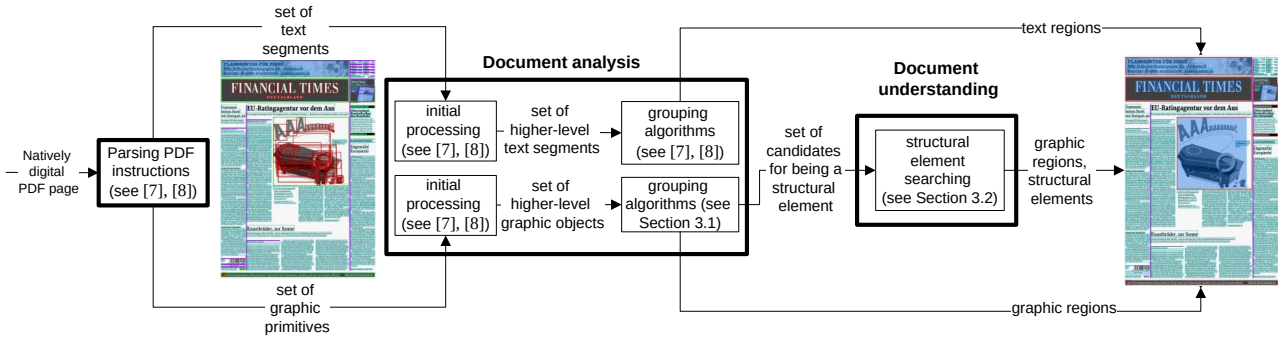


Figure 5. Main steps of the presented algorithm for processing natively digital PDF page

image-based approaches for establishing higher-level vector graphic entities. These entities are obtained by coordinate comparison of potential graphic regions in a document image combined with analysis of path objects extracted from the PDF content stream. Nevertheless there is no explicit utility of the obtained 'logical blocks' during logical structure discovery. Déjean and Meunier [4] present a system that applies XY-cuts and grouping according to the spatial closeness of objects for external and path objects. The purpose is to compose thousands of low-level primitives into physical blocks. Further processing for determining a logical role for each block is not implemented. Bloechle et al. [2] present an object-based system, Dolores (**Do**cument **Lo**gical **Res**tructuring), for restructuring textual and graphical content. The idea is based on using artificial neural networks which are trained to recognize the logical layout of the newspapers. Hassan [7] developed a system, PDF Extraction Toolkit[1], that includes an object-based bottom-up method for extracting logical text blocks, vector-graphics objects (lines, curves, rectangles) and bitmaps. The GUI shows the rectangular bounding boxes of the detected component groups, although the primitives themselves may not be rectangular in shape.

Overall the publications mentioned above do not describe the processing of graphical primitives in sufficient detail. They also do not appear to distinguish between structural and non-structural objects and use this information for document understanding tasks. We believe that this distinction is a powerful feature that document understanding systems can use for logical structure recovery.

This idea was roughly implemented by Gao et al. [6]. Their image-based method aims at extracting structural information from PDF book documents. In one of the processing steps they find separation lines that visually distinguish different parts of the documents. In contrast, our approach is oriented for predominantly textual documents, such as newspapers, which provide a rich variety of layouts and different types of structural elements– lines, rectangles, bitmaps.

We decided to advance the PDF Extraction Toolkit from the point of analysis and understanding of vector graphics. The description of the new methods can be found in [5], as well as in the next section.

3. Methodology

The methodology in this section is largely identical to that introduced in [5] and is described here in detail for consistency and completeness.

The entire system *pdfXtk* performs analysis and understanding of natively digital PDF documents in three steps (Figure 5): parsing the PDF content stream, document analysis (defining a physical structure of a document), and document understanding (defining a logical structure of a document). As a result, the given PDF document is represented as a set of text regions, graphic regions and structural elements.

On the first step the page content is extracted from PDF instructions and transformed into Java object primitives. The location of these primitives on the page is defined by their bounding box coordinates in 2D Cartesian space. In *pdfXtk* we store the following types of primitives: line segments, rectangles, bitmaps, text segments (text blocks of typically 2-3 characters). The second step includes the bottom-up segmentation methods for text and graphic objects, which are based on the principles of visual cognition. In the final processing phase, we determine which of the graphical objects represent structural elements,

[1] pdfXtk: http://pdfxtk.sourceforge.net

as opposed to graphic regions. The result of these algorithms is a representation of a document as a set of text regions, graphic regions and structural elements.

The remainder of this section describes our grouping rules and methods for determining whether a vector object is structural. The task of processing the text blocks is not addressed in this paper (see [8, 7] for a description).

3.1. Grouping rules

The devised grouping rules are based on principles of visual cognition of commonly occurring layout constructions. They take into account the geometrical properties of the graphic objects as well as their mutual spatial arrangement. We classify these rules into two categories: intersection-based and distance-based. When applied in combination with each other, they enable higher-level objects to be constructed, which usually correspond to distinct logical objects in the document's structure.

3.1.1 Based on intersections

Lines. Two lines are grouped together, if the intersection between them is established. For the next line, we check the intersection with each member of the group. The exception case happens, when these lines construct a solid line or visually are perceived as a single object.

Rectangles. This method is applicable not only to rectangles (filled or non-filled), but also to bitmap objects and complex figures (in the latter case, the bounding box is used). It is based on the assumption that two structural rectangle objects on the page are unlikely to intersect. Specifically, when the topmost coordinate of one rectangle is less than the bottommost coordinate of the other or when the leftmost coordinate of one rectangle is greater than the rightmost coordinate of the other.

Often advertisements or other separated content is enclosed in structural rectangles, which are very close to each other or even overlap. Hence, before merging such elements using the above rule, we check whether they enclose further objects. If yes, then the given pair of rectangles is not grouped.

Line and Rectangle. In predominantly textual documents, two types of intersection between line and rectangle objects can occur:

1. structural line intersects the rectangular object;

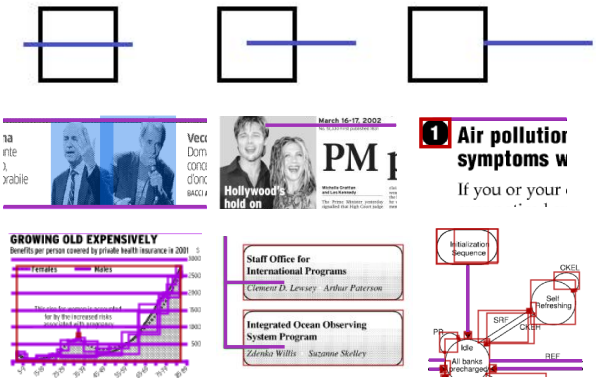


Figure 6. Intersection-based algorithm for line and rectangle. First row: three considered cases of intersection; second row: examples of the above cases with structural lines; second row: examples of the above cases with non-structural lines

2. line segment intersects the rectangular object, both are part of a graphic region.

Structural lines usually appear to intersect rectangular objects in headlines, headers and footers. Here the length of the horizontal or vertical structural line is in large excess over the width or height of a rectangular object correspondingly. On the other hand, in a graphic region lines and rectangles are approximately of the same size. For the purpose of distinguishing these two cases three actions are sequentially performed:

a) the width ratio or height ratio, whichever is the larger, is compared to the given threshold;

b) we determine the type of intersection or, more precisely, the mutual arrangement of the intersecting objects (see Figure 6);

c) the ratio between both parts of the line, split at the intersection point, is compared to a given threshold.

Line and Text. There are a variety of ways in which line and text fragments can intersect each other. In our research we focused on four cases that commonly occur in newspapers:

a) line underscores text block;

b) line crosses word;

c) lines form the axes and text blocks represent labels (as in a chart or diagram);

d) text block is surrounded by lines, which distinguish it from other parts of a document.

Cases a) and b) can be distinguished from each other by the distance between their centres projected on the Y-axis: if the line is closer to the centre of the text bounding box than to its border, then case a) applies; otherwise case b). Cases c) and d) can also be distinguished by the distance between their centres, but projected on the X-axis: if the line is touching or intersecting the text bounding box, then case c) applies; otherwise case d).

Rectangle and Text. As in the previous paragraph there are several possibilities of intersection between the given objects. More precisely:

a) rectangle encloses text fragment;

b) rectangle intersects text fragment;

c) rectangle slightly touches text fragment.

The last case occurs in tight layouts, where the rectangular bounding boxes of neighboring components often slightly overlap each other.

3.1.2 Based on distance

Lines. Here the distance-based rules consider the possibility of dashed lines. Line segments represent small objects with the distance between them less than or equal to the element size.

Rectangles. Two rectangles are considered as a single object if the distance between their centres is less than a given threshold. This threshold depends on two parameters: a granularity-level coefficient and the widths or heights of the rectangles. It is calculated by multiplying the first parameter with the sum of the second.

The granularity-level coefficient depends on a size ratio of the given rectangles and is divided into 3 types: high (0.55), normal (0.6) and low (0.65). These numerical values were obtained experimentally. Next, a mutual arrangement between two rectangles is defined as one of three possible cases: they lie in a horizontal line, in a vertical line, neither in a horizontal nor in a vertical line. Depending on this arrangement the threshold distance is then computed using the sum of rectangles' heights, widths or both – widths and heights – correspondingly. In the last case two threshold distances are computed. Finally, the threshold distance is compared to the distance between the centres of the given objects projected on the appropriate axis – X-axis, Y-axis, both axes correspondingly.

It is worth noting that our system represents composite objects by their rectangular bounding box. In such a manner, glyphs that form parts of logos, newspaper headings, etc. are introduced as rectangular objects. A vivid example of the above glyphs is the heading of newspapers such as The Sydney Morning Herald, International Herald Tribune, etc. Here, low-level primitives are positioned sequentially in one row/column. In order to detect this case, we refer to the golden ratio font rules [2].

Errors can occur with advertisements that have the same size and are close to each other (Section 3.1.1, Rectangle and Rectangle). These advertisements differ from glyphs as they also contain text. Moreover, the advertising boxes are usually filled with objects as large as at least one third of their size, whereas glyphs have a negligibly small filled area.

3.2. Finding structural elements

Line. Generally a structural line occurs as a horizontal/vertical line or rectangle that looks like a line, which does not intersect other non-structural objects (strokes, lines, rectangles, merged graphic regions, text fragments) and is not enclosed in any graphic region. These features represent preliminary conditions for finding a list of structural line candidates. Next, for each candidate the closeness and relation to text fragments is taken into account.

As an example, in Section 3.1.1, paragraph *Line and Text* we considered four cases. Case a) – line underscores text block – is neither structural nor illustrative, but rather an integral part of the formatting of the text. In contrast case b) – line crosses a word – the line is likely to form a part of an illustrative region. In cases c) and d) there is a pair of identical groups of straight lines with different semantics: in case c) these lines are a part of chart, whereas in case d) the lines are used as a barrier' and serve the purpose of separating the the text paragraph from the remaining page content. Thereby we conclude that c) is an example of non-structural lines at the same time as d) is an example of structural lines.

Rectangle. Generally, a structural rectangle does not intersect other graphic primitives and regions, but can enclose them. The special case is that of "stand-alone" rectangle objects, which often look like and serve the same logical function as lines.

4. Evaluation

The aforementioned state-of-the-art methods either do not differentiate graphic objects of a document into structural and non-structural or do not provide any experimental results that can be further used for the reasonable comparison.

For the purpose of evaluation we created an interactive evaluation tool and performed estimation of the obtained results on a bitmap level using similarity measures from [10].

The system performs several tasks: ground truth image generation, resultant image generation and comparison of the above images. As input it takes a binary image of a given PDF document at a fixed resolution, 72dpi, which is sufficient for this purpose. Ground truth image generation is obtained by manually marking the appropriate logical graphic regions on a binary image. The resultant image is produced by automatic mapping of the output XML file to the binary image. Each logical type of a graphic object(structural line, structural rectangle, illustrative region) is represented by a specific color. Correspondence between two images is established via pixel-by-pixel color-value comparison. The interface of the system is shown in Figure 9.

For the purpose of defining the measures that determine the nature of overlapping between the regions, we borrow ideas from the approach proposed in [10]:

- *correct detection* – regions are mainly overlapping ;
- *partial detection* – some overlapping detected, but not sufficient as in the first case;
- *over-segmentation* – a single object in the ground truth is detected as two separate segments;
- *under-segmentation* – two segments in the ground truth are erroneously merged in the algorithm's output;
- *incorrect detection* – the types of region in ground truth and result are different (structural rectangle, structural line or graphic region);
- *false positives* – the region is marked by the algorithm, but does not occur in the ground truth;
- *missed objects* – the region is marked in the ground truth, but has not been detected by the algorithm.

4.1. Experimental results

Our approach was tested on predominantly textual electronic PDF documents from two domains: newspapers and technical manuals. For the newpaper dataset we took 140 pages from 13 different newspapers taken from 15-17 April 2012, namely *Nuovo Quotidiano di Rimini, The Wall Street Journal, El Mundo del Siglo XXI, China Daily, Il Tirreno, Die Tageszeitung, il Giornale, Le Monde, L'Eco di Bergamo, Äripäev, La Gazzetta dello Sport, International Herald Tribune, Bresciaoggi*; for the technical manual dataset we took 40 pages from the first 10 different manuals obtained by using a popular search engine. The results of our evaluation are given in Table 1 and Table 2 respectively.

4.2. Discussion

By testing on two datasets from different domains, we can see how the algorithm performs on different types of document. Newspapers provide a rich layout variety and sparse complex vector-graphic objects, such as drawings. Technical manuals, on the other hand, are represented by a simple logical layout and mostly include sophisticated figures. The algorithm demonstrates a high performance on predominantly textual, natively digital PDF documents from both domains.

An important drawback for the proposed algorithm is that PDF is based on the PostScript page description language. A limited number of rendering instructions can lead to an unexpected set of underlying operator structures even for a simple page layout. Although such documents are easily understood by humans (Figure 7), this can cause problems for algorithms that work directly on the operator level (Figure 8).

5. Conclusion and further work

This paper addresses the problem of studying the properties of graphical content in documents, which can help us divide a page into logical blocks. At this point arises a task of differentiating graphic objects into structural and non-structural. A solution for this task, which is oriented to predominantly textual natively digital PDF documents, is presented. It includes algorithms for

grouping graphic primitives into higher-level logical blocks and for understanding their logical role in a document.

The efficiency and reliability of the system was tested on newspapers and technical manuals and achieved good results. Choosing these document domains was caused by the need to prove that the proposed heuristic rules perform well not only for documents from the newspaper domain, but also for technical figures and schemes. The current implementation is oriented to predominantly textual documents. For the future, we propose to extend our set of heuristic rules by considering the possibility of bitmap elements and text as being structural. This has the potential to lead to better results, particularly on documents with increased amounts graphical content.

Figure 7. Original newspaper page

Figure 8. Parsing the PDF content stream reveals two rectangles (highlighted in red) that cross each other on the left side at the center of the page. As a result, the whole page will be perceived as a non-structural graphic region.

Acknowledgements

This work was funded in part by the Austrian Federal Ministry of Transport, Innovation and Technology (Grant No. 829602).

References

[1] A. Anjewierden. Aidas: Incremental logical structure discovery in pdf documents. In *Document Analysis and Recognition, 2001. Proceedings. Sixth International Conference on*, pages 374–378. IEEE, 2001. 1, 2

[2] J. Bloechle, M. Rigamonti, K. Hadjar, D. Lalanne, and R. Ingold. Xcdf: a canonical and structured document format. *Document Analysis Systems VII*, pages 141–152, 2006. 1, 3

[3] H. Chao and J. Fan. Layout and content extraction for pdf documents. In *DAS 2004: Proceedings of the International Workshop on Document Analysis Systems*, pages 213–224, 2004. 1, 2

[4] H. Déjean and J. Meunier. A system for converting pdf documents into structured xml format. *Document Analysis Systems VII*, pages 129–140, 2006. 3

[5] A. Gabdulkhakova and T. Hassan. Document understanding of graphical content in natively digital pdf documents. In *Proceedings of the 2012 ACM symposium on Document engineering*, pages 137–140. ACM, 2012. 3

[6] L. Gao, Z. Tang, X. Lin, Y. Liu, R. Qiu, and Y. Wang. Structure extraction from pdf-based book documents. In *Proceedings of the 11th annual international ACM/IEEE joint conference on Digital libraries*, pages 11–20. ACM, 2011. 1, 3

[7] T. Hassan. Object-level document analysis of pdf files. In *Proceedings of the 9th ACM symposium on Document engineering*, DocEng '09, pages 47–55, New York, NY, USA, 2009. ACM. 3, 4

[8] T. Hassan. *User-Guided Information Extraction from Print-Oriented Documents*. PhD thesis, Citeseer, 2010. 4

[9] W. S. Lovegrove, David, and F. Brailsford. Document analysis of pdf files: methods, results and implications. In *Electronic Publishing*, pages 207–220, 1995. 2

[10] A. Shahab, F. Shafait, T. Kieninger, and A. Dengel. An open approach towards the benchmarking of table structure recognition systems. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, DAS '10, pages 113–120, New York, NY, USA, 2010. ACM. 6

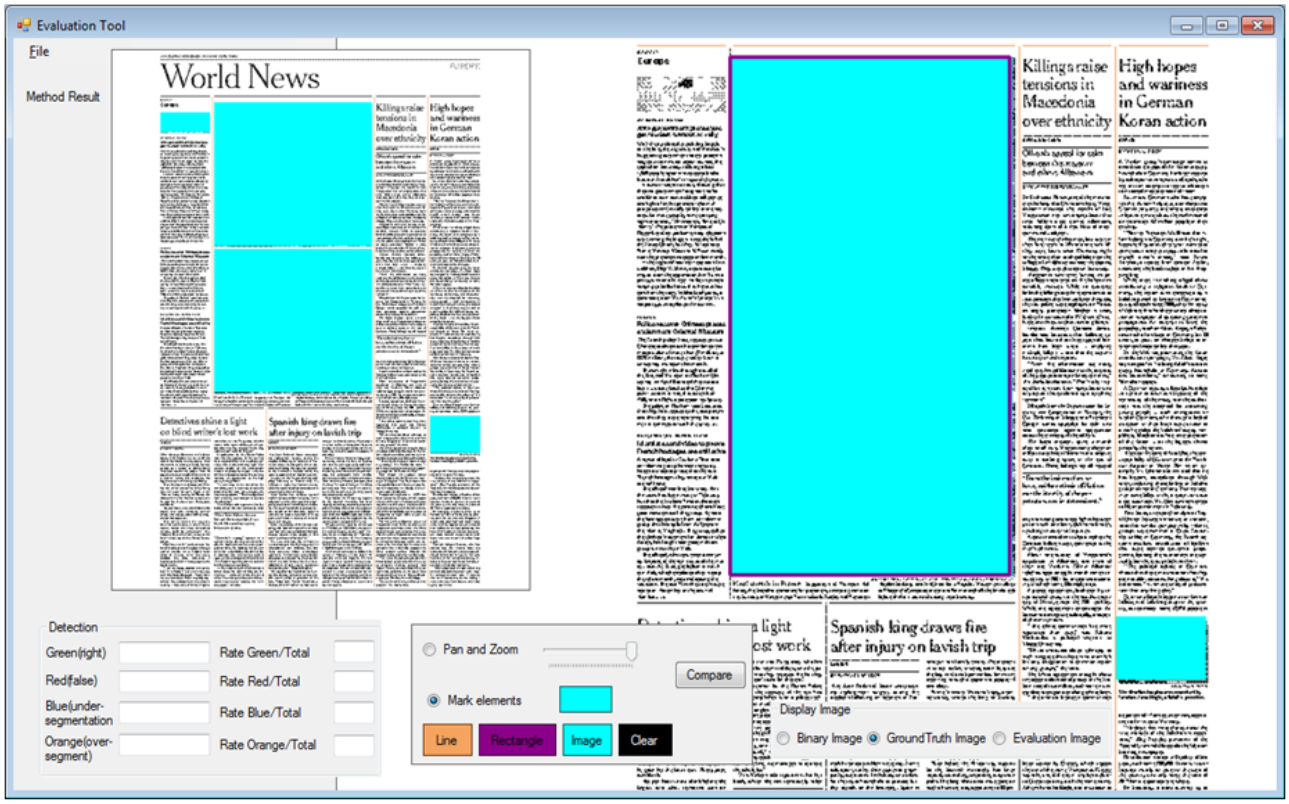


Figure 9. Interface of the evaluation tool

	Total	Retrieved	Correct detection	Incorrect detections	Partial detections	Over segment.	Under segment.	False positives	Missed objects
Structural lines	1235	1272	1062(86%)	126(10%)	0	0	36(3%)	126	51
Structural rectangles	637	655	452(71%)	226(35%)	1(<1%)	11(<2%)	4(<1%)	78	50
Graphic regions	559	977	470(84%)	66(12%)	9(<2%)	216(38%)	36(6%)	231	12

Table 1. Evaluation results on newspapers

	Total	Retrieved	Correct detection	Incorrect detections	Partial detections	Over segment.	Under segment.	False positives	Missed objects
Structural lines	96	106	89(92%)	18(18%)	0	0	0	1	0
Structural rectangles	21	47	19(90%)	7(33%)	1(4%)	1(4%)	1(4%)	11	2
Graphic regions	130	153	121(93%)	6(4%)	2(<2%)	25(19%)	3(2%)	1	1

Table 2. Evaluation results on technical manuals

ViCoS Eye - a webservice for visual object categorization

Domen Tabernik[1], Luka Čehovin[1], Matej Kristan[1], Marko Boben[1] and Aleš Leonardis[1,2]
[1]Faculty of Computer and Information Science, University of Ljubljana, Slovenia
[2]CN-CR Centre, School of Computer Science, University of Birmingham
{domen.tabernik,luka.cehovin,matej.kristan,marko.boben,ales.leonardis}@fri.uni-lj.si

Abstract. *In our paper we present an architecture for a system capable of providing back-end support for web-service by running a variety of computer vision algorithms distributed across a cluster of machines. We divide the architecture into learning, real-time processing and a request handling for web-service. We implement learning in MapReduce domain with Hadoop jobs, while we implement real-time processing as a Storm application. An additional website and Android application front-end are implemented as part of web-service to provide user interface. We evaluate the system on our own cluster and show that the system running on a cluster of our size can learn Caltech-101 dataset in 40 minutes while real-time processing can achieve response time of 2 seconds, which is adequate for multitude of online applications.*

1. Introduction

Increased processing power behind server based computers has in the recent years enabled many online services to offload their processing into a cloud-based computing. This has also become beneficial for computer vision problems where many computationally expensive algorithms are already enabling services such as TinEye[1], Macroglossa[2], Google Image Search[3] or Google Goggles[4]. These online services work particularly well for images that the system has previously seen on the internet but do not perform very well on previously unseen images. For instance, the TinEye service does not perform any object recognition[5], while querying Google Image Search with an camera snapshot of a simple coffee mug or a chair produces results where there is no mug or chair in any of the first 50 hits (see, Fig. 2). The only visual similarity between the query image and the results is a similar color distribution.

Adding more advanced computer vision algorithms such as object categorization would be highly beneficial as it could provide more information on objects that have not been previously seen but belong to a known category of objects. This would in turn open up the service for multitude of applications. For instance, the service could provide a cloud-based machine vision for robotic systems, or it could be used as a back-end for an application that is capable of poisonous plant or dangerous animal identification.

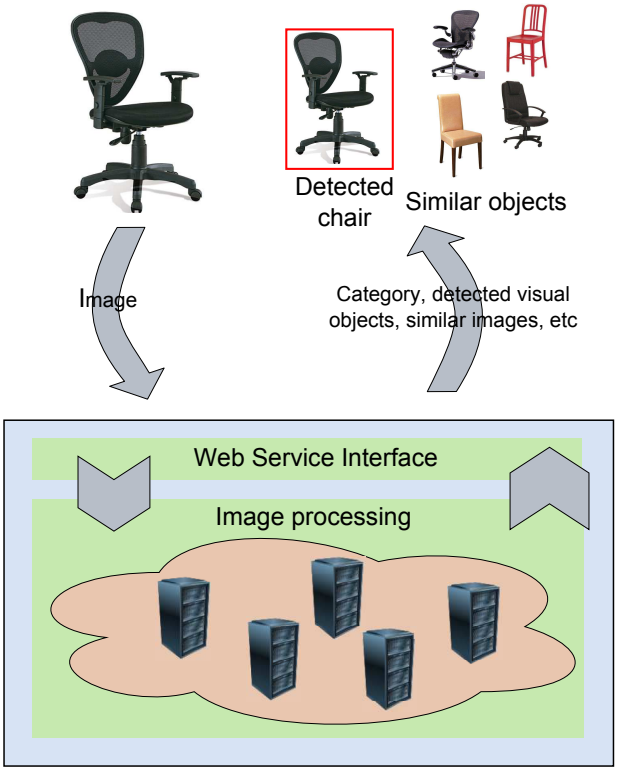


Figure 1. Conceptual overview of online computer vision service. Service takes a single image as a query, relays it for processing on cluster of machines and returns a result in a form of recognized category, detected objects, similar images, etc.

Most of the existing online computer vision services provide only simple image similarity searches. We show in this paper how advanced computer vision algorithms can be implemented as a cloud-based application to provide an online service. Simple offloading of computer

[1]http://www.tineye.com
[2]http://www.macroglossa.com
[3]http://images.google.com/
[4]http://www.google.com/mobile/goggles
[5]http://www.tineye.com/faq#similar

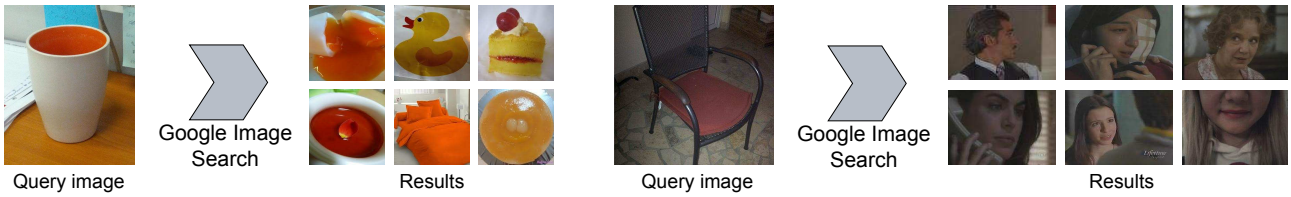


Figure 2. Two examples of Google Image Search on unknown images. Returned similar images match only based on color distribution but ignore any other information such as visual category.

vision processing to a single server has already been done [5, 7], but they implemented the algorithm on a single server which does not scale well and cannot be directly distributed across many servers. In our paper we therefore focus primarily on describing a system that can efficiently run computer vision algorithms on a cluster of machines. We analyze requirements for such a system for providing online service, describe our implementation and provide some performance analysis. The architecture of the system we are describing is general enough to allow for implementation of different kinds of computer vision algorithms, thus providing variety of services such as: detection of objects in images, recognition and identification of different visual categories or content based image search, not just based on color distribution but based on objects and visual categories identified in the query image. In our paper we demonstrate this service on a problem of object categorization using HoC descriptor [8].

This paper is structured as follows. In Section 2 we state the requirements that a computer vision algorithm must meet, the architecture is presented in Section 3 and the implemented computer vision algorithm is presented in Section 4. In Sections 5 and 6 we analyze in more detail how different back-end aspects of the service can be implemented and provide a general implementation details on user interface in Section 7. In Section 8 we provide some performance analysis of implemented service and conclude the paper in Section 9.

2. Requirements

We first present the following requirements for our system:

- providing an online service (a web-service) for computer vision algorithms,
- capability of distributed processing in a cluster of machines (a cloud),
- ability to handle hundreds of requests per second.

Running a system on a cluster of machines requires that all the implemented algorithms run distributed in order to utilize the resources as efficiently as possible. By distributing the processing across a cluster we also enable efficient scalability. Adding new machines to the cluster should be a straightforward process and should automatically provide higher throughput to handle more and more requests easily as the service expands.

Additionally, the system has to function as a service. This requires the system to handle requests that come either directly from the internet or from any underlying system using this service. Each request comes in a form of an image and the system has to be able to process it with computer vision algorithm and return a result in the form of a category or objects detected in the image or similar images. A general overview is depicted in Fig. 1.

As a service, the system also has to enable handling hundreds of requests simultaneously and process them sufficiently fast. This puts additional constraints on the algorithms that we can use. In general, the recommended tolerable waiting time (TWT) for web pages is approximately 2 seconds [6], but that number can vary for different applications. In our case, any algorithm capable of processing the image in up to five seconds or less should be acceptable for a multitude of online applications, while any further processing delay might deter the user from using this service.

3. Architecture

Many computer vision algorithms are divided into two stages: (i) learning and (ii) testing/classifying. We therefore implement two different subsystems. The first subsystem corresponds to the learning stage. As we also have to meet the required resource efficiency utilization by distributing the processing across different machines in the cluster setup, we term this subsystem as ***distributed learning***. The second subsystem provides a testing/classifying stage. This subsystem is also the main part of the system that has to be connected to the input of our web-service and in real-time provide results on each query requested by the user. As such we call this subsystem ***real-time stream processing***. We also implement a third subsystem called ***Web Service Interface*** that provides service API over the internet.

All subsystems have completely different assignments within the framework and are therefore implemented using different techniques. ***Distributed learning*** has to process relatively high number of training images (from thousands of images and upwards) therefore this process can be best distributed across a cluster by transforming it into a *MapReduce* [3] problem. In this domain the problem is represented by a set of input items which are processed with a *Map* and a *Reduce* function. By transforming the algorithm to a *MapReduce* domain a problem now becomes relatively straightforward to be distributed across

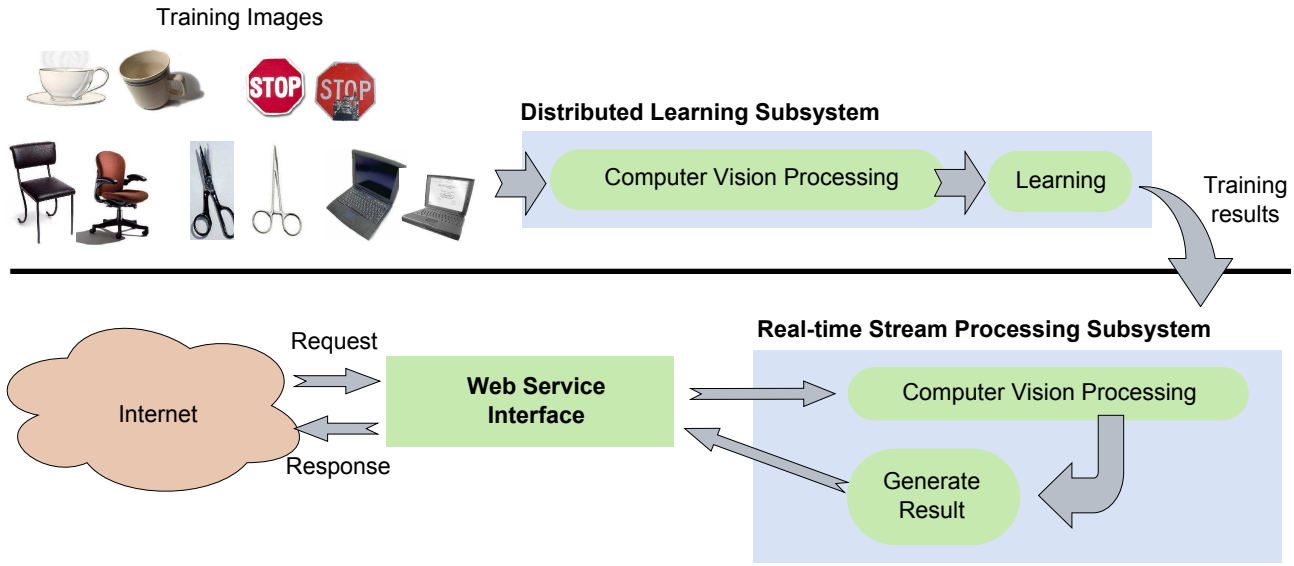


Figure 3. Architecture of the system providing a computer vision service. Service is divided into learning subsystem handling any learning and training of library, models, etc, and into real-time processing subsystem handling incoming request and generating results.

cluster machines, provided that each item can be processed with *Map* and *Reduce* function independently of other items. In our case we treat each training image as a single item. Any part of a computer vision algorithm that can process images independently can be trivially transformed into this domain. More complex parts would have to be re-factored in order to efficiently run on a cluster of machines.

Real-time stream processing must be implemented using different technique as its job and requirements are completely different. This subsystem is directly connected to the web-service and must serve hundreds of requests and return corresponding results in near real-time. For each query this system has to process a single image. Transforming this subsystem into MapReduce domain would not be possible since that system works as a batch processing of jobs, where all input data is known in advance. But in this subsystem we have to continuously handle requests that are coming into the system and therefore we do not know in advance how many request we will have. The best way to efficiently run computer vision algorithm on a cluster of machines would be to assign each request to a single machine for processing.

Web Service Interface subsystem provides API for access to the service and additional user interface in a form of a web site and an Android application. The subsystem also has to enable communication with *Real-time stream processing* subsystem. *Web Service Interface* has to work as a relay between the user and *Real-time stream processing* by forwarding the request from user for further processing and retrieving the results, which are then relayed back to the user. Graphical representation of all three subsystems can be seen in Fig. 3.

4. HoC-based object categorization

In general, the algorithm for image processing can be a multitude of different computer vision algorithms but in this paper we demonstrate the system on the problem of multiclass object categorization. For object categorization we use the LHOP [4] based HoC [8] descriptor with an SVM [2] classifier. The advantage of using this algorithm is LHOP's efficient inference. As the service has to enable quick response to user's query, this method with its efficient inference allows us to quickly process the image and generate the HoC descriptor. The descriptor based approach is used on the whole image. Therefore our system provides only object categorization service and not object localization or detection. In this section we briefly describe the process of categorization using HoC descriptor and refer the reader to [8] for further details.

Histogram of Compositions is a shape-based descriptor which uses learning of shapes to find only shapes and structures that are most relevant for object description. Relevant shape fragments identification is achieved by the learning process of learnt-hierarchy-of-parts [4] (LHOP) model, which produces a vocabulary of hierarchical compositions. This compositions are then further used in the process of constructing the HoC descriptor.

Constructing a HoC descriptor $\mathcal{H}$ from an image is a two step process. In the first step, image is processed with a previously learnt LHOP library $\mathcal{L}$ to produce a list of compositions $\{\pi_k\}_{k=1:K}$. Since LHOP represents shapes using hierarchical compositions we produce a list of compositions for each layer of hierarchy. Descriptor $\mathcal{H}$ is then constructed in second step from compositions of the desired layer(s) $\{\pi_k \mid \pi_k \in \text{selected layers}\}$. This process applies partitioning scheme of M-regions and creates small histograms of compositions $\mathcal{H}_m$ of each region that form a part of a final descriptor $\mathcal{H} = \alpha[\mathcal{H}_1, ...\mathcal{H}_M]$. This

descriptor is then further used in an SVM to perform the final step in the object categorization.

5. Distributed learning subsystem

In this section we detail implementation of distributed learning for object categorization using HoC descriptor. We define the process of learning as follows:

Input: $\mathcal{L}$ is a pre-learnt LHOP library and $\{(\mathcal{I}_i, c_i)\}_{i=1:I}$ is a set of training images $\mathcal{I}_i$ each annotated with a proper visual category name $c_i \in \mathcal{C}$.

Output: $\{m_j\}_{j=1:J}$ is a set of SVM models representing trained model m_j for each different category found in the set of category names $\mathcal{C}$.

Algorithm steps:

1. Process each image $\mathcal{I}_i$ from the set of training images $(\mathcal{I}_i, c_i)$ with an LHOP model to produce compositions π_k for the whole hierarchy:

$$\mathcal{P}(\mathcal{I}_i, \mathcal{L}) = \{\pi_k\}_{k=1:K_i}.$$

2. Generate HoC descriptor $\mathcal{H}_i$ for each image $\mathcal{I}_i$ processed with an LHOP model that produced the list of compositions $\{\pi_k\}_{k=1:K_i}$.

3. Group HoC descriptors $\mathcal{H}_i$ based on its category label c_i and train SVM model m_j for each grouped visual category c_j:

$$svm(\{\mathcal{H}_i\}_{i=1:I}, c_j) = m_j.$$

We use our implementation of LHOP [4] and HoC [8], and LIBSVM [2] for support vector machine.

Before translating our problem into MapReduce domain we first provide basic MapReduce notation and refer the reader to [3] for more detail. Input for a MapReduce problem is always an array of key-value pairs $\{(\kappa_l^{input}, \omega_l^{input})\}_{l=1:L}$ (shown as (a) in Fig. 4) that gets processed with two different functions. First with a map function $\mathcal{M}$ to produce (b) intermediate result of a key-value pair :

$$\mathcal{M} : (\kappa_l^{input}, \omega_l^{input}) \mapsto (\kappa_l^{inter}, \omega_l^{inter}).$$

All intermediate pairs are then grouped by their keys into (c) N different groups $\{(\kappa_n^{inter}, \{\omega_{o,n}^{inter}\}_{o=1:O_n})\}_{n=1:N}$. Each group with the same key is then further processed by a reduce function $\mathcal{R}$ that returns (d) final key-value pairs for each group with the same key:

$$\mathcal{R} : (\kappa_n^{inter}, \{\omega_{o,n}^{inter}\}_{o=1:O_n})) \mapsto (\kappa_n^{output}, \omega_n^{output}).$$

A simple graph of this process is depicted in Fig. 4. We can now transform each step of our object categorization procedure into a separate MapReduce problem and connect them together by chaining output of each step into the input of the next step.

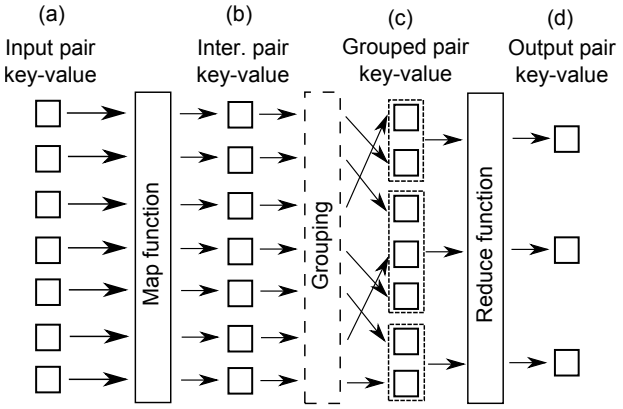


Figure 4. MapReduce processing pattern. Set of input pairs (a) gets processed in parallel by Map function then this intermediate result (b) gets grouped by its keys and each group (c) gets finally processed in parallel by Reduce function to produce final output (d).

5.1. Object categorization learning as MapReduce

In the first step we have input in a form of list of training images with category labels, which we now represent as MapReduce input pair with empty key and value as image with category:

$$\kappa_i^{input\_LHOP} = \emptyset,$$
$$\omega_i^{input\_LHOP} = (\mathcal{I}_i, c_i).$$

Processing with LHOP library can then be easily represented as a MapReduce map function $\mathcal{M}^{LHOP}((\kappa_i^{input}, \omega_i^{input})) = \mathcal{P}(\mathcal{I}_i, \mathcal{L})$, while reduce function $\mathcal{R}^{LHOP}$ is not needed in this case. Intermediate result of map function is directly result of MapReduce process for this step which in this case is represented as empty key and list of compositions with category name as value:

$$\kappa_i^{output\_LHOP} = \emptyset,$$
$$\omega_i^{output\_LHOP} = (\{\pi_k\}_{k=1:K_i}, c_i).$$

The second step is transformed in a similar manner. We first chain output of previous MapReduce step directly into input pair of current step:

$$\kappa_i^{input\_HoC} = \emptyset,$$
$$\omega_i^{input\_HoC} = \omega_i^{output\_LHOP} = (\{\pi_k\}_{k=1:K_i}, c_i).$$

Map function $\mathcal{M}^{HoC}$ in this case generates HoC descriptor, while reduce function $\mathcal{R}^{HoC}$ is not needed, so intermediate output of map directly becomes output of the current MapReduce step, in which we return an empty key and HoC descriptor with category label as value:

$$\kappa_i^{output\_HoC} = \emptyset,$$
$$\omega_i^{output\_HoC} = (\mathcal{H}_i, c_i).$$

For the last step, which is SVM training, we need to distribute processing not by images, but by all the possible categories as training of a single category will require

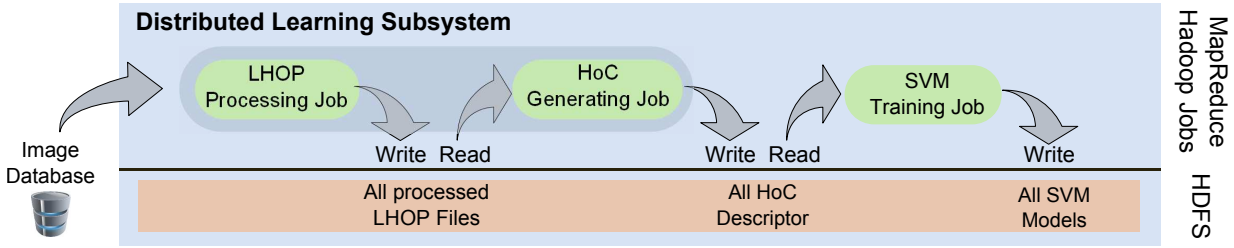


Figure 5. The process of learning visual categories using HoC descriptor and support vector machine. In general process is composed out of three different Hadoop jobs: LHOP processing, HoC generating and SVM training. We merge first two steps to avoid costly read/write to HDFS.

descriptors from all training images. However we can still train a single category independently of other categories. Additionally, we leverage the number of cluster machines to implement a simple grid optimization of SVM parameters. We achieve this by adding additional input pairs for each parameter configuration we check. We can now write the last step as MapReduce problem with input pair as category name for key, and HoC descriptor of all training images together with specific SVM parameter values for the input value

$$\kappa_j^{output\_SVM} = c_j,$$
$$\omega_j^{output\_SVM} = (\{\mathcal{H}_i\}_{i=1:I}, svm\_params).$$

Map function is then implemented as an SVM training $\mathcal{M}^{SVM} = \text{svm}(\{\mathcal{H}_i\}_{i=1:I}, c_j, svm\_params)$ which returns category name for key, and a trained SVM model with some performance metric p for value:

$$\kappa_j^{inter\_SVM} = c_j,$$
$$\omega_j^{inter\_SVM} = (m_j, p).$$

The reduce is then implemented as a *max* function across all the trained models for specific category:

$$\mathcal{R}^{SVM}(c_j, \{(m_o, p_o)\}_{o=1:O_j}) = (c_j, \underset{p_o}{argmax}(\{(m_o, p_o)\})).$$

The result of reduce function is then a set of pairs with category names and appropriate best-performing SVM model (c_j, m_j).

5.2. Hadoop as MapReduce implementation

There are many available implementations of MapReduce. We have chosen Apache Hadoop [9] which is an open source implementation written in Java. It is intended for processing of BigData (tens of Terabytes of data) as a batch of jobs and is easily scalable up to 1000 or more nodes (machines). Hadoop also comes with its own version of distributed file system called Hadoop Distributed File System (HDFS), which holds all the input and output data as well as any intermediate results.

Distributed learning subsystem can now be implemented as batch of three different Hadoop *job jars* that run in a sequence. In the first job we read images from the database, process them with LHOP and save results onto HDFS file-system. The second job waits for all the images to be processed, then reads them from HDFS, generates appropriate HoC descriptor and saves them back to HDFS. The final job waits until all the HoC descriptors are generated, reads them from HDFS and generates appropriate number of SVM problems for each category which are then trained on the cluster. The final job creates SVM models and sends them to *real-time stream processing subsystem*. This sequence of jobs is depicted in Fig. 5.

We noticed that, between each job, we write results in HDFS file-system and then read them again for the next job. This presents a possible bottleneck as access to HDFS can be expensive. Therefore we additionally optimize the sequence of jobs by merging the first and the second job together. In this way we generate HoC descriptor immediately from the image processed with an LHOP model that is still loaded in memory and can therefore avoid writing and reading results of the first job. Since HoC descriptor can now be generated immediately after the image is processed, we also eliminate any delay between the first and the second job and thus optimize the process even further.

6. Real-time stream processing subsystem

In this section we present a subsystem capable of handling requests from the web-service, process them with object categorization algorithm and provide appropriate response back to the requester. We define the algorithm required for this subsystem as follows:

Input: $\mathcal{L}$ is a pre-learnt LHOP library, $\{(c_j, m_j)\}_{j=1:J}$ is a set of trained SVM models and $(r_i, \mathcal{I}_i)$ is a request with an image $\mathcal{I}_i$ and request information r_i.

Output: (r_i, c_i, p_i) is a response to request r_i with an image classification category c_i and SVM score p_i.

Algorithm steps:

1. Process request image $\mathcal{I}_i$ with an LHOP model to produce compositions π_k for the whole hierarchy:
$$\mathcal{P}(\mathcal{I}_i, \mathcal{L}) = \{\pi_k\}_{k=1:K_i}.$$
2. Generate HoC descriptor $\mathcal{H}_i$ from image $\mathcal{I}_i$ processed with an LHOP model that produced list of compositions $\{\pi_k\}_{k=1:K_i}$.

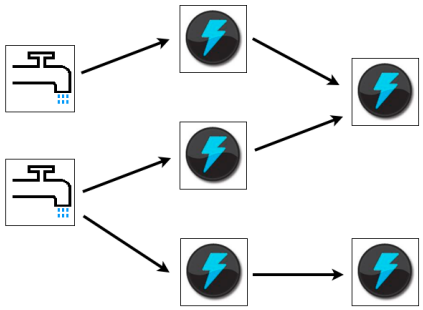


Figure 6. Example of Storm topology with two spouts connected to five bolts.

3. Classify HoC descriptor $\mathcal{H}_i$ into a number of pre-trained categories using SVM. We select category with the best score p_i:

$$c_i = \underset{(c_j, m_j)}{argmax}\{p_j = svm(\mathcal{H}_i, m_j);\ p_j > 0\},$$

if any score is higher then 0 or return empty category otherwise. We return original request r_i and category classification c_i with its score p_i as result.

Implementing this kind of a system would not be possible with the Hadoop in MapReduce domain since in that system we have to know in advance the number of input items to process in order to split them in the most efficient way for processing across different cluster machines. While in our case input data would be unpredictably coming into the system based on user requests. Therefore the system does not know in advance how many requests we will have. The best way to implement this kind of request handling would be with a system of workers and queues distributed across different machines where each worker could then accept the workload as it arrives into the system. To implement this design we use Storm[6] which is an open source distributed real-time computation system. It provides all the necessary infrastructure of workers and queues distributed across cluster machines and provides convenient way for implementing applications on top of this system.

Writing application on top of Storm requires to define a topology. Topology is a digraph where nodes are processing elements and are implemented either as *spout* or *bolt*, and directed edges represent the direction of processed data. A general example of Storm topology is shown in Fig. 6. The work in this topology starts with a *spout*, which can be hooked to the outside world and would be generating (emitting) new data streams. A stream of data would then be sent to the appropriate neighboring *bolts*. Each *bolt* then processes the data and can emit its results as one or more new streams to its neighboring nodes. The process continues until no more new data is emitted. Processing in each node is handled by Storm and gets distributed across the cluster according to availability of workers therefore allowing for best utilization of resources.

[6]http://storm-project.net

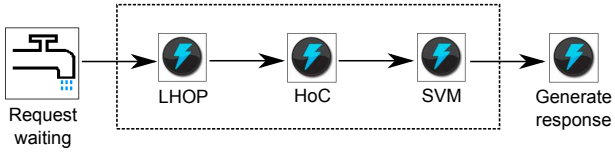


Figure 7. Storm topology for object categorization processing using HoC descriptor and support vector machine.

6.1. Real-time image categorization service as Storm topology

We define our system of image categorization service as Storm application by providing appropriate topology of spouts and bolts. In our case we define one bolt for each step of the algorithm and simply connect them in a sequence. Therefore a single bolt implementation represents LHOP processing, one bolt represents generating HoC descriptor and additional bolt represents classifying with the trained SVM models. We also need to define one spout which will wait for input requests and send appropriate stream of data to the first bolt. Also additional bolt has to be added at the end of the stream that will send response in a form of a category name back to the original caller. A graphical depiction of this topology can be seen in Fig. 7.

The above topology allows for handling of each request by multiple workers across different machines as each bolt can be processed simultaneously. But due to sequential processing we still have response time that is the sum of each individual bolt processing time:

$$t_{response} = t_{LHOP} + t_{HoC} + J \cdot t_{SVM} + \bar{t},$$

where J is the number of categories and $\bar{t}$ is the additional time needed for communication between different bolts and also between user and web-service.

Note, that we have implemented SVM classification as a single bolt worker. This may not appear optimal at first since we need to test descriptor against multiple categories which can be done in parallel. Thus implementing SVM classification of each individual category as single bolt might provide better results. In practice however, the time needed for classification of all categories (100 categories in our case) in a single bolt was only 25% of the response time with the other 75% representing the time required for LHOP processing. Performance benefits of using multiple bolts might become more noticeable with higher number of categories but even in that case it would still be more reasonable to handle a subset of categories at once to avoid any additional delay of sending hundreds of streams across the cluster.

7. Web service interface

As our system is intended to work as a web-accessible service, we have implemented a simple web interface that

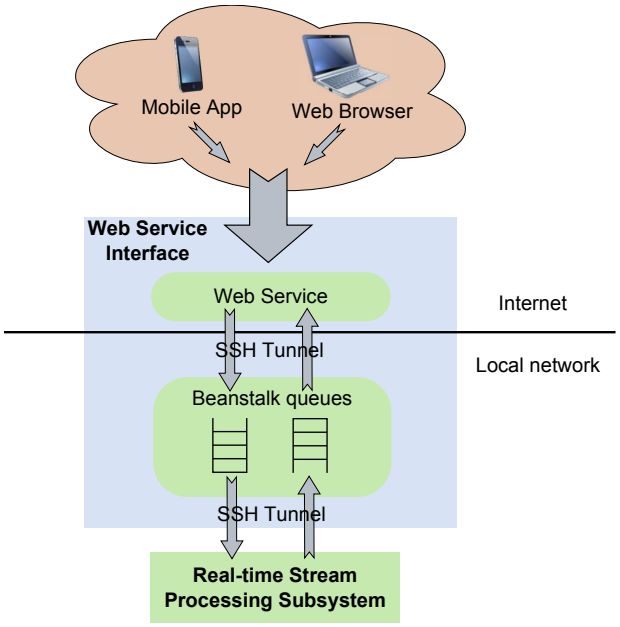


Figure 8. Web Service Interface architecture.

handles user-interaction, relays the requests to the back-end and displays the results to the user after the request has been processed by the service.

7.1. Web site

The main component of our interface is a web-site written in Python and JavaScript. The Python part accepts requests, validates input images (size and type) on the server, equips them with additional meta-data and pushes jobs to a Beanstalk[7] incoming queue. Once the job is processed, the results are inserted to a database by a daemon process where they are available to the service. The JavaScript part of the interface handles smooth transitions of the browser interface.

As security of the system is important the computational back-end is located on the local network and is connected to the public web server that runs the interface front-end using a persistent encrypted tunnel. This relation is illustrated in Fig. 8.

7.2. Android front-end

In our internal user-experience study we have found that a classical web-interface could be improved by allowing user to quickly capture and submit new images as queries. As nearly all mobile smart-phones and tablets now days contain a high-resolution camera, we have created a prototype Android application to streamline the process. The application enables users to capture images, uploads them to the web-service and displays the result.

8. Performance

Evaluation of our system was performed by implementing both subsystems on a cluster of three machines

[7] http://kr.github.com/beanstalkd/

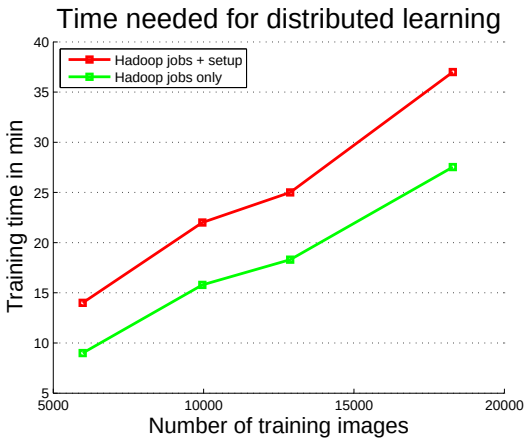


Figure 9. Time in minutes needed for distributed learning relative to the number of training images. The red line represents all time needed for training including Hadoop jobs with any additional time for setup preparation, while the green line represents only time needed for Hadoop jobs.

t_{LHOP}	t_{HoC}	$J \cdot t_{SVM}$	$\bar{t}$	$t_{response}$
1547 ms	85 ms	462 ms	48 ms	***2144 ms***

Table 1. Response time of object categorization service using Storm processing.

each with more than 30 CPU cores and more then 80 GB of memory. For Hadoop system we assigned 35 slots on each machine which allows simultaneous execution of 105 Map or Reduce functions. The distributed learning was preformed on all 9124 images of Caltech-101 dataset with 103 categories (with background images and separate faces and faces_easy category). We also added mirror image for each training example. This produced 18248 different training examples. We evaluated the performance of distributed learning subsystem by varying the number of training examples and timing each Hadoop job with any additional time for framework setup. The results are shown in Fig. 9. We were able to learn all 103 categories with 18248 images in less then 40 minutes but there is also some room for improvements as Hadoop jobs took less then 30 minutes. Based on the observed numbers we can see that we can easily learn Caltech-101 in less then an hour and since learning could be done only occasionally (depending on newly gathered image dataset from crawled web pages or user feedback once or twice a week could be enough) we could easily scale our system to hundred of thousands or even millions of training images as this could be processed in a day or two.

Evaluation of real-time stream processing subsystem was performed on a single machine with 8 CPU cores where we assigned 4 slots for Storm task processing. We evaluate only response time of a single query and provide theoretical extrapolation for max server load. Results of the evaluation can be seen in Table 1. We tested the server with 20 different images and provide an average time for each component. Assuming we can use the same cluster as for Hadoop (105 nodes) we can calculate the maximum

number of incoming requests that we can handle. Using queueing theory [1] we can define our Storm application service as M/M/105 queue model, where arrival rate λ is according to a Poisson process and service times have an exponential distribution with parameter μ. Based on observed averaged processing time $t_{response} = 2.144\ sec$ we can set the service time $\mu = 0.4664\ sec^{-1}$ and calculate the maximum arrival rate before the system's queue grows to infinity (i.e. when $\rho > 1$):

$$\rho = \frac{\lambda}{m \cdot \mu,}$$

where $m = 105$ is number of processing servers. We calculate the max arrival rate $\lambda_{max} = m \cdot \mu = 48.9720$, which tells us that our service would be able to handle 48 requests per second before the queue would start to grow to infinity. The observed response time and a calculation of handling 48 requests per second in a cluster of 105 nodes could effectively allow us to use the system for multitude of online services. Scaling to any more traffic or requests would only require additional computing power which can be added trivially even without stopping the existing service thanks to flexibility of the Storm and Hadoop systems.

9. Conclusion

In this paper we presented implementation of a computer vision algorithm as an online web-service and analyzed the efficiency of implementation on a distributed processing platform. In particular we implemented a service for object categorization using HoC descriptor [8] and a support vector machine [2]. We divided the architecture of the system into two stages: distributed learning and real-time stream processing. The first part was responsible for the learning and was implemented in MapReduce [3] domain as a Hadoop job. While the second part, real-time stream processing, was responsible for handling online service requests and was implemented as a Storm application. An additional part of the system was also the web-service interface that acted as an relay between users and distributed processing platform. We implemented it as an web site and also as an Android application.

We analyzed the performance of each stage on a cluster with more then 100 CPU cores. The system was able to complete the learning phase in less than an hour for Caltech-101 dataset with around 18000 training images and more then 100 categories. Accounting for infrequent, runs and based on performance analysis, we have shown that such system should be able to scale to hundreds of thousands or even millions of training images. The analysis of the second part, the real-time stream processing, has shown that our service has response time of around 2 seconds which is completely adequate for multitude of online applications. We also showed that the system deployed on a cluster of 105 nodes should be able to handle 48 requests per second and we could easily scale to higher number by adding additional machines to the cluster.

Note, that the presented system can potentially be used not only as an online computer vision service but by adding user feedback to the web interface we could use the system as image annotation tool. For instance, a user might be able to provide a feedback for each missing object or improperly classified category. This data can then be collected by the system and used to generate new dataset of training images. This dataset could augment the existing training images and together with retraining enable better performance of the whole service. We would like to implement this extension in our future work together with some other small optimizations. Additionally, we would also like to extend the system from only object categorization to object detection and localization. In our future work we will use the system for implementing content-based image search that would take into account information about object categories found in the image.

Acknowledgments. This work was supported in part by ARRS research program P2-0214 and ARRS research projects J2-4284, J2-3607 and J2-2221.

References

[1] A. O. Allen. *Probability, statistics, and queueing theory with computer science applications*. Academic Press Professional, Inc., San Diego, CA, USA, 1990.

[2] C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[3] J. Dean, S. Ghemawat, and G. Inc. Mapreduce: simplified data processing on large clusters. In *In OSDI'04: Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation*. USENIX Association, 2004.

[4] S. Fidler and A. Leonardis. Towards scalable representations of object categories: Learning a hierarchy of parts. In *CVPR*. IEEE Computer Society, 2007.

[5] S. S. Kumar, M. Sun, and S. Savarese. Mobile object detection through client-server based vote transfer. In *CVPR*, 2012.

[6] F. F.-H. Nah. A study on tolerable waiting time: how long are web users willing to wait? *Behaviour & IT*, 23(3):153–163, 2004.

[7] D. Omerčević and A. Leonardis. Hyperlinking reality via camera phones. *Machine Vision and Applications*, 22(3):521–534, 2011.

[8] D. Tabernik, M. Kristan, M. Boben, and A. Leonardis. Learning statistically relevant edge structure improves low-level visual descriptors. In *International Conference on Pattern Recognition*, 2012.

[9] T. White. *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 1st edition, 2009.

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

3D Change Detection by Inverted Distance Fields

Jean-Séverin Morard, Manfred Klopschitz,
Stefan Kluckner, Claudia Windisch
Siemens Corporate Technology, Graz, Austria
first.last@siemens.com

Christof Hoppe, Horst Bischof
Institute for Computer Vision and Graphics
Graz University of Technology, Austria
last@icg.tugraz.at

Abstract.

3D reconstruction from sets of images has reached a high level of maturity, but there is still a lack of automated understanding in terms of semantics. In this work we propose a technique to detect changes in 3D reconstructions that are acquired over time. Given multiple sparse 3D reconstructions, that are aligned in a canonical coordinate system, a semi-automated procedure generates dense information on a region of interest and relates this information overtime by using a novel representation of geometric change based on inverted distance fields. For evaluation, we perform experiments on synthetic datasets as well as on real world images.

1. Introduction

The automated detection of change from visual data is a very active research topic due to enormous amount of available imagery on the Internet and cost-effective visual data acquisition by digital technologies. In particular, the redundancy in the available data enables the generation of scene geometry, which supports the detection of change directly in the third dimension [21]. Typically, the change detection based on geometry overcomes the typical problems of well-established image to image comparison like variations in illumination, seasonal changes and changes in view points.

Recent approaches for 3D reconstruction enable the accurate mapping of real scene observations into virtual world representations. Such approaches take into account large collections of still images [19, 12, 1] or even active sensors like Kinect [16] or Time-of-Flight [5]. Although 3D reconstruction has reached some sort of maturity, the understanding of the acquired 3D model is till lacking in terms of semantics. Popular scene understanding approaches [13, 8, 4] would benefit from time-dependent 3D scene observations due to additional scene knowledge like scene dynamics or foreground-background situation.

Most of recent reconstruction algorithms focus on creating 3D reconstructions of a scene without consideration of the timestamp of the image acquisition [19, 1]. However, the processing of time sorted data sets enables the generation of individual 3D models, which typically map the as-built situation at individual time steps. Relating and linking the models within a common spatial world enables the creation of 4D representation enabling a highly automated procedure of detecting change in the observed scene. Additionally, the detected change supports the quantification of increasing and decreasing volumetric changes which can also be used for improved material flow estimation, object counting or visualization.

In this paper, we thus address the task of a user driven change detection within a 4D representations. The main contribution is an efficient approach for 3D change detection that takes into account real-world scene information e.g. mapped by overlapping images. Based on tuples of reconstructed scenes, we propose a generic concept to consider time dependent 3D observations by inverted distance field representations in order to localize and quantify volume differences in the third dimension.

The remaining part of the paper is structured as follows: In Section 2 related work is summarized, in Sections 3 and 4 we describe our method in detail. Section 5 highlights the performed experiments on synthetic and real data. In Section 6 we conclude our work and discuss future work.

2. Related Work

Detecting geometric change in large scale scenes in high quality depends on creating surface models

and implicitly high resolution volumetric models of some kind for the complete scene. This is computationally very demanding and can be accelerated by computing change only for a region of interest. We present a method for geometric change detection that uses a user selected key frame to select the region of interest and present a novel volumetric change detection algorithm. The change is computed directly on the signed distance function implicit surface representations. No direct surface extraction for the individual data sets is necessary, as opposed to the method of [21]. In [22], a simple 2.5D change detection algorithm based on ortho view image differences is presented. Our method in contrast computes full 3D change surfaces.

Change detection algorithms that are based on appearance change have been a popular research topic in the past. For one static camera, appearance based background models like [20] have a long history in computer vision. Generalizations of appearance based learning methods to multiple views are presented in [17].

Our approach computes the geometric change of a scene by taking two implicit surface representations of the area of interest. The geometry of the individual data sets is represented by signed distance volumes that are obtained by multi-view stereo for the data sets at each time step. The change surface is computed as the iso-surface of the signed distance change volume. The signed change volume is obtained from a pair of signed distance volumes.

3. 4D Scene Mapping

In this Section, we describe some preliminary processing steps in order to visually map a scene of interest from images into a virtual world. Our work is focused on the detection of changes that happened over time, we thus outline how to generate a 3D model that also takes into account the aspect of time. In the following, we denote time-depending 3D models as 4D representation.

In order to be able to detect changes in 4D over time, 3D scene data sets have to be estimated from overlapping images initially. To obtain accurate 3D information, we utilize well established procedures and methods. In a first step, Structure from Motion (SfM) is applied to estimate sparse geometry and camera poses for the defined timestamps. Since the sets of images, collected at specified timestamps, are processed individually, the obtained results are typically not aligned within a common coordinate system. We thus apply an iterative scheme to estimate transformations that align the SfM results (3D points and camera poses) to a defined reference model within a common world. Finally, we are now ready to perform multi-view matching based on plane-sweeping to densify the SfM point clouds time-dependently.

An area-based multi-view matching algorithm produces dense range images for each image in the image sets. Initial range images are computed from overlapping images (a reference image and N best fitting neighbors) with the plane sweeping approach, as described in [3]. In order to determine the best fitting neighbors, we consider both the median triangulation angle and image overlap according to the convex hulls computed over determined SIFT [15] key locations from SfM. We use plane sweeping because it is a robust approach to compute a depth image from multiple views. Image rectification based depth estimation in contrast is not possible for all camera motions.

For the key-view, a corresponding 3D space (the relevant sweep space is determined from visible 3D points) is traversed by fronto-parallel planes at different depth steps. For a plane at depth d, a homography H can be estimated from the relative pose between the reference view and the sensor view, which maps the sensor view onto the plane. Taking into account the reference and projected sensor views, a correlation measurement can be computed and aggregated for every pixel and its local neighborhood (typically 3, 5 or 7 pixels). In case of a potentially true depth hypothesis, the correlation would be high. In our case, we use truncated zero-mean normalized cross correlation (ZNCC) [9] as similarity measurement in order to implicitly handle occlusions and to be invariant to illumination changes. To obtain the final depth estimates for each pixel in the range image, we utilize a rapid winner-takes-all (WTA)[18] strategy, where we choose the corresponding depth value with highest ZNCC value obtained for the multi-view matching procedure.

4. Change Detection

This part presents how we compare two data sets and highlight their differences. We call a data set an acquisition of the scene at a given timestamp. It is composed of all the photos, all the camera positions and the sparse geometry of the scene (i.e. the

point cloud) related to its acquisition. In order to detect if an object appeared or disappeared, we define which data set is the reference (DS_R), and which one is compared (DS_C).

4.1. View point selection

Our algorithm takes depth information from the range images and fuses them into a voxel space. Computing high resolution voxel spaces for complete scenes is very time and memory demanding. Many parts of a scene are usually not of interest for the user. To increase the voxel space resolution in the area of interest and reduce computational time, we use a region of interest for change detection. The region of interest operator in this work is simply a user selected reference view chosen in DS_R. This camera is called C_R^0. The difference is then computed for a volume in this viewing frustum, so that every change which is not in the field of C_R^0 is ignored.

The next step is to find the closest camera in DS_C which see the scene under the same point of view as C_R^0. As in the plane sweep algorithm [3], this is done by considering both the median triangulation angle and the image overlap. The best fitting neighbour camera C_C^0 is set as the reference camera in DS_C.

4.2. Voxel space creation

In order to construct the voxel space V, the first thing to do is to determine the bounding box B of the region of interest. This region is composed of the 3D points visible both in C_R^0 and C_C^0. Consequently, B is the intersection of the reference bounding box B_R and the compared bounding box B_C.

This step is carried out for building both B_R and B_C and their intersection gives B. The voxel space is then built inside B with cubic and identical voxels. Its resolution is adjusted to the resolution of the range images : to avoid aliasing after filling the voxel space, its resolution is chosen bigger than the plane sweep resolution.

Finally, each data set gets its own voxel space (V_R and V_C) which has the same dimensions as V. We note $N_{\{x,y,z\}}$ the number of voxels on each dimension, $N = N_x \times N_y \times N_z$ the total number of voxels and $v(j)$ a voxel, where $j \in \{0..N\}$.

4.3. Voxel space filling

This part presents how to fill V_R with the reference data set, assuming that the procedure is exactly the same for V_C and DS_C. The reference voxel space is filled with the range image R_R^0 computed from C_R^0 with the plane sweep algorithm. In order to improve the quality of the final result, it is possible to use more range images computed from complementary views. The two following sections give details of how these complementary views are chosen and how their information are fused into V_R.

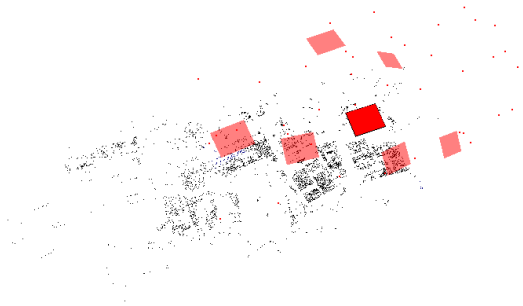


Figure 1: The opaque camera is the reference camera. The other ones are chosen to look at the scene from a complementary point of view.

4.3.1 Complementary views

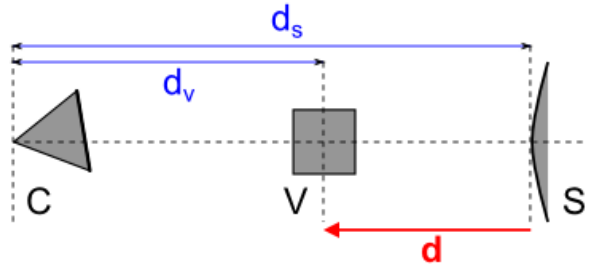
Using only the reference camera stereo result to construct the 3D model of the scene is generally insufficient because of occlusions. Similarly, as a depth sensor has to be moved around an object to get a well defined surface, like for example in [16], we select complementary stereo views C_R^i ($i \in \{0..n\}$) from the sparse reconstruction images to remove occlusions. Depth images R_R^i are computed again from C_R^i with the plane sweep algorithm.

The selection criterion is that each new camera must look at the scene from a new different point of view and also cover a large number of 3D points visible from C_R^0. Figure 1 illustrates how complementary cameras are chosen.

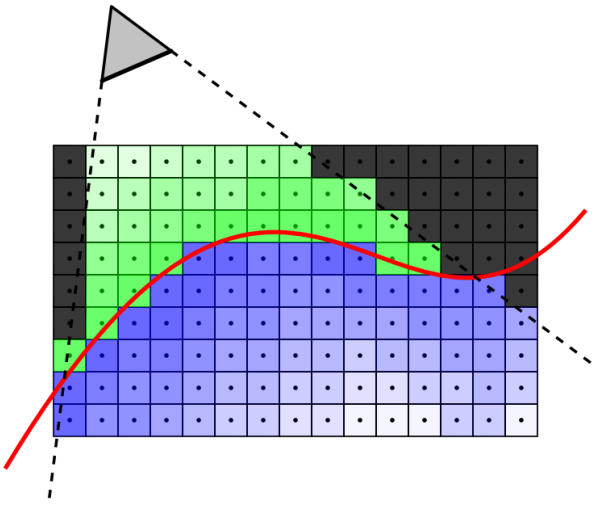
4.3.2 Data fusion

For range image integration we use and adapt a method based on the signed distance field representation introduced in [6]. First, we define M_R as the 3D real model of the scene for DS_R. The goal here is to compute this model by determining which voxels the surface of M_R goes through, given that the only informations we have about surfaces are the range images R_R^i. In this way, two values are assigned to each couple $(v(j), R_R^i)$, with $i \in \{0..n\}$ and $j \in \{0..N\}$. The first value is $d^i(j)$. This is the signed distance

value between the nearest surface of M_R along the sensor's line of sight from C_R^i and the center of $v(j)$. The distance from the camera to the nearest surface is contained in R_R^i : it is the value of the pixel where the voxel's center is projected. The second value is $w^i(j)$. This is the unsigned weight that we take equal to 1 if the voxel's projection is inside R_R^i and 0 otherwise. Curless et al. [6] give another values to $w^i(j)$, depending on the type of camera used to capture the image (quality, resolution, technology...). In our case, every photo is taken with the same camera, so we use a constant value. Figure 2 illustrates the construction of the signed function d^i for one camera in 2D.



(a)



(b)

Figure 2: Distance function $d^i(j)$. (a) illustrates how to fill one voxel. d_s is the distance between the camera and the surface (that we want to locate in the voxelspace) and is readable in the range image. d_v is the distance between the camera and the voxel center. (b) shows a voxel space in $2D$. The red line is the position of the surface we are looking for. Voxels in grey have an undefined value (not a number). Voxels in green and blue are respectively positive and negative. The darker they are, the smaller the absolute value of $d^i(j)$ is.

In the original algorithm, the weighted combination of these two functions gives $D(j)$. This is this value that is stored in $v(j)$:

$$D(j) = \frac{\sum_{i=0}^{n} w^i(j) d^i(j)}{\sum_{i=0}^{n} w^i(j)} \quad (1)$$

It is also possible to compute $D(j)$ iteratively, by using the cumulative formula:

$$D^{i+1}(j) = \frac{W^i(j) D^i(j) + w^{i+1}(j) d^{i+1}(j)}{W^i(j) + w^{i+1}(j)} \quad (2)$$

$$W^{i+1}(j) = W^i(j) + w^{i+1}(j) \quad (3)$$

$D^i(j)$ and $W^i(j)$ are respectively the cumulative signed distance and the unsigned weight functions after integrating R_R^i. Finally, $D(j)$ is normalized ; it is divided by the size of a voxel, so that distances are expressed in voxels. If $\sum_{i=0}^{n} w^i(j) = 0$, then $D(j)$ is assigned to not a number.

Once V_R is completely filled, it is admitted that the voxels with value zero are traversed by M_R. As for the other voxels, the sign of $D(j)$ indicates if $v(j)$ is in front of or behind the surface (from the size of the surface where C_R^0 is localised), and the larger $|D_i(j)|$ is, the larger is the distance of the surface. The resulting estimated surface of the model is the isosurface at the zero-crossing of $D(j)$. This zero-isosurface is extracted with the marching cubes algorithm described in [14].

4.4. Voxel space comparison

At this point of the algorithm, V_R and V_C are filled and contain informations about the 3D position of M_R and M_C and the goal is to fill V that contains the change. The principle we use consists in comparing V_R and V_C element by element.

In this section, one voxel at the indice j ($j \in \{0..N\}$) is noted $v_R(j)$ and $v_C(j)$ respectively in V_R and V_C. $D_R(j)$ and $D_C(j)$ are its respective stored values. The values in $D_R(j)$ and $D_C(j)$ are furthermore clamped to the interval $< -1...1 >$. Lastly, the voxels in V are noted $v(j)$. We define the change distance field $Diff(j)$ by using inverted distance field:

$$Diff(j) = 1 - |D_R(j) - D_C(j)| \quad (4)$$

$$Sign(j) = D_R(j) - D_C(j). \quad (5)$$

Since we want to obtain the surface of the change, the relevant part of the distance field is located around the zero-crossings of the distance functions.

Thus, if $Diff(j)$ is close to 1, it means that there is no changement from M_R to M_C in $v(j)$. If $Diff(j)$ is close to 0, then $v(j)$ belongs to the border between changement and no changement. Conversely, if $Diff(j)$ is negative, the changement in $v(j)$ is significant. In this last case, the sign of $Sign(j)$ gives information about the type of changement (new or missing object). When $Sign(j)$ is positive, $v(j)$ is a part of a new object in DS_C, and if it is negative, $v(j)$ is a part of a missing object. Figure 3 illustrates the comparison of two voxelspaces in 2D.

The surface S we are interested in is obviously the zero-isosurface and is extracted with the marching cubes algorithm.

5. Experiments and results

In order to evaluate our proposed method, we outline experiments on synthetic data and demonstrate the robustness of our approach on real-world imagery in a practical scenario additionally.

The first experiment is made with a very simple and controlled change that we call synthetic scene, see Section 5.1. The second scene, called real scene described in Section 5.2, consists of real construction site data. Accurate, efficient and fast methods for monitoring building under construction, is a challenging research problem, and image acquisition seems to fit to this domain [7, 2].

5.1. Synthetic scene

We built two datasets ($DS1$ and $DS2$) of 40 and 42 images from the synthetic scene. In each acquisition, the scene was composed of the same objects located at the same place, except a tea box which position changed from $DS1$ (taken as reference data set) to $DS2$ (taken as compared data set). Lastly, every object is known, so we can easily measure their real volume, and we know in advance the desired results.

Figure 4 illustrates the results of our algorithm applied to these two data sets. We can see that the reconstructed meshes of both data sets match with the photos. Just like the difference voxel space which matches with the difference of the two reconstructed meshes: except a few noisy voxels, the blue and red volumes locate respectively the absence and the appearance of the tea box from $DS1$ to $DS2$.

The internal parameters used to obtain this result are: the maximum number of disparities used for plane sweeping is 250; the resulting dimensions of the voxel space is $125 \times 65 \times 49$; only 1 similar view is used to build each range image; 5 complementary views are used in addition to each reference view to fill the voxel spaces. With these parameters, we compared the real volume of the tea box and the detected volume for different reference cameras in DS_R regarding the same scene. To achieve metric scale we obtain the scaling factor from the distance of two known world points. Table 1 gives the results.

Camera ID	9	15	16	23	28
Voxel size	0.051	0.051	0.051	0.052	0.052
Nb. new voxels	190	53	118	153	90
Nb. missing voxels	7781	7825	8201	7650	7344
New volume (cm^3)	12.932	3.604	8.026	10.923	6.409
Missing volume (cm^3)	529.60	532.13	557.83	546.16	522.95
Ratio to tea box volume	1.026	1.031	1.081	1.058	1.014

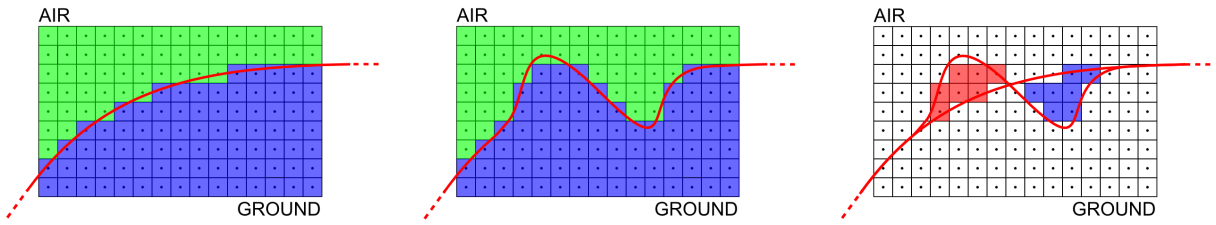
Table 1: To do these measures, we fixed the bounding box so that it contains the tea box in DS_R and nothing in DS_C. Then, we used several cameras (camera ID) to obtain different results. To compute the corresponding volume, we used the scale factor which is 7.986. The real volume of the tea box is $8.5 \times 13.8 \times 4.4 = 516$ cm^3.

The ratios of blue and red voxels show that there is about the same volume that appears and disappears from $DS1$ to $DS2$. Furthermore, the value of this volume corresponds to the volume of the tea box.

5.2. Real scene

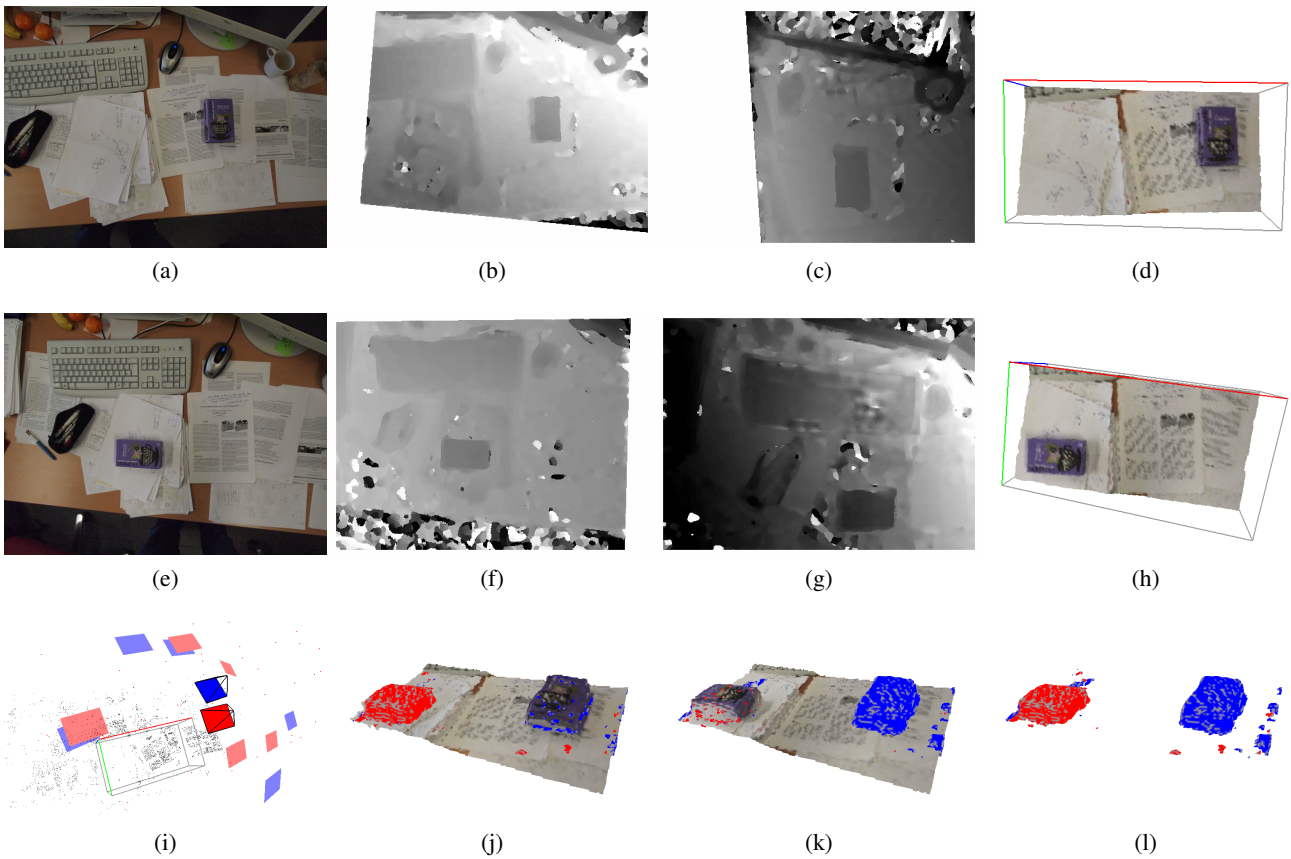
This real-world experiment is conducted on a data set acquired at different timestamps of of a building construction process. The image acquisitions have been made with an unmanned aerial vehicle (UAV) [12, 11, 10]. Each data set was acquired at an individual timestamp over a period of two months and was composed of 146 to 292 high-resolution photos. Figure 5 shows some visual data of the acquired data set at timestamp 1. For each acquisition, lighting conditions were different and the geometric changes result from new or missing structures, moved machines or workers.

The results presented in Figure 6 are obtained by comparing timestamp 1 (reference ; 159 photos) and timestamp 2 (175 photos). For this experiment, we set the maximal number of disparities for plane sweeping to 250, using only 1 similar view for the



(a) (b) (c)

Figure 3: (a) and (b) are V_R and V_C after truncation. The green and blue voxels are respectively negative and positive. (c) is the comparison of V_R and V_C : the red surface is new from DS_R to DS_C and the blue one is missing. In these three images, the red line is the implicit surface which is detected with the marching cubes algorithm.



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

Figure 4: For each data set, six cameras are used (i) to do the reconstruction (in red and blue respectively for DS_R and DS_C). Both reference camera ($C^0_{\{R,C\}}$) are opaque and framed in black. (a-d) show the reference photo (taken from C^0_R), two of the six range images ($R_R^{\{0,1\}}$) used to fill the voxel space, and the reconstructed 3D model for DS_R (seen from C^0_R). (e-h) show the same things for DS_C. (j) is the merging of M_R and the surface S, (k) of M_C and S, and (l) shows S: the red and blue surfaces mark off respectively the new and missing elements from DS_R to DS_C. (Best viewed in color)

generation of the depth images; the resulting voxelspace dimensions are $82 \times 48 \times 30$. Since a metric scale is missing in this experiment we performed measurements to voxel counts where 14601 voxels appeared (that is 12% of the total number of voxels) between timestamp 1 and 2, and 3093 disappeared (2% of the total number of voxels). Figure 6 presents the visual results of the experiment.

5.3. Discussion

To get satisfactory results, we use an increased number of complementary views in the synthetic scene than in the real scene, resulting in an increased

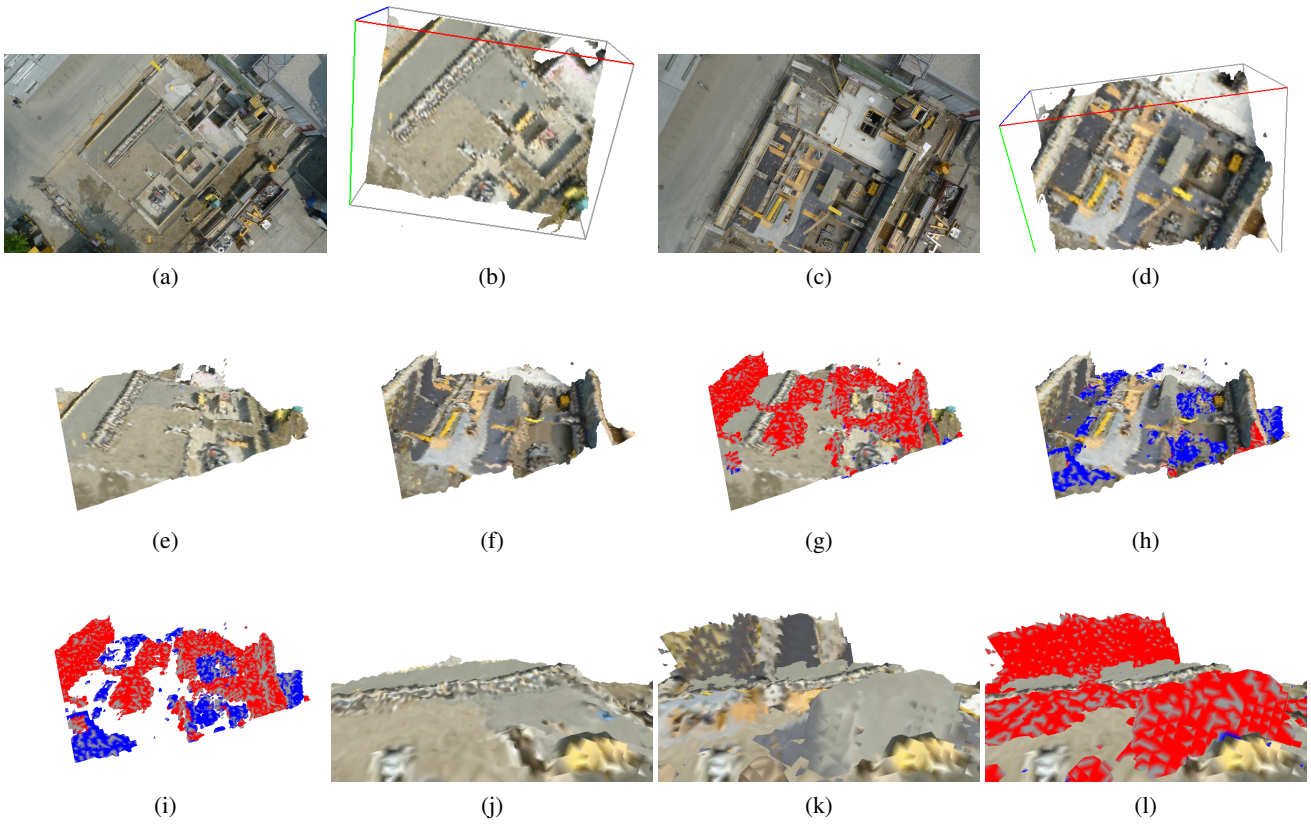


Figure 6: (a) and (b) are the reference photo in DS_R and M_R. (c) and (d) are the reference photo in DS_C and M_C. (e-i) show respectively M_R, M_C, S superimposed on M_R, S superimposed on M_C and S. (j-l) show a zoom on M_R, M_C, and S superimposed on M_R. In each, the red surface delimited the new volume, and the blue surface delimites the missing volume. (Best viewed in color)

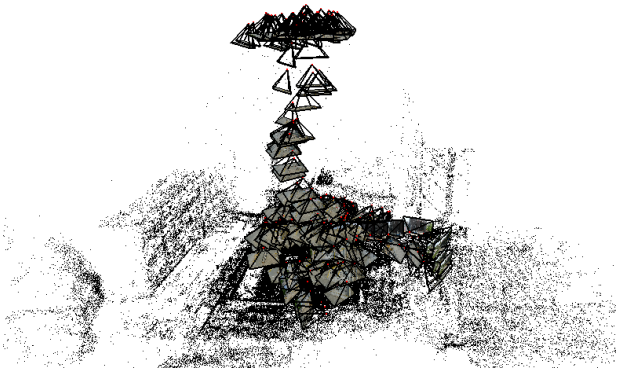


Figure 5: The 3D reconstruction obtained for the real-world scene at timestamp 1. The point cloud and all the camera positions are visible.

computation time. This is necessary because the synthetic scene presents more homogeneous area, which give bad results with the plane sweep algorithm, and less precised range images. But the main reason is that the reference point of view in the reference dataset is above the scene, so there are less occlusions. Without using an optimized implementation, our processing chain needs in the range of 10 seconds to give satisfactory real-world change detection results[1].

6. Discussion and Conclusion

In this work, we have presented a novel method to accurately detect geometric change in 4D reconstructions. Having detected the change over time gives information about the scene in terms of semantics and can be used for further analysis in 3D space. In the synthetic experiment we showed that increasing and decreasing geometric volumes can be detected accurately and that a quantitative evaluation within a metric scale results in acceptable results. The real-world experiments demonstrated practical application scenarios in outdoor scenarios like construction site monitoring. Although we highlighted results

[1]CPU at 2.20GHz, 3GB of RAM

based on photogrammetric 3D reconstructions only, the application of the proposed 4D voxelspace representation e.g. with active sensors like Kinect or a Time-of-Flight is straightforward.

Our proposed method has some limitations that we would like to address in the future. The selection of the complementary cameras is not optimal (we choose them according to their position in comparison with the reference camera). It could be interesting to consider the geometry of the scene to choose the best complementary views, or to improve directly the camera positions during the acquisition [11]. Finally we would like to add semantic informations to the detected volume, by propagating the 3D geometry change back to the input photos. For example, a new object would be labelled and could be selected. Our change detection formulation approach could also benefit from robust volumetric regularization methods. Approaches like the $TV - L_1$ fusion of [23] could be applied to our change detection.

Acknowledgements

This work is supported by Siemens AG Austria and the Austrian Research Promotion Agency (FFG) FIT-IT project Construct (830035).

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *ICCV*, 2009. 1

[2] F. Bosché, C. T. Haas, and B. Akinci. Automated recognition of 3d cad objects in site laser scans for project 3d status visualizationand performance control. *Journal of Computing in Civil Engineering*, 23(6):311–318, 2009. 5

[3] R. Collins. A space-sweep approach to true multi-imagematching. In *CVPR*, 1996. 2, 3

[4] N. Cornelis, B. Leibe, K. Cornelis, and L. v.Gool. 3d urban scene modeling integrating recognition and reconstruction. *IJCV*, pages 121–141, 2008. 1

[5] Y. Cui, S. Schuon, D. Chan, S. Thrun, and C. Theobalt. 3d shape scanning with a time-of-flight camera. In *CVPR*, 2010. 1

[6] B. Curless and M. Levoy. A volumetric method for building complex models from ranges images. *SIGGRAPH*, 1996. 3, 4

[7] M. Golparvar-Fard, F. P. a Mora, and S. Savarese. Monitoring changes of 3d building elements from unordered photo collections. In *ICCV, Workshop on CV for Remote Sensing of the Environment* , 2011. 5

[8] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, 2009. 1

[9] H. Hirschmuller and D. Scharstein. Evaluation of stereo matching costs on images with radiometric differences. *PAMI*, 2009. 2

[10] C. Hoppe, M. Klopschitz, M. Rumpler, A. Wendel, S. Kluckner, H. Bischof, and G. Reitmayr. Online feedback for structure-from-motion image acquisition. In *BMVC*, 2012. 5

[11] C. Hoppe, A. Wendel, A. Irschara, S. Zollman, H. Bischof, and S. Kluckner. Photogrammetric camera network design for micro aerial vehicles. In *CVWW*, 2012. 5, 8

[12] A. Irschara, C. Zach, and H. Bischof. Towards wiki-based dense city modeling. In *ICCV, Workshop on VRML*, 2007. 1, 5

[13] L. Ladicky, P. Sturgess, K. Alahari, C. Russell, and P. H. Torr. What, where and how many? combining object detectors and crfs. In *ECCV*, 2010. 1

[14] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *COMPUTER GRAPHICS*, 21(4):163–169, 1987. 4

[15] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, pages 91–110, 2004. 2

[16] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. W. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *ISMAR*, 2011. 1, 3

[17] T. Pollard and J. L. Mundy. Change detection in a 3-d world. In *CVPR*, 2007. 2

[18] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, pages 7–42, 2002. 2

[19] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. In *SIGGRAPH*, pages 835–846, 2006. 1

[20] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, 1999. 2

[21] A. Taneja, L. Ballan, and M. Pollefeys. Image based detection of geometric changes in urban environments. In *ICCV*, 2011. 1, 2

[22] J. Tian and P. Reinartz. Multitemporal 3d change detection in urban areas using stereo information from different sensors. In *International Symposium on Image and Data Fusion*, 2011. 2

[23] C. Zach, T. Pock, and H. Bischof. A globally optimal algorithm for robust TV-L^1 range image integration. In *ICCV*, 2007. 8

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Flexible Projector Calibration for Fringe Projection based 3D Scanning Systems

Hafeez Anwar, Martin Kampel
Computer Vision Lab
Institute of Computer Aided Automation
Vienna University of Technology
1040, Vienna Austria
hafeezcse@gmail.com
martin.kampel@tuwien.ac.at

Irfanud Din
Department of Electronics Engineering
University of Incheon
Incheon, South Korea
engr_irfan07@yahoo.com

Abstract. *We propose a method for the geometric calibration of a projector. The projector calibration is assisted by an un-calibrated camera. The use of un-calibrated camera is motivated by the fact that camera calibration errors may negatively influence the projector calibration results. Since the projector projects the image instead of capturing, it can be considered as a reverse camera. Therefore the projector can be calibrated in a similar fashion like camera. Regular shaped pattern like chessboard is projected by the projector to a known plane, seen by the camera and then used for calibration. The re-projection error analysis is done for the calculated calibration parameters. For comparison, the re-projection error of the CCD camera is also given.*

1. Introduction

The techniques of 3D shape measurement and reconstruction are vastly used in computer vision, computer graphics, robotics and intelligent manufacturing systems. These techniques are usually adapted by medical, rapid prototyping, defense and other numerous industries. On the basis of their characteristics, these techniques are divided into two subgroups. The first group comprises of passive scanning techniques that use two or more cameras to scan and measure the depth of an object. These cameras are placed at different positions and orientations and take the images of the object simultaneously. Triangulation is then used for measurement of the 3D geometry. The bottleneck in the passive scanning systems is point correspondence. The task is to find image points in two or more camera image planes that correspond to the same 3D points in the scene. To cope with the correspondence problem, various image processing techniques are used.

The correspondence problem is not involved in the active scanning method. In this method the projector projects a structured light on the 3D geometry which is usually captured by a single camera. Most commonly, the sinusoidal or binary fringe patterns are used but there exist many other types of fringe patterns as well [8]. During the past few years a lot of work has been done on active scanning techniques and many researchers have come up with diverse ideas [4][3]. A simple 3D scanning system is shown in Fig.1. Active scanning techniques for reconstructing and measuring the 3D geometry are fast, robust and inexpensive. These days the decreasing prices of projectors and cameras have made it easy to have an affordable 3D measurement system. But before doing any re-construction and measurement, the active scanning system must be calibrated. Camera calibration has been heavily studied by the researchers and different algorithms exist for it [8]. This article focuses on the geometric calibration of the projector which deals with the calculation of intrinsic and extrinsic parameters for the projector.

Before moving to further discussion, the definitions of the two types of calibration patterns are given to avoid any confusion.

- *Printed chessboard*: A regular chessboard pattern of known dimensions printed on a page.
- *Projected chessboard*: A regular chessboard pattern projected by the projector.

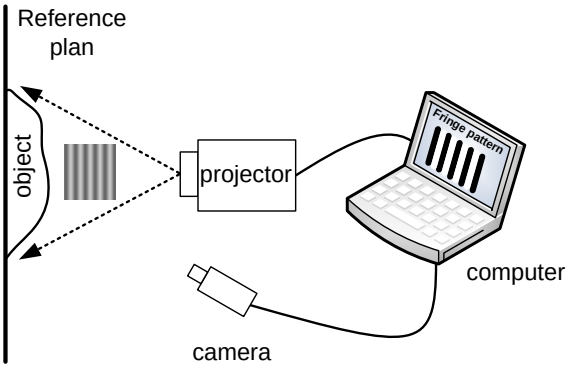


Figure 1. An active 3D shape measurement system based on fringe projection

1.1. Related work

Zhang et al. [9] make the projector able to capture images of the printed chessboard. A camera assists this process and the captured images are then used for calibration. In their method, the main difficulty lies in making the special setup of white and red light illumination. Apart from this, heavy calculations to find the absolute phase map make it a computationally expensive and time consuming method. Li et al. [5] also proposed the calculation of the DMD image i.e. the image of the printed chessboard captured by the projector with the help of a camera. These images are then used to calibrate the projector. They also use vertical and horizontal sinusoidal fringe patterns to recover the points seen by the projector, thus making it a computationally expensive method as well. Gao et al. [2] also generate the image of the printed chessboard seen by the projector using the geometric transformations between camera, screen and projector. They use the red and blue chessboard pattern like Zhang et al. [9]. Apart from that, too many images of the printed chessboard are used because when they move the screen to a different position they must take the image of the printed chessboard attached to the screen in order to know the extrinsic parameters of the camera. Our previously proposed method [1] takes a similar approach as Gao et al. [2] but we make the process of projector calibration much simpler by using image of a single printed chessboard and not using the red and blue light illuminations. However generating the images seen by the projector using the geometric transformations makes that method computationally expensive. Liao et al. [6] have proposed the projector calibration by uncoupling the projector and camera. They use a calibrated camera for calibrating the projector due to which the errors of the camera calibration may affect the calibration of projector. The other limitation is that they use image of the printed chessboard for each position of the screen in order to calculate the extrinsic parameters of the camera.
We propose a simple yet accurate method for projector calibration by,

- making both the screen and camera stationary.
- using an un-calibrated camera to assist the process of projector calibration.
- moving the projector to various positions and orientation instead of the screen.

By adapting the above points we get the following advantages.

- The transformation between camera and screen is fixed therefore we need only one image of the regular chessboard pattern to calculate this transformation.
- The use of un-calibrated camera prevents the errors of the camera calibration to get propagated to the process of projector calibration.
- In methods where the calibrated camera is used [2][6], it is necessary to move the screen to different orientations and positions to take images of the projected chessboard for projector calibration. Apart from that, for each new position of the screen, the image of the printed chessboard attached to the screen must be captured to calculate the extrinsic parameters of the camera. In our case, since we move the projector and the screen is fixed, we only have to capture the image of the projected chessboard for each new position of the projector.
- Moving projector to various positions and orientations makes our method very favorable for the calibration of pocket projectors[7].

Rest of the paper is arranged as follows; section 2 describes the basic concept of planar homography and calibration, section 3 sheds some light on projector calibration, section 4 gives the 3D shape measurement system setup, section 5 gives the results of experiments and their verification in OpenCV and finally the conclusion and future work is given.

2. Basic concept

Suppose a 3D point in the world co-ordinate system with its coordinates with respect to the world plane as Q_w=$\{X_w,\ Y_w,\ Z_w,\ 1\}^T$ is projected to a point q_{cam}=$\{m,\ n,\ 1\}^T$ in the image plane of the camera. Then according to pinhole camera model,

$$q_{cam} = sM_{cam}(R_{cam}|t_{cam})Q_w\ , \qquad (1)$$

where M_{cam} is the set of intrinsic parameters of camera and is given as,

$$M_{cam} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}$$

The parameters (c_x,c_y) are the co-ordinates of the principal focus and f_x and f_y are the focal lengths along the x and y axes of the image plane respectively. $[R_{cam}\ t_{cam}]$ represents the transformation matrix called extrinsic parameters. It gives the rotation and translation between world and camera planes. Extrinsic matrix of a camera is given as follows.

$$[R_{cam}\ t_{cam}] = \begin{pmatrix} r_{11} & r_{21} & r_{31} & t_x \\ r_{12} & r_{22} & r_{32} & t_y \\ r_{13} & r_{23} & r_{33} & t_z \end{pmatrix},$$

or in short,

$$[R_{cam}\ t_{cam}] = \begin{pmatrix} r_{1c} & r_{2c} & r_{3c} & t_{cam} \end{pmatrix}$$

The projector can be considered as a camera that acts in reverse manner. It projects the image instead of capturing it. Therefore the pinhole camera model can be applied to the projector as well. For a 2D-3D point pair the equation of pinhole camera model for projector is given as follows.

$$q_{pro} = sM_{pro}(R_{pro}|t_{pro})Q_w\ ,$$

where M_{pro} is the projectors intrinsic set of parameters and $[R_{pro}\ t_{pro}]$ gives the extrinsic parameters of the projector.

2.1. Planar homography

Planar homography is defined as the projective transformation from one plane to another. In our method, the mapping of known points from the world plane to the camera image plane is an instance of homography. Let 'Q_w' be a point in the world plane and 'q_{cam}' is its corresponding point in the camera image plane. Let both the points be presented as follows.

$$Q_w = [X_w,\ Y_w,\ Z_w,\ 1]^T\ , q_{cam} = [m,\ n,\ 1]^T$$

Based on the pinhole camera model, the relationship between the two corresponding points is given according to (1) as

$$q_{cam} = sM_{cam}(R_{cam}|t_{cam})Q_w\ ,$$

where $(R_{cam}|t_{cam}) = [r_{1c}\ \ r_{2c}\ \ r_{3c}\ \ t_{cam}]$,
(1) can be expanded as,

$$q_{cam} = sM_{cam}[r_{1c}\ \ r_{2c}\ \ r_{3c}\ \ t_{cam}]Q_w \qquad (2)$$

Without loss of generality the world plane can be chosen in such a way that Z = 0. Therefore the 3D point in the world plane becomes,

$$Q_w = [X_w\ \ Y_w\ \ 0\ \ 1]^T$$

So (2) can be written as,

$$q_{cam} = sM_{cam}[r_{1c}\ \ r_{2c}\ \ r_{3c}\ \ t_{cam}][X_w\ \ Y_w\ \ 0\ \ 1]^T\ ,$$

or

$$q_{cam} = sM_{cam}[r_{1c}\ \ r_{2c}\ \ t_{cam}][X_w\ \ Y_w\ \ 1]^T \qquad (3)$$

So planar homography is then a 3×3 matrix written as follows.

$$H = sM_{cam}[r_{1c}\ \ r_{2c}\ \ t_{cam}]\ , \qquad (4)$$

and (3) can be written as,

$$q_{cam} = sHQ_w \qquad (5)$$

2.2. Steps for camera calibration

Zhang's method [10] for camera calibration is followed in this work. This technique uses pinhole camera model which has focal length, pixel size, and skews factor as intrinsic parameters and the translation and rotation of the camera reference frame with respect to the world reference frame as extrinsic parameters. The calibration is simply a process that finds the intrinsic and extrinsic parameters of the camera. A brief description of Zhang's method is as follows.

This method uses a regular shaped object e.g. a chessboard pattern. Let $q_{cam} = \{m,\ n,\ 1\}^T$ be the 2D point in the image plane and Q_w=$\{X_w,\ Y_w,\ Z_w,\ 1\}^T$ be the corresponding 3D point in the world frame of reference. According to the pinhole camera model

$$q_{cam} = sM_{cam}(R_{cam}\mid t_{cam})Q_w \qquad (6)$$

In (6), M_{cam} is the set of intrinsic parameters and $(R_{cam}|t_{cam})$ is the set of extrinsic parameters whereas *'s'* is an arbitrary scaling factor. It is assumed that the model plane is at Z = 0, so (6) becomes

$$q_{cam} = sM_{cam}(r_{1c} \;\; r_{2c} \;\; t_{cam})Q_w$$

The summary of the method is given as follows.

1. A regular shaped object like a chessboard is attached to a flat and smooth plank of plastic.
2. Images of the object are captured by moving it to various positions and orientations.
3. The feature points in the image are detected and stored in a matrix called the *'image points'*.
4. The corners of the object i.e. chessboard are measured in millimeters and stored in another matrix called the *'object points'*.
5. Both of the matrices are then used to find the intrinsic and distortion parameters of the camera.
6. The set of extrinsic parameters of the camera are calculated with the help of the intrinsic parameters.

3. Projector calibration

As it is mentioned earlier that projector acts in reverse manner of a camera, therefore the pinhole model also applies to the projector. According to pinhole camera model,

$$q_{pro} = sM_{pro}(R_{pro}|t_{pro})Q_w \tag{7}$$

For calibration of the projector, a regular chessboard pattern is projected on a screen and its images are then used. Let q_{pro} be the corners of the image of the chessboard to be projected and Q_w be their corresponding projected corners of the chessboard. In (7) q_{pro} can be directly read from the image that the projector is projecting. To find the object points i.e. Q_w of the projected chessboard is a challenging problem. If the chessboard pattern is projected to a *known* plane then the object points can easily be calculated. To make the plane known, a known printed chessboard is attached to the world plane i.e. the screen. It is assumed that the world plane coincides with the frame of reference of the screen. The camera takes an image of this printed chessboard and then the camera to screen homography is calculated. Let Q_w be the point in world plane and q_{cam} be the corresponding point in the camera image plane, then according to (5) the homography from world plane to camera image plane up to a scale factor can be given as;

$$q_{cam} = sH^w_{cam}Q_w \tag{8}$$

The corresponding chessboard corners in world and camera planes are used to compute the 3×3 homography matrix. Once this homography is calculated, a regular pattern is projected by the projector while moving it to different orientations and positions. Here it must be noted that the plane on which the chessboard pattern is projected is known and stationary. For each position of the projector, the projected pattern is imaged by the camera and then the homography from screen to camera is applied to this image to calculate the corners of the projected pattern. In other words these are the calculated object points for each position of the projector. Let for i^{th} position of the projector, q_{pro} be a point in the projector's image plane. This point is projected to an unknown point Q_w in the world plane. The relationships between q_{pro} and Q_w can be given from (7) as follows.

$$q^i_{pro} = sM_{pro}(R_{pro}|t_{pro})Q^i_w \tag{9}$$

The point Q_w is then imaged by the camera. Let the imaged point be q_{cam}. For the i^{th} position of the projector (8) can written as

$$q_{cam} = sH^w_{cam}Q^i_w \,,$$

or

$$Q^i_w = s[H^w_{cam}]^{-1}q_{cam} \tag{10}$$

Once the object points of the projected chessboard are calculated, the image points are directly read from the image to be projected. These image and the object points are then used to calculate the intrinsic and distortion parameters of the projector through Zhang's method [10]. Since the projector is rotated and positioned randomly, the projected chessboard must be seen by the camera. In other words the projected chessboard must lie within the FOV of camera. Fig.2 shows three different positions of the projector. Fig.3 depicts the step-wise flow diagram of the whole process.

Some images of the projected chessboard along with their detected corners are shown in Fig.4. The camera to screen homography is applied to these corners to obtain the object points i.e. corners of the projected chessboard for projector calibration.

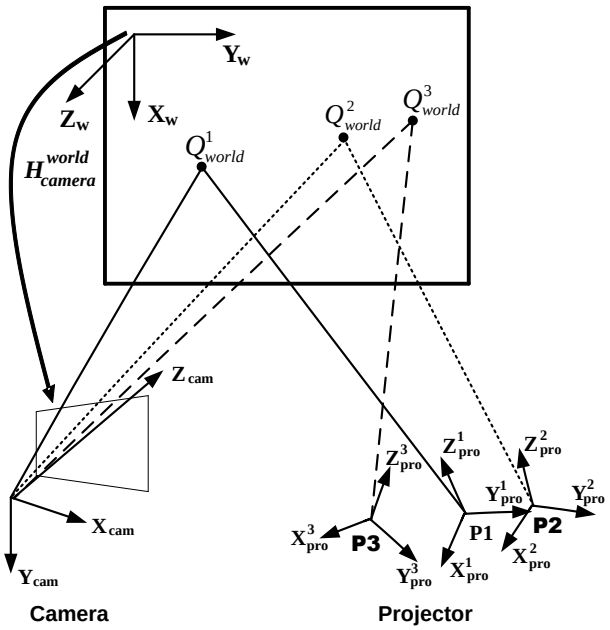


Figure 2. Various orientations and positions of projector w.r.t world

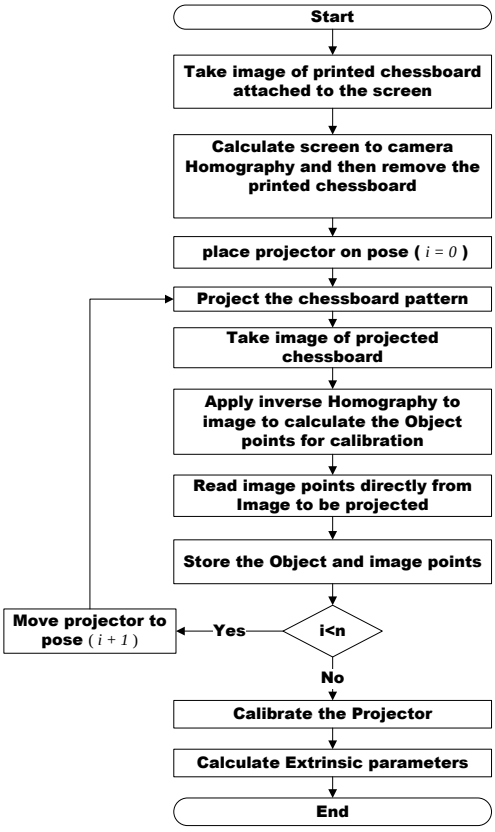


Figure 3. Flow chart for projector calibration procedure

4. Reference frames setup

The relative position of camera and projector need to be set with respect to a world coordinate system i.e. a unique world co-ordinate system between camera and projector needs to be defined. This is done with the help of the same chessboard pattern. The projector is placed normal to the screen and the chessboard is projected on the screen. The camera to screen homography is applied to the projected chessboard to measure the corners of the projected chessboard. Then using these measured corners the extrinsic parameters of the projector are defined. The origin of the projected chessboard is set as the origin of world coordinate. The x-y axes are on the plane and z-axis is perpendicular to the plane. The origin is selected due to the fact that OpenCV starts detecting the corners from that corner. The reference frames of camera, projector and screen are shown in Fig.5.

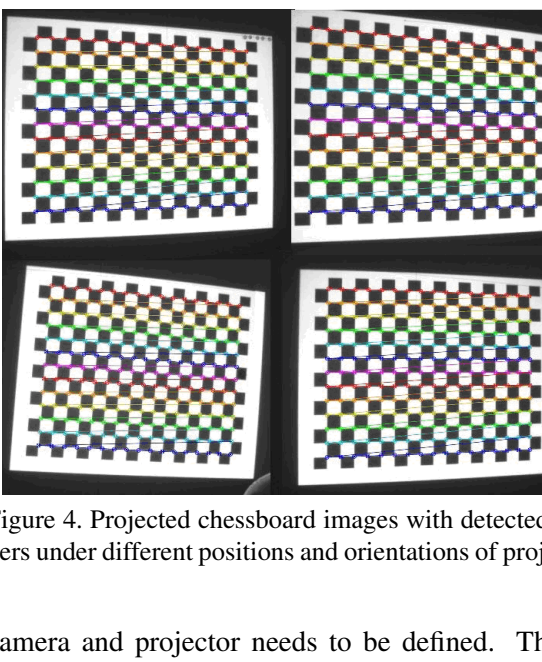


Figure 4. Projected chessboard images with detected corners under different positions and orientations of projector

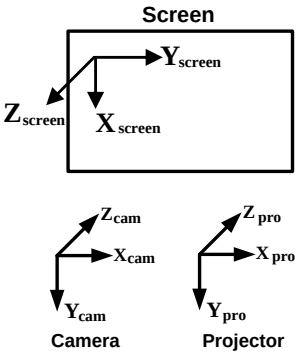


Figure 5. Reference frames of camera, projector and screen

5. Experiments, results and verification

In this work the IDS uEye® CCD camera is used. The resolution of this camera is 1024×768. The LCD projector used is EPSON EB-1735W with a resolution of 1024×768. Fig.6 shows the setup used

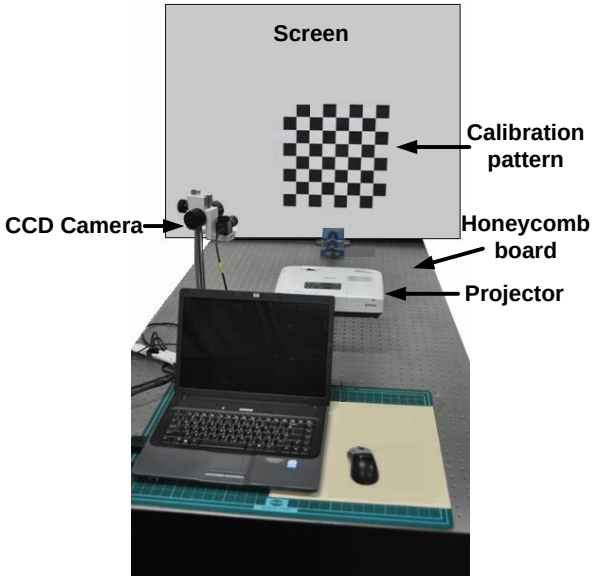


Figure 6. 3D shape measurement system

in this research.

To verify the result, re-projection error is calculated. To do this error analysis we need two images of the regular chessboard pattern. The first image is the *projected chessboard* from projector's image plane to the screen. The second image is the *back-projected chessboard* from screen to projector's image plane. The *back-projected chessboard* image is calculated using the projector's intrinsic and extrinsic parameters. We call the image points of the image to be projected as the *measured image points* and the image points of the back-projected image as the *calculated image points*. Therefore in our terms, the re-projection error is defined as the Euclidean distance between measured image points and calculated image points. Re-projection error is calculated in the following manner.

1. Place the projector normal to the screen.
2. Project the chessboard to the screen and capture its image.
3. Apply the screen to camera homography to calculate the object points.
4. Back-project the object points by applying the calculated intrinsic and extrinsic parameters of the projector to obtain the *calculated image points*.
5. Calculate the Euclidean distance between the *measured image points* and the *calculated image points*.

The re-projection error is shown in Fig.7. Table.1 shows the statistics of the re-projection error in pixels. For comparison, the re-projection error of the camera is shown in Fig.8

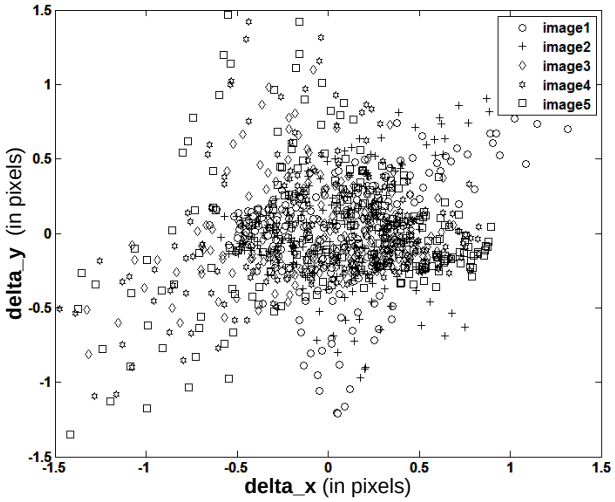


Figure 7. Re-projection error of projector calibration

image no	$Mean^x$	std^x	$Mean^y$	std^y
1	0.013	0.384	-0.001	0.375
2	0.004	0.331	0.004	0.351
3	0	0.397	0.001	0.296
4	-0.010	0.567	0.001	0.456
5	-0.004	0.608	0	0.499

Table 1. Statistics of Fig.7 in pixels

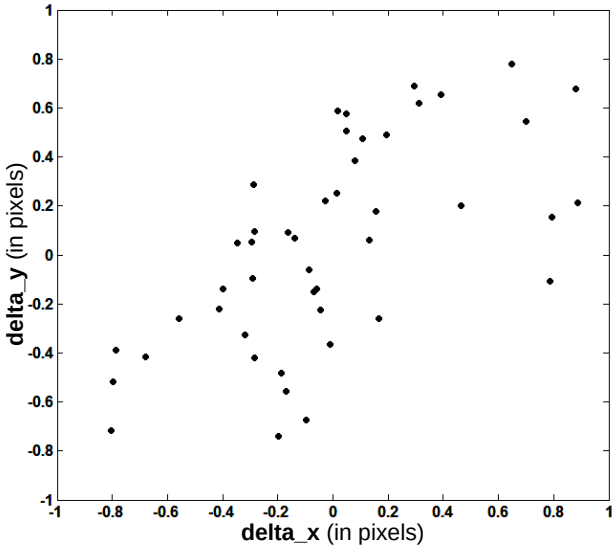


Figure 8. Re-projection error of camera calibration

6. Conclusion

A simple yet accurate framework for projector calibration is proposed. An un-calibrated camera assists the process of projector calibration thus avoiding the errors of camera calibration to get propagated to projector calibration. Both the screen and the camera are fixed and projector is moved to various positions and orientation to take images for calibration. Doing so avoids the movement of the big screen as well as making the camera to screen transformation constant. This constant transformation is then used to calculate the corners of the projected chessboard called the object points. These object points are then used to calculate the intrinsic and extrinsic parameters of the projector. The re-projection error is then calculated for the calculated calibration parameters and for comparison the re-projection error of the camera is given as well.

References

[1] H. Anwar, I. Din, and K. Park. Projector calibration for 3D scanning using virtual target images. *International journal of precision engineering and manufacturing*, 13:125 – 131, 2012. 2

[2] W. Gao, L. Wang, and Z. Hu. Flexible method for structured light system calibration. *Optical Engineering*, 47(8):083602–1 – 083602–9, 2008. 2

[3] S. S. Gorthi and P. Rastogi. Fringe projection techniques: Whither we are? *Optics and Lasers in Engiering*, 48(2):133–140, 2010. 1

[4] Y. Hu, J. Xi, E. Li, J. Chicharo, and Z. Yang. Three-dimensional profilometry based on shift estimation of projected fringe patterns. *Applied Optics*, 45(4):678–687, 2006. 1

[5] Z. Li and Y. shi. Accurate calibration for structured light system. *Optical Engineering*, 47(5):053604–1 – 053604–9, 2008. 2

[6] J. Liao and L. Cai. A calibration method for uncoupling projector and camera of a structured light system. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, page 770, 2008. 2

[7] P. Mistry and P. Maes. Sixthsense: a wearable gestural interface. In *ACM SIGGRAPH ASIA 2009 Sketches*, pages 11:1–11:1, 2009. 2

[8] J. Salvi, X. Armangué, and J. Batlle. Pattern codification strategies in structured light systems. *Pattern Recognition*, 37(4):827–849, 2004. 1

[9] S. Zhang and P. Huang. Novel method for structured light system calibration. *Optical Engineering*, 45(8):83601–83608, 2006. 2

[10] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientation. In *International conference on Computer Vision*, page 666, 1999. 3, 4

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Line-based 3D Reconstruction of Wiry Objects

Manuel Hofer, Andreas Wendel, Horst Bischof
Institute for Computer Graphics and Vision
Graz University of Technology, Austria
{hofer,wendel,bischof}@icg.tugraz.at

Abstract. *Man-made environments contain many weakly textured surfaces which are typically poorly modeled in sparse point reconstructions. Most notable, wiry structures such as fences, scaffolds, or power pylons are not contained at all. This paper presents a novel approach for generating line-based 3D models from image sequences. Initially, camera positions are obtained using conventional Structure-from-Motion techniques. In order to avoid explicit matching of 2D line segments in the various views we exploit the epipolar constraints and generate a series of 3D line hypotheses, which are then verified and clustered to obtain the final result. We show that this approach can be used to densify various sparse occupied point clouds of urban scenes in order to obtain a meaningful model of the underlying structure.*

1. Introduction

Generating 3D models from a set of images has become a widely studied field of research over the last few years. The majority of available algorithms is based on point correspondences between multiple views using various local descriptors such as the Scale-Invariant Feature Transform (SIFT) [13] in order to obtain a 3D point cloud while simultaneously estimating the camera parameters. This process is called Structure-from-Motion (SfM). The density of the resulting point cloud highly depends on the amount of texture available in the images. Therefore, point-based SfM may fail in man-made environments with a low amount of distinctive interest points (e.g. urban scenes, indoor scenes). To tackle this issue, many line-based approaches have been presented over the years, due to the fact that especially man-made objects (e.g. buildings) can usually be represented by a set of 3D line segments. Similar to traditional SfM it is usually necessary to match 2D line segments from various views to triangulate a 3D line segment. This can be done using appearance-based similarity measures, e.g. normalized-cross-correlation (NCC) or line descriptors [12, 21], which can be combined with additional geometric constraints [3]. Since the endpoints of matched line segments usually do not correspond to each other due to inexact line segment detection or occlusions, creating 3D line segments from matched 2D lines is much more difficult than traditional point-to-point matching.

Figure 1. Two examples for wiry structures. The left image shows a power pylon and the right image a scaffold in front of a house.

Most of the previous approaches rely on an accurate line matching process between the various views using some appearance-based similarity measures. This usually works fine if the lines are located on a planar surface with constant background, for instance when matching window frames. However, when dealing with wiry structures such as power pylons, bridges or scaffolds (see Figure 1 for some examples), appearance-based matching is hardly possible due to changing surroundings of the line segments in different views (see Figure 2). We present an approach which is especially designed to handle such cases but also performs well on solid objects.

Figure 2. An example were no appearance based line matching can be performed. Note that corresponding line segments have different surroundings in both views (yellow lines).

2. Related Work

In the following we present selected papers from the field of line-based 3D reconstruction. We start with an overview of appearance-based methods which cannot directly be applied to our problem but share some ideas with our approach.

Baillard et al. [1] presented a method which makes use of the epipolar constraint by estimating line correspondences along the epipolar beam. To find the correct match they evaluate the NCC score for candidate lines using patches around the line segments. The estimated 3D line segment is the intersection of the half-planes through the lines of sight of the two endpoints in both views. They further verify their hypotheses by minimizing the reprojection error using the trifocal tensor [6].

Bay et al. [2] use optional region matches in addition to line matches based on color histograms in order to establish an initial set of candidates. They apply a topological filter in order to remove wrong candidates and increase the candidate set by adding unmatched line segments which fit to the topological structure of the already matched hypotheses. They further estimate the epipolar geometry using coplanar subsets of their candidate set. Very accurate results are reported, even for sparsely textured scenes.

In order to generate 3D line models for urban scenes, Schindler et al. [16] proposed an approach which takes vanishing point information into account. They assume that relevant edges are located along mutually orthogonal vanishing directions which reduces the degrees of freedom for 3D line estimation. Their approach delivers pleasant results for urban structures but unfortunately is limited to pictures taken at near-ground level due to their assumptions.

Another approach presented by Kim et al. [8] is based on the intersection context of coplanar line pairs. They match line intersection context features across multiple views using NCC as similarity measure and reject false intersections using coplanarity constraints on the corresponding line segments. The proposed method works well for a wide range of scenarios even when only little texture is available.

Unfortunately, all of these appearance-based approaches usually do not perform well for wiry structures, since they technically do not match the line itself, but rather its surroundings. In our case, explicit matching may be impossible, since the ever changing background is not coplanar with the line and often very far away from the object to be reconstructed. In order to create 3D models without the need of explicit line matching, Jain et al. [10] developed a sweeping based approach which defines the unknown 3D locations of the endpoints of 2D line segments as random variables. They estimate 3D line hypotheses by generating all possible endpoint locations in a certain depth interval (assuming known camera intrisics and extrinsics) and keep the one with the highest score based on the gradient images of many neighboring views. Hence, they create a 3D line for every 2D line in every view. In order to delete outliers and cluster corresponding line segments together, they group 3D line segments which lie close in space and discard all segments which do not have at least one such neighbor. They also perform an optimization based on 2D line connections using loopy belief propagation to enforce connected 3D lines. Even though their approach delivers very accurate results and is very robust against noise and partial occlusions, it is very slow compared to previous approaches.

In our approach we build upon the principles presented in [10] but use a different matching strategy. Instead of using a time consuming sweeping approach we generate hypothetical 3D line segments using epipolar constraints, which drastically limits the number of possible 3D locations for each 2D line segment. We will show that this leads to a significant performance increase while still creating accurate results.

3. Sparse Structure-from-Motion

Given an unordered set $I = \{I^1, ..., I^n\}$ of n images and the corresponding cameras $C = \{C^1, ..., C^n\}$ we want to generate a set of 3D line segments $S = \{S^1, ..., S^k\}$. Since we do not perform explicit line matching and line-based relative

pose estimation the cameras have to be known beforehand. For this purpose we use a point-based SfM system. This limits the application to scenes where interest points can be found, but we have seen that we can usually find enough correct correspondences for an accurate relative pose estimation in the background of wiry structures.

We follow the approach of Wendel et al. [20] and Irschara et al. [9] which enables us to perform sparse SfM for unordered image sets. The three necessary processing steps are feature extraction, feature matching, and geometry estimation. In the first step we extract SIFT [13] features from all images. SIFT has been shown to work well in general scenes [14], but it also works surprisingly well in scenarios with wiry structures. The reason is that matches are obtained either in the background, or in the foreground in case of a homogeneous background such as sky. Afterwards, we match the resulting keypoint descriptors between all possible image pairs and perform a geometric verification procedure using the Five-Point algorithm [15]. In order to eliminate possible outliers we use RANSAC [5] for robust estimation. The resulting pairwise reconstructions are then merged to obtain a sparse reconstruction of the scene. Finally, bundle adjustment [17] is applied to minimize the global reprojection error over all measurements. See [19] and [7] for further details.

As a result we know the relative positions of all cameras C in a common coordinate frame, and we can thus proceed to the task of 3D line segment estimation.

4. Reconstruction of 3D Line Segments

Our algorithm consists of three steps: 2D line segment detection extracts line segments from each input image, 3D line segment hypotheses generation tries to estimate the 3D position of these segments, and finally 3D line grouping and outlier removal merges corresponding segments from different views and removes incorrect estimates. In the following sections these steps will be explained in detail.

4.1. 2D Line Segment Detection

In order to generate triangulated 3D line segments from a set of images, we first have to apply a line segment detection algorithm onto our input images. We employ the Line Segment Detector (LSD) [18] algorithm to extract all relevant line segments with as few incorrect detections as possible. The authors report their algorithm to be significantly faster than previous methods while producing very accurate results. Their approach is based on the grouping of points with a high gradient and similar level line angle, followed by a least squares line fit. All detections are validated using the Helmholtz principle [4] which proves to be very effective for the general case. Figure 3 shows the detected line segments for a power pylon image.

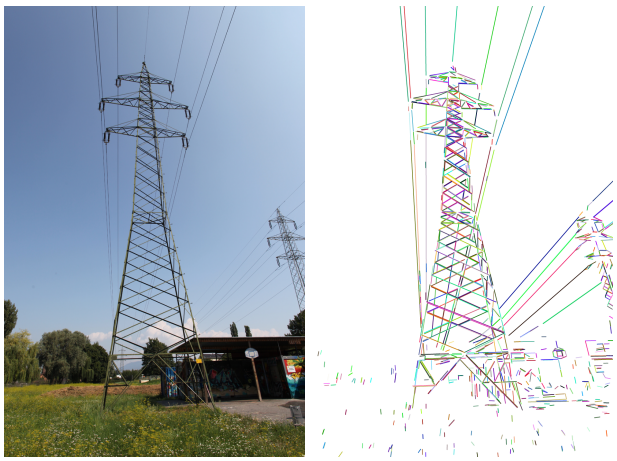


Figure 3. Line Segment Detection. The line segments extracted using the *LSD* [18] algorithm are visualized in pseudo-colors. The underlying wiry structure is represented very well, except for a few outliers due to noisy gradients, cause for instance by grass.

4.2. 3D Line Segment Hypotheses Generation

Assuming no false detections in the previous step, every 2D line segment from image I^i corresponds to a 3D line segment in world space. Since we can not perform an explicit appearance-based matching procedure and triangulation, we have to estimate the correct 3D location of each segment in a different way.

As we know the projection matrix P^i of the camera, we are able to compute the *epipolar geometry* between I^i and some other view I^j. Using the epipolar lines e_p and e_q defined by the two endpoints p and q of a certain line segment l in view i, we can limit the possible matches for l to those line segments whose endpoints lie on e_p and e_q respectively. In practice it is unlikely that we will find an exact match with both endpoints being located exactly on the epipolar lines, e.g. due to imprecise line detection or occlusions. Therefore we extend all candidate segments which overlap with the region between the two epipolar lines to infinity (from line segments to actual lines) and intersect them with e_p and e_q in order to generate hypothetical matches. This enables us to

find correct matches even if the current line segment is shorter or longer in I^j (see Figure 4). For every hypothesis we create a 3D line segment L by triangulating the two corresponding endpoint pairs from the two views I^i and I^j.

Since we usually have more than one hypothesis for each 2D line segment (because the epipolar lines do not provide enough information to perform exact matching), we have to determine which one is correct. Therefore we adopt a gradient based scoring approach similar to [11, 10]. We then backproject each 3D line segment L into all neighboring views $N(I^i)$ of I^i with a camera center closer than a certain distance d_c and an absolute viewing angle difference smaller than d_{ang} to the current camera C^i. For each camera we compute a set of measurement points M along and perpendicular to the backprojected line, and compute the image gradient-based score

$$s(L) = \frac{1}{|N(I^i)|} \sum_{I \in N(I^i)} \sum_{x \in M(I)} \frac{\|\nabla I(x)\|}{|M(I)|} e^{-(\frac{\lambda \cdot dist(x,L)}{2 \cdot dist_{max}(L)})^2} \quad (1)$$

for every 3D line segment L, where $\nabla I(x)$ denotes the image gradient at position x, $dist(x, L)$ is the perpendicular Euclidean distance to the backprojected line in the current image I and $dist_{max}(L)$ denotes the maximum distance based on the configuration of the measurement points. Assuming that line segments correspond to high gradient areas in images, this method ensures that we choose the hypothesis which fits best to the image data. Using this formula we give more weight to measurement points which are closer to the backprojected line, and less weight to those perpendicular to it depending on the distance. An illustration is given in Figure 5.

After computing the score for each hypothesis we choose the one with maximum score, denoted as L_{best}, which is then added to our 3D line segment hypotheses set H. Since we generate 3D line segments for all views individually, we end up with a quite large hypotheses set which has to be pruned. Figure 6 shows an example for a 3D line model before grouping and outlier removal.

4.3. 3D Line Grouping and Outlier Removal

It is possible that the correct matches for 2D line segments are not among the candidates, for instance because the line segments are not redetected in any neighboring view, and therefore we have to remove possible outliers. The outlier removal process goes hand in hand with the line grouping step which has to be performed in order to remove multiple detections. Since we match and triangulate the 2D line segments individually for every view, the same 3D line might be generated in multiple views. Assuming a correct matching procedure, all the hypotheses in H which correspond to the same 3D line should be located close in space. Hence, a line clustering algorithm is performed in order to generate the final 3D line model.

In order to remove incorrectly triangulated 3D line segments and cluster corresponding segments, we adopt the idea of spatial proximity based grouping from [10]. First, we order the hypotheses set H by

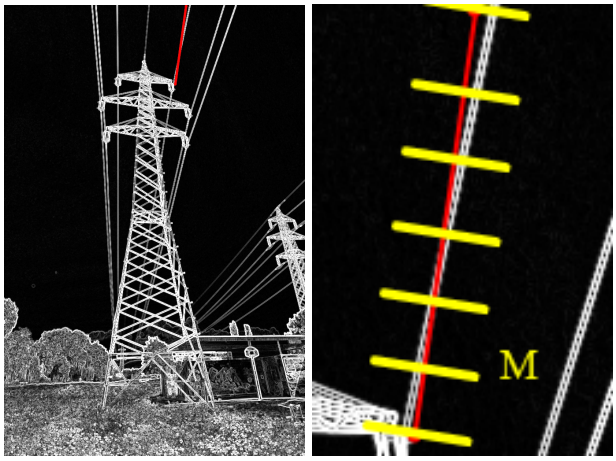


Figure 5. The left image shows the gradient magnitudes from a power pylon image with a backprojected 3D line hypothesis shown in red. The right image shows a close-up of the line segment with the set of measurement points M illustrated as yellow lines. The weighted sum of the gradient magnitudes over all measurement points is computed and then divided by the number of points in order to compute the score for this view. The average score over all neighboring views is then used to evaluate the best hypothesis (see Equation 1).

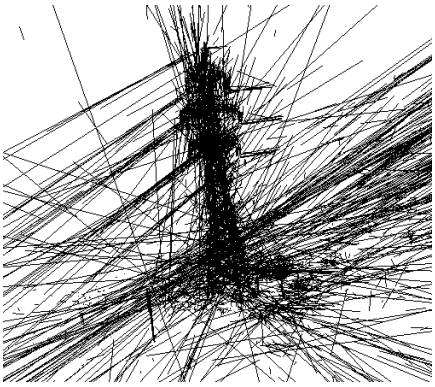


Figure 6. 3D line segment hypotheses before the grouping and outlier removal procedure. Our approach generates 71538 segments from 106 views. Note that there is a large number of outliers due to incorrect matches, but the power plyon which appears in the imagery is clearly recognizable.

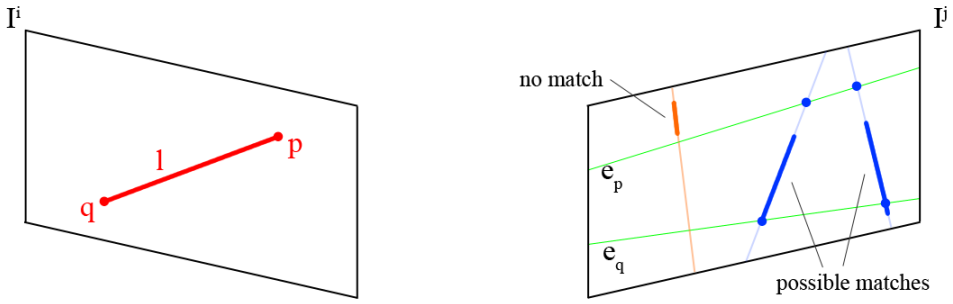


Figure 4. We match the line segment L in view I^i with line segments from view I^j using its epipolar lines e_p and e_q. The blue line segments are possible candidate matches because of their overlap with the region between the two epipolar lines. The endpoints of the hypothetical line segments used for triangulation are shown as blue dots. The orange line segment does not overlap with the epipolar lines and is therefore not considered to be a possible match.

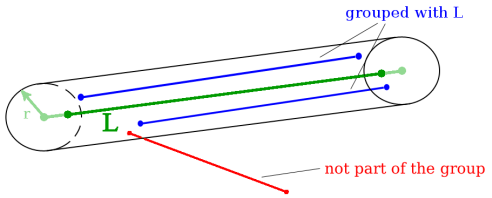


Figure 7. To group corresponding 3D line segments together, the true segment L (green) is expanded by 10% in each direction. All other line segments with both endpoints within a cylinder of radius r, defined by the new endpoints, are considered to be in the same group as L (blue lines). The red line does not belong to the group, because one of its endpoints is outside the cylinder.

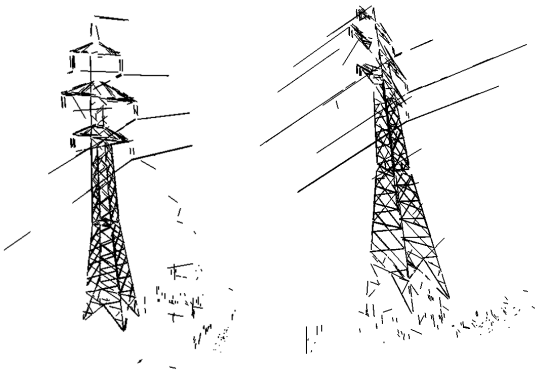


Figure 8. After the grouping procedure most of the outliers have been successfully removed resulting in an accurate 3D model (1381 line segments).

score in descending order to start grouping with lines which are best aligned with the image gradients. For each line $L_m \in H$ we define a cylinder of a fixed radius r by expanding the central axis of the line segment by 10% in both directions. We then try to find all line segments $L_n, n \neq m$ where both endpoints are located within the cylinder (Figure 7).

If the final line group (including L_m) has at least h_{min} members we consider it to be valid and exclude all line segments in the group from further grouping, otherwise L_m is removed from H and we continue with the next best hypothesis.

After the clustering step, each group is replaced with one single line segment for our final 3D line segment set S. To define this line we first compute the center of gravity of all line segment endpoints from the group. Afterwards we perform a singular value decomposition of the scatter matrix containing all endpoints and take the eigenvector corresponding to the maximum eigenvalue as new line direction. We now project all endpoints onto the new line and add the line segment defined by the two outmost points to S. Figure 8 illustrates the outcome of the grouping procedure. Note that compared to Figure 6 most of the outliers have been successfully removed.

5. Experiments

In the previous section we have already shown a resulting 3D model of a power pylon. In this section we want to present additional results and finally compare our algorithm to [10] using one of their test-cases.

5.1. Parameter Selection

The various steps of our approach require a set of parameters in order to generate pleasant results. Most of them are valid for a large number of scenarios and therefore do not need to be especially tuned.

The line segment detection algorithm (LSD [18]) does not need parameters. In order to eliminate outliers and speed up the computation we reject 2D line segments smaller than 1% of the diagonal length of the image in pixels, which is usually sufficient to capture the underlying structure of our images.

During hypotheses generation we need to deter-

mine which views are considered neighboring views $N(I^i)$ for the current view I^i (see Section 4.2). The maximum viewing angle difference $d_{ang} = 50°$ and the maximum distance between the camera centers $d_c = 30$ for all our experiments. In order for the second parameter to make any sense we need to know the scale of our 3D model. In our experiments the result achieved during preprocessing (camera estimation) is transformed to a metric scale ($1 \equiv 1m$), using either a marker with known size [11] or manual user interaction. Assuming equidistant camera centers we usually have a large number of views available for scoring. In order to increase the performance (since scoring has to be done for every 2D line segment in every view) we limit the number of neighboring views to 20.

For the scoring procedure we choose the set of measurement points (M) in a way that the distance between the points on the backprojected line is 5 pixels. The number of perpendicular points is set to 5 in each direction (with a distance of 1 pixel), meaning that $dist_{max} = 5$. The parameter λ is set to 10 (see Equation 1).

The parameters for the grouping procedure are the only ones which have to be estimated for each testcase individually. For most scenarios setting $r = 0.05$ and $h_{min} = 3$ yields good results, meaning that the grouping radius is $5cm$ and every 3D line segment has to be correctly estimated in at least 3 different views.

5.2. Results

In traditional point-based SfM it is often the case that the resulting point cloud is sparsely distributed due to the lack of distinctive features especially for man-made structures. Many of the keypoints may be rather located on the background instead of the object. Nevertheless, background features can be used for relative pose estimation and therefore our line matching algorithm can be applied in order to densify the 3D model.

Figure 9 shows an example 3D point cloud of a house surrounded by a scaffold, and Figure 10 shows a model of a staircase. As we can see, the point clouds are rather sparsely occupied and the visible objects are difficult to determine for the viewer. Adding 3D line segments clearly improves the result and allows the viewer to identify the underlying structure.

Our algorithm is designed to handle wiry structures. Nevertheless, it is not limited to such scenarios and can also handle solid objects. In order to compare our approach to [10], we reconstructed their *Timber-frame house* sequence[1] using our algorithm. The sequence consists of 240 synthetic images. Figure 11 shows exemplar views from the sequence along with our 3D reconstruction and the result from Jain et al. [10], colored using the Hausdorff distance as similarity measure (for densely sampled points along the lines) to the ground truth model. Table 1 shows the root mean square (RMS) error for both reconstructions compared to the CAD model.

As we can see, both algorithms manage to recon-

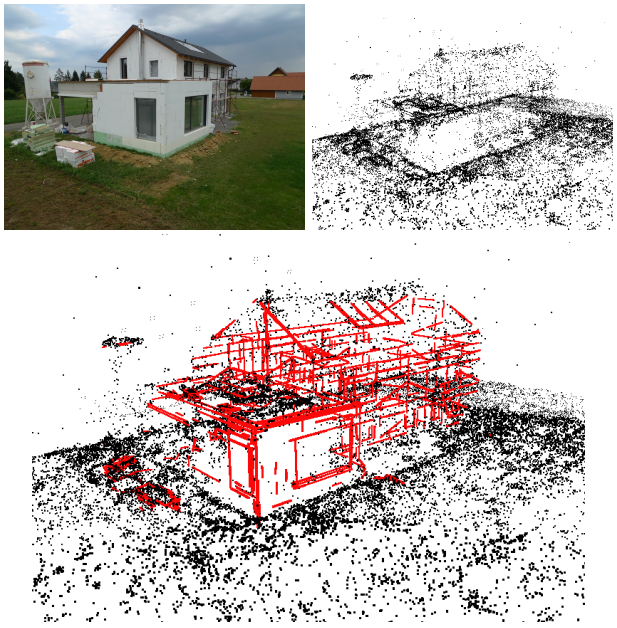


Figure 9. The top images show an example view from a house sequence (93 images) and a SfM point cloud. The bottom image shows the densified 3D model with the reconstructed line segments.

Figure 10. The left image shows an example view from a staircase sequence (14 images). The right image shows the densified 3D model with the reconstructed line segments.

[1] http://www.mpi-inf.mpg.de/resources/LineReconstruction/

Method	min error	max error	RMS error
Jain et al.	0.000	0.019	0.0036
Ours	0.000	0.023	0.0013

Table 1. The comparison to the method by Jain et al. [10] revealed that our method performs better in terms of the RMS error but their method has a slightly lower maximal error.

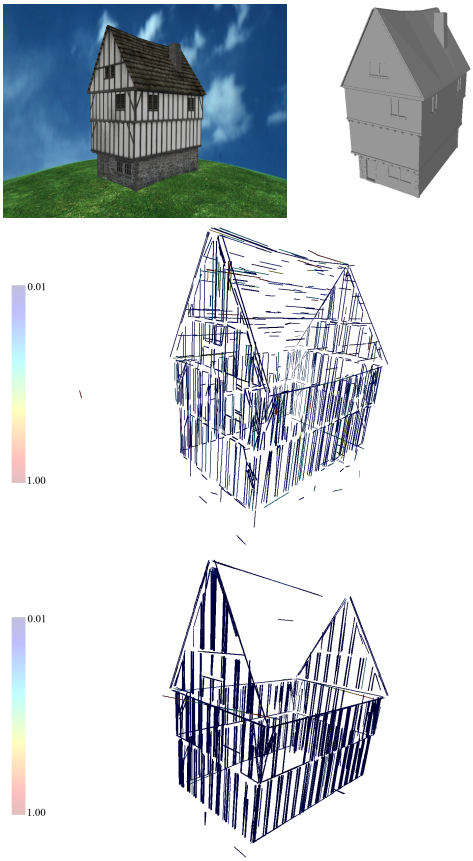


Figure 11. The top images show an example view from the synthetic Timber-frame house sequence (240 images) along with the ground truth CAD model. The middle image is the reconstruction achieved by [10], the bottom image is our reconstruction. The color reveals the errors compared to the CAD model (from 0.01 to 1.00). Best viewed digitally and in color.

struct the building in a qualitatively accurate way. Our approach performs better in terms of the RMS error, while Jain et al. are able to reconstruct a few more lines, especially on the roof. Even though the resulting models are similar, the computational time differs highly. The authors report that their algorithm often needs several hours to deliver the result, while our method is able to reconstruct this sequence in 7.5 min using all 240 images and not only a subset of 72 as [10].

5.3. Performance Evaluation

Since we have to evaluate many possible matches for each 2D line segment (to avoid appearance-based matching) our algorithm is more time consuming than traditional line-matching approaches. Nevertheless, we manage to generate accurate results for the general case in reasonable time. Table 2 shows a performance evaluation for the three test sequences presented in this paper. All experiments were performed on a desktop PC equipped with an Intel Core2 4×2.66 GHz processor. Note that for our own sequences (Pylon, House and Staircase) the image size was significantly larger than for the Timber-frame house, which explains the difference in speed.

6. Conclusion

We have presented a novel approach for the purpose of generating 3D line models without explicit appearance-based matching. The proposed algorithm performs well for wiry structures as well as solid objects. In contrast to a previous approach by Jain et al. [10], we exploit epipolar constraints to speed-up the computation while still creating accurate results. We have shown that for scenes with few keypoints located on the actual foreground object, 3D line segments can be used to densify the resulting model. This is of particular importance for urban scenes and man-made structures which often provide few distinctive feature points.

While our approach is able to generate 3D line segments even when a 2D segment is not exactly redetected in any other view (due to the matching strategy based on epipolar lines), it usually generates a large set of outliers. These outliers have to be removed in a computationally expensive grouping step, which may take a lot of time depending on the number of hypotheses. Therefore, our future work will be to formulate the matching procedure in a probabilistic way to allow online hypotheses generation, in order to further improve the performance.

Acknowledgements

This work has been supported by the Austrian Research Promotion Agency (FFG) project FIT-IT Pegasus (825841/10397) and OMICRON electronics GmbH.

References

[1] C. Baillard, C. Schmid, A. Zisserman, and A. Fitzgibbon. Automatic line matching and 3d re-

Sequence	#img	Resolution	#lines raw	#lines final	LSD	Matching	Grouping	Total
Pylon	106	5616×3744	71538	1381	17.2	15.5	34.1	66.8
House	93	3048×2736	30967	1262	13.8	3.2	6.7	23.7
Staircase	14	3072×2304	16517	265	2.7	0.8	1.1	4.6
Timber-f. H.	240	1280×960	29927	2113	4.2	2.5	0.8	7.5

Table 2. This table shows a performance evaluation for the test sequences used in this paper. All times are in minutes (last four columns).

construction of buildings from multiple views. In *ISPRS Conference on Automatic Extraction of GIS Objects from Digital Imagery*, volume 32, 1999. 2

[2] H. Bay, V. Ferrari, and L. Van Gool. Wide-baseline stereo matching with line segments. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, 2005. 2

[3] F. Bin, W. Fuchao, and H. Zhanyi. Robust line matching through linepoint invariants. *Pattern Recognition*, 45(2), 2012. 1

[4] A. Desolneux, L. Moisan, and J.-M. Morel. Meaningful alignments. *International Journal of Computer Vision*, 40, 2000. 3

[5] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communication Association and Computing Machine*, 24(6), 1981. 3

[6] R. Hartley. A linear method for reconstruction from lines and points. In *Fifth International Conference on Computer Vision*, 1995. 2

[7] C. Hoppe, M. Klopschitz, M. Rumpler, A. Wendel, S. Kluckner, H. Bischof, and G. Reitmayr. Online feedback for structure-from-motion image acquisition. In *In Proceedings of the British Machine Vision Conference (BMVC)*, 2012. 3

[8] K. Hyunwoo and L. Sukhan. A novel line matching method based on intersection context. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2010. 2

[9] A. Irschara, V. Kaufmann, M. Klopschitz, H. Bischof, and F. Leberl. Towards fully automatic photogrammetric reconstruction using digital images taken from uavs. In *Proceedings International Society for Photogrammetry and Remote Sensing Symposium, 100 Years ISPRS - Advancing Remote Sensing Science*, 2010. 3

[10] A. Jain, C. Kurz, T. Thormaehlen, and H. Seidel. Exploiting global connectivity constraints for reconstruction of 3d line segments from images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 2, 4, 5, 6, 7

[11] T. Kempter, A. Wendel, and H. Bischof. Online model-based multi-scale pose estimation. In *Proceedings of the Computer Vision Winter Workshop (CVWW)*, 2012. 4, 6

[12] B. Khaleghi, M. Baklouti, and F. Karray. SILT: Scale-invariant line transform. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, 2009. 1

[13] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60, 2004. 1, 3

[14] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(10), 2005. 3

[15] D. Nister. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6), 2004. 3

[16] G. Schindler, P. Krishnamurthy, and F. Dellaert. Line-based structure from motion for urban environments. In *Third International Symposium on 3D Data Processing, Visualization, and Transmission*, 2006. 2

[17] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment a modern synthesis. *Vision Algorithms: Theory and Practice*, 2000. 3

[18] R. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. LSD: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4), 2010. 3, 5

[19] A. Wendel, C. Hoppe, H. Bischof, and F. Leberl. Automatic fusion of partial reconstructions. In *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2012. 3

[20] A. Wendel, A. Irschara, and H. Bischof. Automatic alignment of 3d reconstructions using a digital surface model. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), Workshop on Aerial Video Processing*, 2011. 3

[21] W. Zhiheng, W. Fuchao, and H. Zhanyi. MSLD: A robust descriptor for line matching. *Pattern Recognition*, 42(5), 2009. 1

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Localizing and Segmenting Objects with 3D Objectness

Aitor Aldoma, Markus Vincze
Vision 4 Robotics, Technical University of Vienna
aldoma@acin.tuwien.ac.at

Federico Tombari
DEIS, University of Bologna

Walter Kropatsch
PRIP, Technical University of Vienna

Abstract. *This paper presents a novel method to localize and segment objects on close-range table-top scenarios acquired with a depth sensor. The method is based on a novel* objectness *measure that evaluates how likely a 3D region in space (defined by an oriented 3D bounding box) could contain an object. Within a parametrized volume of interest placed above the table plane, a set of 3D bounding boxes is generated that exhaustively covers the parameter space. Efficiently evaluating — thanks to integral volumes and parallel computing — the 3D objectness at each sampled bounding box allows defining a set of regions in space with high probability of containing an object. Bounding boxes characterized by high objectness are then processed by means of a global optimization stage aimed at discarding inconsistent object hypotheses with respect to the scene. We evaluate the effectiveness of the method for the task of scene segmentation.*

1. Introduction and related work

Accurate robotic perception is a fundamental feature for most envisioned application scenarios related to service and industrial robotics. The capability of segmenting a scene perceived by a sensor onboard a robotic agent into a set of coherent patterns (or objects) is a classical - though challenging - step standing at the grounds of numerous tasks related to robotic perception such as 3D object recognition, point cloud registration, object grasping and manipulation. As commonly deployed onboard most robotic architectures, we assume the presence of a 3D perception system, acquiring RGB-D data (a color frame plus an associated *organized* point cloud), as well as that of a dominant plane in the scene, which can be represented by either the ground floor or a table on which objects are lying. The assumption of a dominant plane has been extensively used in the field of robotic perception to speed up segmentation such as in [6, 1, 2]. Other 3D segmentation methods without the dominant plane assumption are those presented in [8, 9]; even though they are more general than those constrained by the dominant plane assumption, they are characterized by a higher computational complexity.

Under these conditions, we have devised a novel algorithm aimed at automated localization of *salient* volumes from the data related to the scene currently in front of the robot. Our definition of saliency is driven by the concept of *objectness*, i.e. a portion of volume of the analyzed 3D space is salient if the characteristics of the surface therein enclosed have a high probability of representing an object, and viceversa. To this aim, the first contribution of this work is the definition of an objectness measure for 3D data which can be computed on a 3D bounding box of generic dimensions, inspired from the work of Alexe et al. [4], who proposed an analogous measure for images. Based on our definition of objectness, we then propose an effective optimization framework to simultaneously detect the presence of several salient bounding boxes in a 3D scene, which is able to discard unrealistic object hypotheses such as objects intersecting one another or bounding boxes that do not fit tightly the object surface. Although our method is not explicitly aimed at segmentation, in order to provide a quantitative and qualitative experimental analysis, we compare the results of our approach with those of state-of-the-art segmentation methods for point clouds, demonstrating the effectiveness of our proposal. We expect our method to prove useful as a pre-processing step of 3D object recognition

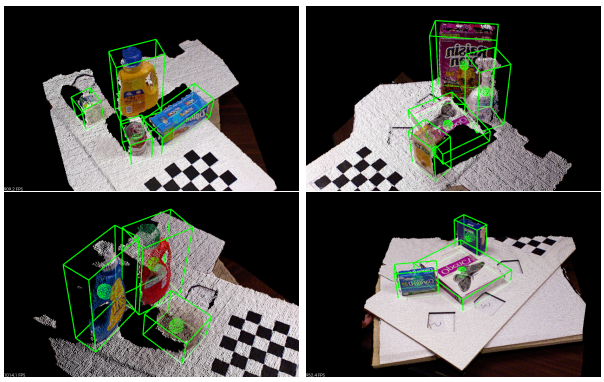


Figure 1: Typical results obtained by the proposed method. The green bounding boxes show the regions selected by the method that are likely to contain an object. The green sphere depicts the center of the bounding box.

algorithms, in order to reduce the number of false positives and improve the efficiency of current proposals relying on matching global - as well as local - 3D descriptors [10, 1, 2, 3, 11].

1.1. Objectness for images

Alexe et al. presented in [4] an *objectness* measure for color images in an attempt to evaluate the presence of an object without any specific object class knowledge. They presented several image cues aimed at capturing the closed boundaries of objects, saliency as well as color contrast that are finally combined into a single objectness measure, to be computed on a (2D) rectangular bounding box. The objectness measure is evaluated at different randomly sampled locations and proved to be useful in speeding up specific object class detectors.

In our case, the availability of 3D information provides powerful cues to reason about objects directly on the same 3D domain where the actual objects reside. This allows computing tight volumes enclosing the objects as well as the ability to reason about free and occupied space. Additionally, normals computed on the surface of the objects provide a powerful cue to assess surface continuity and smoothness. Throughout the work, we will show how this additional tools prove useful to evaluate the presence of an object within a closed region in 3D space.

2. Notation and preliminaries

For a scene of interest, let $\mathcal{M}$ represent the depth map acquired from the RGB-D sensor and $\mathcal{S}$, in sensor coordinates, the 3D point cloud, as seen from a certain viewpoint $\vec{vp}$, reconstructed from $\mathcal{M}$. We assume that $\mathcal{S}$ contains a dominant planar surface $\mathcal{P} = \{\vec{n}, d\}$, $\vec{n}$ being the normal to the plane, d being the distance to the global reference frame, on which objects lie upon. Using $\mathcal{P}$, we apply a rotation and translation to the global reference frame of $\mathcal{S}$ so that its z-axis is aligned with $\vec{n}$ and its origin is on the plane ($d = 0$). We are now able to compute the *Volume of Interest* (VoI), a region in the 3D space containing all objects of interest, by checking the maximum extensions of the points above the table. The VoI defines as well the region where the bounding boxes will be sampled and the objectness measure evaluated.

2.1. Complexity of the parameter space

Differently from [4], in a 3D domain each bounding box b is characterized by 9 degrees of freedom: $b = b(x, y, z, s_x, s_y, s_z, r_x, r_y, r_z)$, where (x, y, z) represents the reference corner of a bounding box, (s_x, s_y, s_z) its extension along the positive direction of the 3 axes and (r_x, r_y, r_z) its orientation. To reduce the complexity of the parameter space, we model only r_z (rotations about $\vec{n}$), assuming $r_x = r_y = 0$, this being motivated by the fact that most objects lying on a table are well contained by a bounding box with one plane parallel to the dominant planar surface $\mathcal{P}$. Additionally, since objects lie on $\mathcal{P}$, it is possible to set $z = 0$, this resulting in 6 dimensions to be sampled and yielding $b = b(x, y, s_x, s_y, s_z, r_z)$ (the dependency of b from its independent variables will be dropped hereinafter for conciseness of notation).

Even after reducing the complexity of the problem, the number of bounding boxes that need to be evaluated remains high. However, thanks to *Integral Volumes* (IV) [5] it is possible to evaluate in constant time sums of elements (points, edges, etc.) contained in the volume of space within (x, y, z) and $(x + s_x, y + s_y, z + s_z)$. To model r_z, it is possible to rotate $\mathcal{S}$ at different angles ($0^\circ \leq r_z < 90^\circ$) and compute additional IVs for each r_z. Modeling r_z allows to obtain bounding boxes that enclose the object tightly. Next section discusses the different IVs that need to be computed to represent the cues required for the 3D objectness measure. Where not differently stated, the IVs are computed at a resolution of 1 cm.

2.2. Sampling bounding boxes

To cover the VoI, we perform an exhaustive sampling of the parameter space generating a bounding box b at each possible location. Corners defined by triplets (x,y,$z = 0$) are sampled every 2 cm along x and y directions; the bounding box extension defined by triplets (s_x, s_y, s_z) are sampled respectively every (2,2,1) cm; finally, r_z - the rotation angle about z - is sampled every 5°. We include a prior on the minimum and maximum size of objects, thus restricting s_x, s_y, s_z to be within the range $[3; 45]$cm. Such parameterization typically results in about 600 million bounding boxes to be evaluated for each scene acquired by the sensor. Thanks to the IVs and the parallel computing capabilities of GPU devices, it is possible to evaluate such amount of bounding boxes in a reasonable amount of time (approximately 3s on a GTX460M device with a straightforward implementation).

2.3. Occlusion and occupancy volumes

Previous to the definition of the 3D objectness measure, we need to introduce the concepts of occlusion volume and occupancy volumes. These are binary volumetric representations with an extension equal to that of the VoI and will allow us, later on, to derive important cues such as the free space inside a bounding box.

The occupancy volume – $\mathcal{V}$, is simply a binary representation of $\mathcal{S}_p$ where a voxel takes the value of 1 when at least a point $p \in \mathcal{S}_p$ falls inside the voxel boundaries, 0 otherwise. The occlusion volume – $\mathcal{V}_\mathcal{O}$, is likewise a binary set of voxels encoding whether a voxel is visible from the viewpoint $\vec{vp}$ or not, respectively taking values 0 and 1. To build $\mathcal{V}_\mathcal{O}$ we make use of the depth map $\mathcal{M}$ and the VoI previously computed. Concretely, we build a dense point cloud $\mathcal{C}_\mathcal{O}$ spanning the VoI with a resolution of 1cm. Afterwards, we backproject each point $p_i \in \mathcal{C}_\mathcal{O}$ to $\mathcal{M}$ using the calibration parameters of the sensor and reject all p_i with a depth value lower than the corresponding depth value in $\mathcal{M}$. The last step removes all visible points in $\mathcal{C}_\mathcal{O}$ and allows us to generate $\mathcal{V}_\mathcal{O}$ by simply checking if the voxel is empty or not. The middle part of Figure 2 shows an occlusion volume for a specific scene.

3. 3D Objectness

This section presents the cues as well as how they are combined together to define the 3D objectness measure. Similar to [4], the goal behind such cues is to capture the closed boundaries of objects in order to obtain high values for bounding boxes that contain an object entirely and that enclose it tightly.

3.1. Edge density

The first cue under consideration regards edges. Differently from [4], by reasoning in the 3D space it is possible to define a much richer set of edges than on the image plane. Specifically, we have deployed an edge extraction algorithm, available on the Point Cloud Library (PCL) [1], which is able to extract and discriminate between edges derived by surface curvature variations (blue), edges causing occlusions (orange), edges caused by occlusions (green) as well as scene border edges (red); reported colors refer to the left image of Figure 2, where a sample scene is depicted together with the extracted edges. The whole set of edges associated to a bounding box b will be referred to as $\varepsilon(b)$.

A nice property of such edges is that they are usually found on the surface of the objects of interest. Therefore, when a bounding box encloses a high number of edges compared to the area of its visible faces, this intuitively represents a strong cue for the presence of an object inside it. Observe that from a certain viewpoint, there will be always at most 3 faces of a bounding box that are visible. We thus define the edge density cue δ_i for a bounding box b as follows:

$$\delta_i(b) = \frac{|\varepsilon(b)|}{a(b)} \tag{1}$$

where $|\cdot|$ is the cardinality operator and $a(b)$ is the visible area of b.

3.2. Outer edges

A second cue derived as well from edges is aimed at penalizing bounding boxes that contain edges in their immediate surroundings. The neighborhood of a bounding box b is defined by a bigger bounding box $b_\epsilon = b(x, y, s_x + s_{x,\epsilon}, s_y + s_{y,\epsilon}, s_z + s_{z,\epsilon}, rz)$ (we set $s_{x,\epsilon} = s_{y,\epsilon} = 2$cm and $s_{z,\epsilon} = 4$cm). This cue is represented by the ratio between the number of edges inside b, and the number of edges inside the expanded bounding box b_ϵ:

$$\delta_o(b) = \frac{|\varepsilon(b)|}{|\varepsilon(b_\epsilon)|} \tag{2}$$

[1] www.pointclouds.org/blog/gsoc12/cchoi/index.php

Figure 2: From left to right: different types of edges computed on the point cloud, occlusion volume and smooth superpixels. Within the smooth superpixels image, points sharing the same color indicates belonging to the same superpixel. Points with high curvature (in red) are ignored during the oversegmentation stage and do not get assigned to any superpixel.

It is worth noting that this term is close to 1 when the surroundings do not contain edges and decreases linearly to 0 otherwise.

3.3. Smooth superpixels straddling

A third cue on which our objectness term relies aims at penalizing bounding boxes that intersect smooth surface segments (*superpixels*), as this is usually the indication that the bounding box does not contain entirely one object. Indeed, since one common assumption is that superpixels do not to straddle different objects, bounding boxes not containing a complete superpixel are penalized accordingly. The smooth segments are obtained performing an oversegmentation of $\mathcal{S}$ based on point proximity and surface curvature smoothness. The right part of Figure 2 shows the results of such over-segmentation stage. Observe how non smooth regions are all assigned the same label 0 (depicted in red).

Let $|p(b)|$ be the number of points inside a bounding box b with a superpixel label different than 0, $|p(s)|$ the number of points in $\mathcal{S}$ assigned to a superpixel s and $|p(b \cap s)|$ the number of points belonging to s and within b; the third cue $\delta_l(b, s)$ relative to a single superpixel s is then defined as follows:

$$\delta_l(b, s) = \frac{|p(b \cap s)|^2}{|p(s)|} \tag{3}$$

The final term $\delta_l(b)$ relative to a bounding box b and all its enclosed superpixels can then be obtained as follows:

$$\delta_l(b) = \frac{\sum_{s \in \Omega} \delta_l(b, s)}{|p(b)|} \tag{4}$$

where Ω is the set of superpixels extracted from $\mathcal{S}$.

3.4. Free space

A final cue taken under consideration regards the free space within a bounding box aim at favoring bounding boxes that tightly enclose the object of interest. Let $|p_{\mathcal{V}_\mathcal{O}}(b)|$ be the number of occluded voxels inside a bounding box b computed by means of the occluded volume $\mathcal{V}_\mathcal{O}$, and let $V(b)$ be the volume of b; the free space cue is then defined as follows:

$$\delta_f(b) = \frac{V(b) - |p(b)| - |p_{\mathcal{V}_\mathcal{O}}(b)|}{a(b)} \tag{5}$$

3.5. 3D Objectness measure

Given the aforementioned cues, we can thus define the objectness measure for a bounding box b by weighted sum of the previously introduced cues:

$$\delta(b) = w_i \cdot \delta_i(b) + w_o \cdot \delta_o(b) \cdot \delta_l(b) + w_f \cdot \delta_f(b) \tag{6}$$

As it can be seen, eq. 6 includes also a feature combination aimed at dimensionality reduction of the weights ($\delta_o(b)$ being multiplied by $\delta_l(b)$), this being motivated empirically. Instead of a heuristic measure like the one in eq. 6, a learning approach for the different weights and combination of cues is desired. We leave a more grounded approach outside of the scope of the paper and will address it in future work.

4. Point cloud segmentation

This section details how the objectness measure presented in the previous section can be successfully applied for the task of point cloud segmentation. We

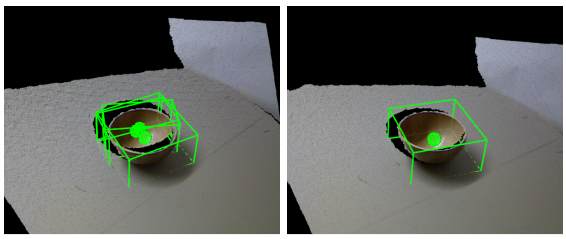


Figure 3: *Left*: Remaining bounding boxes after the post-processing stage in SubSection 4.1. *Right*: the final bounding box selected by the method presented in Section 4.2 that simultaneously considers the interaction between all hypotheses to find out a consistent segmentation of the scene.

carry out this by means of three successive steps: (i) we use the measure itself to filter out bounding boxes below a certain *object* threshold (ii) the remaining bounding boxes are clustered together based on the amount of scene overlap within them, then for each cluster only those with the highest objectness measure are kept, finally (iii) a global cost function is defined over the scene aimed at determining a globally consistent subset of bounding boxes that best segment the scene. The output of the algorithm is thus represented by the set of bounding boxes surviving these three steps or, equivalently, the scene segment associated to each bounding box, where a segment includes all pixels falling within a bounding box.

4.1. Filtering bounding boxes based on objectness

Consider $\mathcal{B} = \{b_1, \cdots, b_N\}$ a set of bounding boxes with an associated objectness score. As a first step, bounding boxes are discarded by thresholding the objectness score. Successively, we group the remaining bounding boxes into a set of clusters $\mathcal{C} = \{c_1, ..., c_m\}$, where each cluster c_i groups together bounding boxes based on the proximity of their centers' coordinates. Within each cluster c_i we analyze the first n_b, sorted by their objectness score, looking for conflicting bounding boxes. We say that two bounding boxes b_i, b_j are in conflict if they share at least 95% of scene points within them (with respect to the bounding box with a higher amount of points). We create a *conflict graph* within each cluster and perform a non-maxima supression based on the objectness measure to keep the best bounding box among those in conflict. This process results in a new set of bounding boxes $\mathcal{B}^* = \{b_1, \cdots, b_n\}$, with usually $n << N$. The left part of Figure 3 shows the bounding boxes after this stage while, on the right, it depicts the finally selected bounding boxes by means of the final post-processing stage, presented in the next section.

4.2. Global hypotheses selection

Here we provide a framework for establishing the most plausible configuration of salient objects in the current scene under evaluation. The problem can be formalized as follows. We start from a set of n object hypotheses, $\mathcal{B}^* = \{b_1, \cdots, b_n\}$, represented by the bounding boxes that survived the filtering step described in Section 4.1.

We adopt the framework (and notation) proposed in [3] to optimize the problem of finding the best configuration of plausible hypotheses simultaneously present in $\mathcal{S}$. Specifically, a cost function is defined over the solution space defined by the set of boolean variables $\mathcal{X} = \{x_0, \cdots, x_n\}$ having the same cardinality as $\mathcal{B}^*$, with each $x_i \in \mathbb{B} = \{0, 1\}$ indicating whether the corresponding hypothesis $h_i \in \mathcal{B}^*$ is dismissed/validated (i.e. $x_i = 0/1$). Hence, the problem can be formulated as finding the best configuration that minimizes a *cost* function expressed as $\mathfrak{F}(\mathcal{X}) : \mathbb{B}^n \rightarrow \mathbb{R}$, $\mathbb{B}^n$ being the solution space, of cardinality 2^n:

$$\tilde{\mathcal{X}} = \underset{\mathcal{X} \in \mathbb{B}^n}{\operatorname{argmin}} \{ \mathfrak{F}(\mathcal{X}) \} \tag{7}$$

where

$$\mathfrak{F}(\mathcal{X}) = \sum_{i=1}^{n} \delta_f(b_i) \cdot x_i + \lambda \sum_{p \in \mathcal{S}'} \omega_{\mathcal{X}}(p) \tag{8}$$

As it can be seen from (8), the cost function we aim at minimizing is composed by two terms weighted by a regularizing parameter λ. The left-hand term aims at enforcing tight bounding boxes around the objects, and thus penalizes the free space (through term $\delta_f(b_i)$) of the currently activated bounding box hypotheses within configuration $\mathcal{X}$. As for the right-hand term, it is a sum over all points of the scene surface $\mathcal{S}'$: for each point, a weight $\omega_{\mathcal{X}}(p)$ is thereby associated which enforces instead several cues penalizing invalid combinations of active bounding boxes over p in the current configuration $\mathcal{X}$. $\mathcal{S}'$ represents the initial scene $\mathcal{S}$ downsampled to a lower resolution (for efficiency reasons) after removing points on $\mathcal{P}$ or below it.

To define this weight, we first have to introduce a new term, $\kappa(p, b)$, which takes the value 1 when the point p is within the bounding box b, 0 otherwise.

On top of the definition of $\kappa(p, b)$, we define a term $\kappa(p)_{\mathcal{X}}$ counting the number of bounding boxes activated within a specific configuration $\mathcal{X}$ that enclose point p:

$$\kappa_{\mathcal{X}}(p) = \sum_{i=1}^{n} \kappa(p, b_i) \cdot x_i \tag{9}$$

The weight $\omega_{\mathcal{X}}(p)$ can be thus defined as:

$$\omega_{\mathcal{X}}(p) = \begin{cases} \sum_{i=1}^{n} \kappa(p, b_i) \cdot x_i, & \kappa_{\mathcal{X}}(p) > 1 \\ -\sum_{i=1}^{n} \kappa(p, b_i) \cdot x_i \cdot \delta(b_i), & \kappa_{\mathcal{X}}(p) = 1 \\ \sum_{i=1}^{n} \kappa(p, b_{i,\epsilon}) \cdot x_i, & \kappa_{\mathcal{X}}(p) = 0 \end{cases} \tag{10}$$

The three conditions included in (10) are relative to three different cues being simultaneously enforced by the proposed cost term. In the first condition (case i), $\kappa_{\mathcal{X}}(p) > 1$), point p introduces a penalty due to the fact that it being enclosed by more than one bounding box (*multiple assignment*). The penalty is proportional to the number of bounding boxes enclosing p. As for the second condition (case ii), $\kappa_{\mathcal{X}}(p) = 1$), the cost is being penalized by the objectness measure associated to the unique bounding box that encloses p, as we aim at retaining hypotheses characterized by high objectness. Finally, as for the third condition (case iii), $\kappa_{\mathcal{X}}(p) = 0$), if a point is not enclosed by any bounding box, it adds a penalty to all active hypotheses for which it falls in their proximity. This final cue is computed by means of an expanded bounding box b_ϵ as done in (2), and tends to penalize a bounding box if it has points lying in its surroundings that are not explained by any other active hypotheses, this being usually a sign of a not good enclosure of the object.

To find a minimum for the cost function $\mathfrak{F}$ we deploy Simulated Annealing[7], a typical metaheuristic algorithm used for finding approximated solutions of non-linear pseudo-boolean programming problems.

5. Experimental evaluation

In order to assess the performance of the 3D objectness measure as well as of the proposed segmentation approach presented in Section 4, we have performed an evaluation regarding segmentation accuracy on the publicly available Willow ICRA Challenge dataset [2] containing 434 object instances lying on a table. The dataset contains pixelwise annotated ground-truth segmentation and allows us to evaluate over- and under-segmentation. It contains typical household objects such as cereal boxes, food cans, detergent bottles, books, etc. (see Figure 4-(d)). Figure 1 show some scenes from the dataset.

We have compared the performance of our method with the segmentation method based on [6]; a simple but highly efficient two step strategy: (i) multi-plane segmentation of the scene and (ii) connected component clustering of points above any detected plane. To efficiently compute planar regions in a scene, it uses a connected components strategy where neighboring pixels are considered to be in the same component (planar region in this case) if the dot product of their normals and the Euclidean distance between the points are within a certain range. The planar regions found are further analyzed in order to merge regions that share the same planar model and were not detected during the first stage due to the constrained 4-neighbor search. The second step performs similarly to the first step, and groups points (without taking into consideration the points belonging to the detected planes) in the same component if their Euclidean distance is smaller than τ. The resulting components form the segmentation hypotheses. Such a segmentation strategy assumes that the objects will lie on a planar surface and that points belonging to different objects lie farther than τ. Hereinafter, we will refer to this method as *MPS*.

Additionally, we carried out an experiment to evaluate solely the objectness measure. To do so, we computed, on the same dataset, the Precision and Recall values for the bounding box with highest objectness score. In this case, we are interested in assessing how often the bounding box with highest score completely contains a ground truth object without including other objects or part of the background.

5.1. Results and discussion

Figure 4-(a) and -(b) compare respectively Precision and Recall results yielded by *MPS* and the proposed method for the task of scene segmentation. Each point in the scatter plot represents one scene in the dataset. The Precision and Recall values are computed for each scene by averaging the

[2] The whole dataset with annotated segmentation labels can be downloaded from `http://svn.pointclouds.org/data/ICRA_willow_challenge_segmentation_gt/`

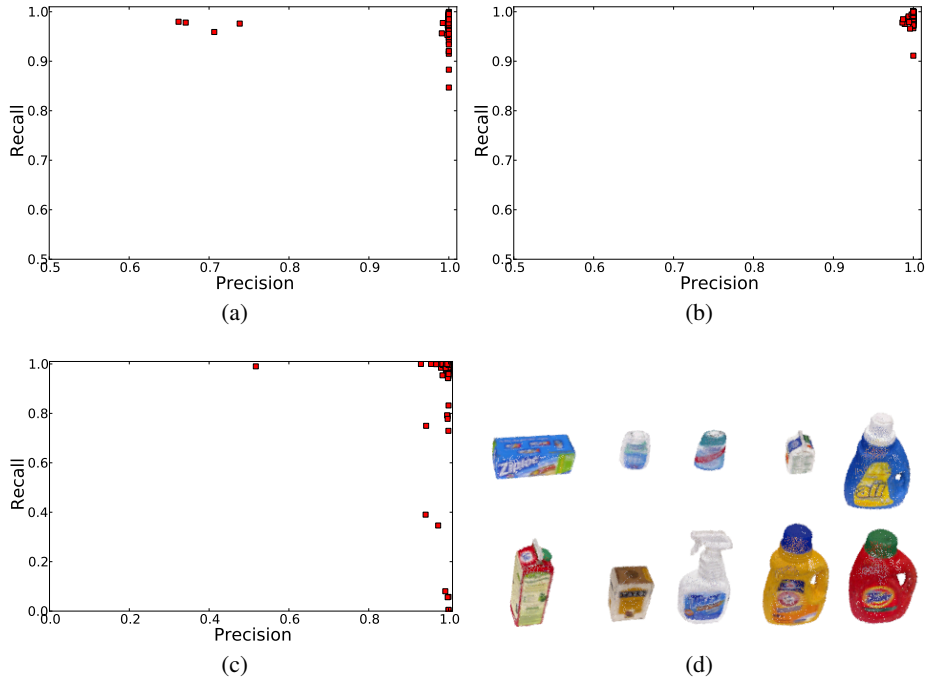


Figure 4: Evaluation on the Willow ICRA Challenge dataset (Precision vs Recall per scene) (a) segmentation results with *MPS* and (b) results with the proposed pipeline. (c) Precision and recall values for the bounding box with highest objectness score. (d) 10 of the 35 objects in the dataset.

respective values relative to each ground-truth object present in the scene. A low Precision value indicates that the object was undersegmented, while a low Recall indicates that the object was oversegmented. Observe how *MPS* presents undersegmentation on 4 scenes were objects are touching each other as well as oversegmentation on several other scenes caused by self-occlusions or missing data. Overall, the proposed approach outperforms *MPS*, with an average Precision/Recall of 99.9% versus 99.1%. Due to the dataset characteristics (scene objects often lie far away from each other), *MPS* performs, on the average, similarly to the proposed method. However, in some scenes where objects are quite close to each other, *MPS* tends to yield to undersegmentation (note the 4 sample points with Precision values lower than 0.8 in Figure 4-(a) indicating that two or more objects were merged in the same cluster), as well as minor oversegmentation artifacts (several points with Recall values lower than 0.95). Observe as well, on the left part of Figure 5, a scene where the presented method correctly splits two objects touching each other; *MPS* would fail to properly segment those objects due to their proximity ending up in a single cluster.

Figure 4-(c) shows the Precision and Recall results obtained by considering only the bounding box with highest objectness score. We can observe how just a single bounding box with high objectness resulted in undersegmentation of one scene. On the other hand, the best bounding box is relatively often presenting oversegmentation, yielding Recall values below 0.9. The scatter points with very low Recall values ($< 10\%$) appear when the best bounding box encloses only background (in some scenes, the hand of the person setting up the dataset is visible and within the VoI but annotated as background).

By analyzing Figure 4-(b) and -(c) simultaneously we can note that even on situations where the bounding boxes with higher objectness were causing oversegmentation (Recall values below 0.9), the segmentation method in Section 4 was ultimately selecting other bounding boxes providing a more pleasant and consistent configuration. The same applying for the undersegmentation case, indicates that simultaneously analyzing nearby bounding boxes allows to overcome some errors caused by individual local decisions.

6. Conclusion and future work

We have presented several cues to be computed on 3D point clouds aimed at evaluating how likely it is

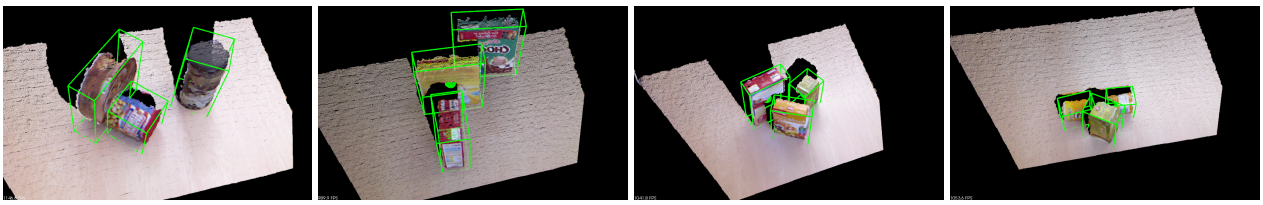


Figure 5: Some qualitative results with the presented method. Observe on the rightmost image in the figure a failure case where the cylinder is split into two separate bounding boxes. These scenes belong to the *Object Segmentation Database* `http://www.acin.tuwien.ac.at/?id=289`.

for a closed region in 3D space to enclose completely a single object. The cues have been combined in a preliminar *objectness* measure formulation that has shown great potential during the experiments. We have also presented a framework for scene segmentation based on the *objectness* measure as well as other physical constraints being able to find a plausible segmentation of table-top scenarios even under challenging clutter situations, represented by objects touching each other.

Based on the encouraging results obtained in this initial work as well as the observed limitations of the methods, there exist several directions that ought to be explored in the future. As already pointed out in Section 3.5, a more grounded approach for the combination of cues needs to be investigated as well as additional cues that might help solving even more challenging scenes. Another direction of research aims at reducing the computational complexity of the method in order to be able to explore the additional degrees of freedom that were ignored in the scope of this work (especially, we would like to allow objects to be on top of each other, so as to remove the table-top constraint). In this direction, we would like to explore bottom-up strategies to infer promising subspaces where the objectness measure will be evaluated. This strategy would allow to replace the current exhaustive enumeration resulting in a much lower complexity even when removing some or all of the constraints currently used.

Acknowledgements

The research leading to these results has received funding from the Doctoral College on Computational Perception at TU Vienna and the Austrian Science Foundation under grant agreement No. I513-N23.

References

[1] A. Aldoma, N. Blodow, D. Gossow, S. Gedikli, R. B. Rusu, M. Vincze, and G. Bradski. CAD-Model Recognition and 6DOF Pose Estimation Using 3D Cues. In *Workshop: 3rd IEEE Workshop on 3D Representation and Recognition, ICCV*, 2011. 1, 2

[2] A. Aldoma, F. Tombari, R. Rusu, and M. Vincze. Our-cvfh: Oriented, unique and repeatable clustered viewpoint feature histogram for object recognition and 6dof pose estimation. In *Joint DAGM-OAGM Pattern Recognition Symposium*, 2012. 1, 2

[3] A. Aldoma, F. Tombari, L. D. Stefano, and M. Vincze. A global hypotheses verification method for 3d object recognition. In *Proc. European Conf. on Computer Vision*, 2012. 2, 5

[4] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *CVPR'10*, pages 73–80, 2010. 1, 2, 3

[5] K. G. Derpanis. Integral image-based representations. Technical report, 2007. 2

[6] D. Holz, A. J. B. Trevor, M. Dixon, S. Gedikli, and R. B. Rusu. Fast segmentation of rgb-d images for semantic scene understanding. In *ICRA 2012 Workshop on Semantic Perception and Mapping for Knowledge-enabled Service Robotics*. 1, 6

[7] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, (4598), 1983. 6

[8] A. K. Mishra, A. Shrivastava, and Y. Aloimonos. Segmenting "simple" objects using rgb-d. In *ICRA*, pages 4406–4413. IEEE, 2012. 1

[9] A. Richtsfeld, T. Mörwald, J. Prankl, M. Zillich, and M. Vincze. Segmentation of Unknown Objects in Indoor Environments. In *IROS*, 2012. 1

[10] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In *Proceedings of the 23rd IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, 10/2010 2010. 2

[11] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of Histograms for local surface description. In *Proc. 11th European Conference on Computer Vision (ECCV 10)*, 2010. 2

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Knowledge gap detection for interactive learning of categorical knowledge

Matjaž Majnik, Matej Kristan, and Danijel Skočaj
University of Ljubljana, Faculty of Computer and Information Science
Tržaška 25, 1000 Ljubljana, Slovenia
{matjaz.majnik,matej.kristan,danijel.skocaj}@fri.uni-lj.si

Abstract. *In interactive machine learning the process of labeling training instances and introducing them to the learner may be expensive in terms of human effort and time. In this paper we present different strategies for detecting gaps in the learner's knowledge and communicating these gaps to the teacher. These strategies are considered from the viewpoint of extrospective and introspective behavior of the learner – this new perspective is also the main contribution of our paper. The experimental results indicate that the analyzed strategies are successful in reducing the number of training instances required to reach the needed recognition rate. Such a facilitation may be an important step towards the broader use of interactive autonomous systems.*

1. Introduction

Cognitive systems are often characterised by their ability to learn, interact with the environment, and act autonomously. They are able to respond to requests of human users and other cognitive agents, and are also capable of taking the initiative and engaging in a dialogue with a human. Very importantly, they are able to learn from such interactions; they are able to acquire novel knowledge and update previously learned conceptual models in an incremental manner. They can passively receive the information they need in this incremental learning process. In this case they simply rely on the environment, or on a human tutor, for being provided with appropriate information for efficient learning. However, they can also take an active part in this incremental learning process and try to infer what kind of information is needed to make the learning more efficient. The latter learning approach is known as *active learning*.

Active learning requires that the system identifies learning opportunities. This in turn requires that it must be able to *detect gaps in its knowledge*, which may indicate good learning opportunities. Typically the knowledge gaps are detected in a particular modality; they are usually grounded in a particular representations. Subsequently, the knowledge gaps have to be, in some general form, *communicated* to the rest of the system and to other agents that can plan and execute actions necessary to fill these gaps. After the system obtains the required information, it can extend its current knowledge accordingly. A crucial requirement of a system that is supposed to self-extend is therefore a certain level of self-understanding that enables the detection and communication of gaps in its knowledge.

In our work we focus on this problem. We address the problem of knowledge gap detection in the context of the active learning paradigm and address specific active learning strategies from the viewpoint of extrospective and introspective robotic behavior. This new robotics-oriented view also represents the main contribution of our paper. Since our research has been concentrated around interactive continuous learning of conceptual knowledge in dialogue with a tutor, most of this paper has been written with this learning scenario in mind. However, the proposed solutions are general enough that they can be applied to other learning domains as well.

Our final goal is to develop active learning strategies which would successfully reduce the amount of learning data needed to transfer the categorical knowledge from the human teacher to the robotic learner. These strategies should be used to construct as autonomous and as domain-independent framework for dialogue-based learning as possible.

The paper is organised as follows. In Section 2 we first discuss the related work. In Section 3 we present an approach to knowledge gap detection. In Section 4 we then discuss different ways of knowledge

gap communication and propose four active learning strategies based on them in Section 5. Then we present the experimental results in Section 6 and conclude the paper with some final remarks in Section 7.

2. Related work

Active learning strategies proposed in the literature mainly address the problem of estimating classifiers using minimal amount of data. They are motivated by the fact that there are many situations in which large quantities of unlabeled data are relatively easily obtained, however, the cost of labeling each instance can be high. Two extensive survey papers on active learning literature are available providing a broad overview of the field [11, 13]. Furthermore, a survey has recently been published [7], studying and comparing utility metrics and learning strategies for selecting training instances in active learning.

In this work we focus our attention towards *interactive learning of categorical knowledge in dialogue with a teacher*, similarly as the authors in [5]. For such real-life situations, it is desirable that the learner detects good candidates for querying on the fly and updates its classifiers accordingly, while requiring minimal involvement of the teacher. Also other authors focus more on this *social aspect* of active learning. In [8] the authors present a learning strategy akin to our *LDieSel*. They have similarly combined principles from two active learning scenarios, query synthesis and pool-based sampling. However, in their case the motivation is in obtaining improved performance by eliminating specific disadvantages of each of the two scenarios, while in ours we essentially enable the communication between the robotic learner and the human teacher.

In [1] four learning modes are described, which distinguish in how frequently the learner communicates with the teacher to obtain information, i.e. how often the interaction takes place. Four passive learning strategies based on features of biological systems are implemented in [2], where the strategies differ in the way they update the informativeness of individual objects. In [3] active learning is discussed from the viewpoint of transparency of the learner's internal states, and how these states may be used to inform the teacher about the level of the learner's knowledge. The approach is implemented on a physical robot with socially expressive head and neck, and simple non-verbal gestures are employed to provide the teacher with the transparent insight into the underlying model uncertainties.

In [15] the difference between training instances selected by a human teacher and systematically collected training instances is investigated. Besides, a method for the learner to convey the information about its knowledge gap to the teacher is presented. [1, 15, 3] include experiments with one or more human teachers, whereas in [2] robot-to-robot interaction is employed with the intention of providing a controlled environment for systematically exploring of how the learner is influenced by different teacher behaviours.

The authors in [10] discuss an active learning system from the viewpoint of combining interactive social learning (with a human teacher) and autonomous, non-interactive, intrinsically motivated learning. On the other hand, a learning strategy in [4], combines interactive autonomous selection of training instances (in areas of the problem space where the learner can classify with sufficient certainty) and non-interactive teacher-driven selection (in not-well-explored areas). The underlying learning method is, similarly as our odKDE, based on Gaussian mixture models (GMMs). A system with the goal of autonomous exploration of new knowledge is presented in [12] and discussed from the viewpoint of intrinsic motivation systems for autonomous development of robotic learners. The authors are inspired by human development where intrinsic motivation plays an important role and which may be characterized as progressive, incremental, active and autonomous.

The learning strategies that we address in our work are related to several approaches presented in the above-mentioned papers. We have built on our work in [14] and further developed certain learning strategies. Additionally, the meta-learning framework has been generalized to work on high-dimensional data (i.e. comprising tens of attributes) and tested with two learning methods.

3. Knowledge gap detection

In the active learning cycle there are two very important tasks that the system has to complete in order to get novel information that would help it to improve its knowledge in an efficient way: *detection* (Section 3) and *communication* (Section 4) of knowledge gaps. It should be noted that in the active learning community the process of "knowledge gap detection" is recognized under the term "selection of

informative training instances".

3.1. Extrospection and introspection

The crucial step in an active learning cycle is the detection of ignorance. The system should first self-understand – it should understand what it does and what it does not know. By using its internal modal representations it should detect what information is missing. Generally, the knowledge gap detection can be tackled in two different ways; it can either be related to a particular situation or not. We therefore may distinguish between two types of knowledge gap detection: extrospective and introspective.

In the case of **extrospective** knowledge gap detection, this process is related to a particular situation, i.e., to a particular object in a scene or to some other training instances the learner can perceive. The learner tries to detect the lack of knowledge by observing and trying to recognize a number of existing objects that it might or it might not know. This is a typical pool-based active learning approach – the learner is given a number of unlabeled objects, and it has to select one (or several) of them for labeling. Ideally, it would select the instance that would help to improve its knowledge most (by providing instance's real label).

Another way of detecting knowledge gaps is through **introspection**. In this case the detection of knowledge gaps is completely self-driven and is not related to any particular situation or task. It is not triggered by any external problem; it is triggered by an inner motivational mechanism with the goal of detecting ignorance and proposing actions that would provide the information needed to extend the current knowledge. No sensorial inputs are used in this case; the detection of knowledge gaps is based solely on the current knowledge. Since there are no real instances the learner could estimate its knowledge on, the robot could try to hallucinate sensorial inputs (basically, sample over distributions of feature values it uses) and attempt to interpret these hallucinated situations. Failing to do that would indicate a knowledge gap. This type of knowledge gap detection is thus also based on the output of the classifier, which is built on the top of the models; the only difference is that the input is hallucinated and not perceived.

3.2. Measuring uncertainty

In knowledge gap detection the crucial task is to measure how good the knowledge is, i.e., how certain or uncertain is the classification of particular observation or hallucination. In the case of extrospective knowledge gap detection the learner tries to estimate the certainty of recognition of all available data. Similarly, in the case of introspective knowledge gap detection the learner tries to estimate the certainty of hallucinated instances. In both cases, it has then to select the deepest gap in its knowledge based on the estimated certainties.

Several certainty measures can be employed. As mentioned above, our knowledge gap detection mechanism requires that the underlying learning and recognition methods provide posterior probability over all categories, therefore the responses $p(M_i|\mathbf{z})$ of all k models M_i for every given observation $\mathbf{z}$. Our method for calculating the certainty by analysing the posteriors is presented below.

First, we determine two models with the highest response:

$$M_{maxap} = \arg\max_{M_i}\{p(M_i|\mathbf{z})\} \tag{1}$$

$$M_{maxap2} = \arg\max_{M_i, M_i \neq M_{maxap}}\{p(M_i|\mathbf{z})\} \tag{2}$$

Based on these responses we are able to look for two types of knowledge gaps. A low response of the best model M_{maxap} indicates that the particular region of the feature space is not well modeled. In this case the measure for certainty could be expressed simply as

$$C(\mathbf{z}) = p(M_{maxap}|\mathbf{z}) \; . \tag{3}$$

However, even if the response of the best model is not low, but is on the other hand similar to the response of the second best model, we can consider the particular region in the feature space as a knowledge gap. The reason for such a conclusion is that the models are ambiguous, and can not provide reliable classification. In this case we can express the certainty as

$$C(\mathbf{z}) = p(M_{maxap}|\mathbf{z})/p(M_{maxap2}|\mathbf{z}) \; , \tag{4}$$

which is very similar measure to the margin sampling, known from the active learning literature. In this literature, also the third certainty measure in uncertainty sampling is often used, which is based on the entropy:

$$C(\mathbf{z}) = \sum_{i=1}^{k} p(M_i|\mathbf{z}) \log(M_i|\mathbf{z}) \; . \tag{5}$$

Once the certainties of all samples are estimated, the deepest knowledge gap can be found by looking for the most uncertain sample:

$$\mathbf{z}^* = \arg\min_{\mathbf{z}} \{C(\mathbf{z})\} . \quad (6)$$

In this paper we do not commit to specific underlying learning methods that are actually used to train classifiers using training instances; we rather focus on a higher layer of the proposed learning framework. We try to be as agnostic with respect to the underlying learning method as possible. Any incremental learning method and the classifier that can return posterior probability over all possible classes can, in principle, be used.

3.3. Directed uncertainty sampling

In the case of introspective knowledge gap detection the learner has to sample the feature space to produce the hallucinated instances. This sampling cannot be random, especially in the case of high-dimensional feature spaces; it should be driven by the structure of the current knowledge and by the output of previous classifications.

We have designed the following Monte Carlo–like method to deal with possible high-dimensional feature spaces. In our method the learner executes the following four steps.

1. Take M random samples from the feature-space. These are the first collected samples.
2. Calculate the depth of knowledge gap for all collected samples (as described in Section 3.2).
3. Choose M samples from the set of collected samples; tend to choose samples with deep knowledge gaps. Around each of M samples take a new Gaussian sample from the feature space, calculate the depth of knowledge gap for the sample and add it to the set of collected samples.
4. Repeat the previous step N times or until convergence.

This simple algorithm does not guarantee to find the optimal solution, i.e., the global certainty minimum. But in this task finding the optimal solution is not really necessary; what we actually want is a good enough solution, which is always provided by the algorithm. In fact, this algorithm very often finds knowledge gaps that are very close to the optimal ones.

4. Knowledge gap communication

The second problem we address in this paper is the one of knowledge gap communication. How can the learner communicate the knowledge gap? How can it notify others (e.g., the teacher) about what kind of information is needed? Also in this case we can distinguish between the extrospective and the introspective case.

In **extrospective** knowledge gap communication, the learner refers to existing training instances. It therefore selects one of the available training instances from the pool of instances, or it labels a generated instance based on the label of one of the existing instances.

In the case of extrospective knowledge gap detection such extrospective communication of the detected gap is straightforward, since the gap is located in one of the instances that actually exist.

In the case of introspection, however, it is more difficult to communicate this gap, since the detection is not based on a particular situation the robot could refer to. If the learner is able of inverse mapping from the feature values to the action parameters (as opposed to the mapping from action parameters to feature values, which is a part of the regular feature extraction process), then we say that the knowledge gap communication is also **introspective**. These kinds of learning scenarios can take full advantage of introspective knowledge gap detection, which may lead to a very efficient selection of training instances. If this is not possible, the learner can communicate the knowledge gap by referring to instances which he has access to (from the pool of training instances). In this case, therefore, the knowledge gap communication is extrospective.

5. Active learning strategies

Based on the way the knowledge gap is detected and on the means of how the request for information is communicated to the teacher, we distinguish four different active learning strategies (as also listed in Table 1):

- *LDeeSel (extrospective-extrospective instance selection).* The learner has access to a pool of non-labeled training instances. The learner measures potential knowledge gaps for feature vectors of introduced instances. Subsequently, the learner asks the teacher for the label of one of the not yet chosen instances for which the

deepest knowledge gaps have been detected. In this variant the learner operates on the pool of training instances and does not sample the feature space for detecting possible more significant knowledge gaps.

- *LDieSel (introspective-extrospective instance selection).* The learner samples the feature space and tries to detect gaps in its knowledge independently of the teacher. When found, the learner looks for the most similar training instance from the pool of the instances and communicates it as a knowledge gap. This strategy should be used when the learner can not communicate the detected feature gap directly (i.e. it can not map the feature values into the input space).
- *LDieGen (introspective-extrospective instance generation).* The knowledge gap detection is performed in a similar way as in the previous case. However, the label of the feature representing the detected knowledge gap is then obtained by checking the label of the most similar training instance in the pool of instances. Note that in this case the new training instance consists of the feature vector of the knowledge gap and the label of the most similar instance. The advantage of this approach in comparison to *LDieSel* strategy is that it tends to choose *the most* informative training instances for learning and not just some approximation from the existing pool. The weakness of this strategy, however, is a risk that the nearest training instance does not necessarily belong to the same class as the new instance (i.e. the assigned label may be incorrect).
- *LDiiGen (introspective-introspective instance generation).* This is the ultimate strategy. When this one is possible, the learner introspectively communicates the knowledge gap to the teacher and asks for the label. The teacher replies with the class label of this exact, newly generated training instance. Obviously, in this case an inverse mapping from the feature space to the input space should be possible.

The principles of *LDeeSel*, *LDieSel*, and *LDiiGen* learning strategies are well known in the field of active learning. However, to the best of our knowledge, *LDieGen* provides a *novel approach for choosing informative training instances*, at least from the perspective of human-robot interaction.

Table 1. Four different active learning strategies.

strategy	detection	communication
LDeeSel	extrospective	extrospective
LDieSel	introspective	extrospective
LDieGen	introspective	extrospective
LDiiGen	introspective	introspective

When using the above presented active learning strategies, models built by the underlying learning method are not reliable at the beginning of the learning process. Instead of leaning on unreliable knowledge, a reasonable solution seems to be some level of randomness, i.e. choosing a random training instance with some probability.

6. Experimental results

In this section we present the results of the evaluation of different learning strategies, presented in the previous section. The evaluation has been performed on three data sets: spatial templates, colors, and the UCI Letter recognition data set.

For the underlying learning method we used the odKDE algorithm we have previously developed [9], and in Sec. 6.3 also the mcpIncSVM based on [6]. Both methods are able of incremental learning by updating the representations with one training instance at a time. The corresponding classifiers conveniently calculate posterior probability for all learned concepts, therefore enabling simple detection of knowledge gaps as described in Section 3.

At the beginning of the learning process we have first trained the learning method with a small batch of initial learning instances to initialise the models. Afterwards we have continued to add one instance at a time, according to different learning strategies. At every step we have evaluated the quality of obtained models by trying to classify all test instances. All experiments have been repeated several times and the results have been averaged across all runs.

We begin this section with the summary of some teacher-driven learning strategies since they will be used as a baseline in comparison with the proposed active learning strategies.

6.1. Teacher-driven learning strategies

In the evaluation we used three strategies for teacher-driven selection of training instances, similar to those presented in [14]:

- *TDseq.* The simplest way of presenting the learning instances is to present them in a se-

quential order, one by one: first all of the instances from the first class, then all of the instances from the second and so on. In such a setting, the learner passively receives training instances.

- *TDrnd.* The teacher is showing not yet chosen training instances to the learner in a random order. The learner, again, passively accepts offered training instances.
- *TDfdb.* In this strategy the feedback from the learner is taken into account. The teacher shows a number of exam instances to the learner and then provides as a training instance one of the instances that was not recognised correctly. Clearly such a strategy requires more effort from the teacher and the learner, however, it should lead to better results and more efficient learning in terms of the number of knowledge updates. However, all exam instances also have to be labeled. The labels are not required by the learner but are needed by the teacher himself, therefore in this strategy the number of labeled training instances required could be high.

6.2. Learning spatial templates

Initially we have tested performance of the studied strategies on the data set of spatial templates which had been described in detail in [14]. This set contains 2 attributes and 3 classes and is the only data set of the three in our paper which allows us to compare all seven strategies. The other two (as also the vast majority of real-life domains) do not provide us with possibility of mapping from (low-dimensional) feature space to (high-dimensional) space of learning instances, and the use of fully introspective LDgen is not possible there. Fig. 1 contains averaged results of the experiment on 121 training instances with the odKDE learner over 100 runs. The training images were presented to the learning framework one by one, as selected by different learning strategies. As we may observe, all four variants of the learner-driven active strategies achieve superior recognition rate in comparison to teacher-driven strategies. *LDiiGen* strategy demonstrates the best behavior rising to 99% recognition rate after only 40 training instances. While *LDieGen* progresses quite fast in the beginning it is the only strategy which does not reach perfect classification performance at the end of learning. The reason is in incorrect labeling of some training instances, due to the risk mentioned in Sec. 5.

One thing to note is also a very good performance

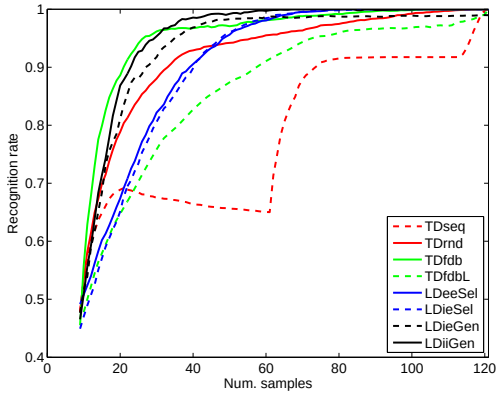


Figure 1. Recognition rate with respect to the number of training instances on the data set of spatial templates.

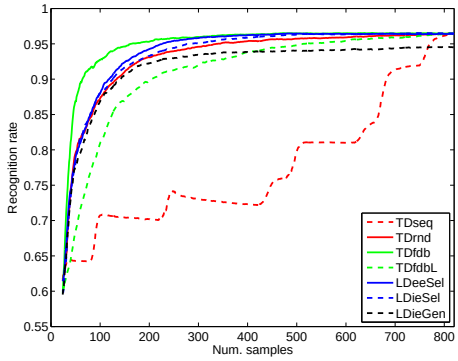
of the *TDfdb* strategy in terms of the number of instances used for updating the model (the solid green curve). However, the number of images that had to be labelled, in order to be presented as exam instances, was significantly higher (the dashed green *TDfdbL* curve).

6.3. Learning colors

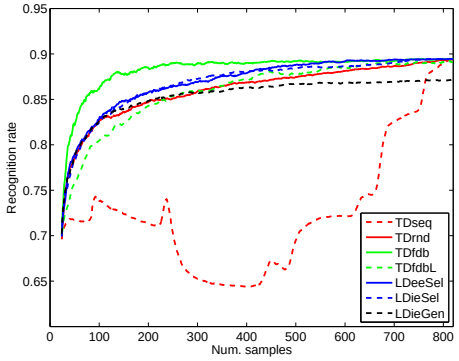
We have made further evaluation on the color data set from [14]. The set consists of 1094 images of 129 objects. We have used 820 images (75%) as training instances and 274 images (25%) for testing the recognition performance. Eight colours were being taught, based on the H, S, and L features from HSL values of the dominant colour. In each run the set of images has been randomly split into training and test sets. We evaluated the strategies using two underlying learning algorithms, odKDE and mcpIncSVM[1]. The mcpIncSVM was set to have linear kernel.

The experimental results are presented in Fig. 2 and show the evolution of the recognition rate with respect to the number of training instances. The results have been averaged over 100 runs. In our scenario not only the final recognition rate is important but also when certain recognition rate is reached. Although the difference between strategies is not significant in view of the final recognition rate, the advantage of our strategies is that *they achieve certain recognition rate (much) earlier than random (blind) strategies*. Fig.3 depicts the number of training instances required to achieve a certain level of recognition rate for all six strategies. This level has been set to 99% of the final result of the baseline

[1]The source code is available at http://www-ti.informatik.uni-tuebingen.de/ spueler/mcpIncSVM/.



(a) odKDE



(b) mcpIncSVM

Figure 2. Recognition rate in the domain of learning qualitative descriptions of object colours.

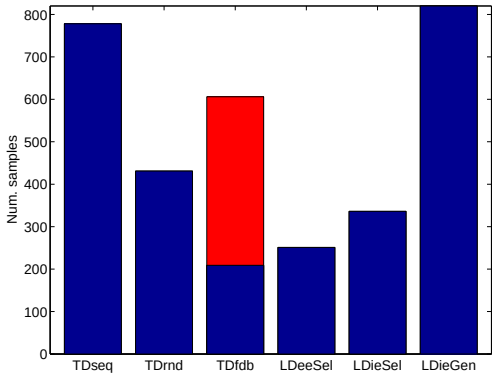


Figure 3. Number of training instances needed to achieve the same recognition rate in the case of learning colors.

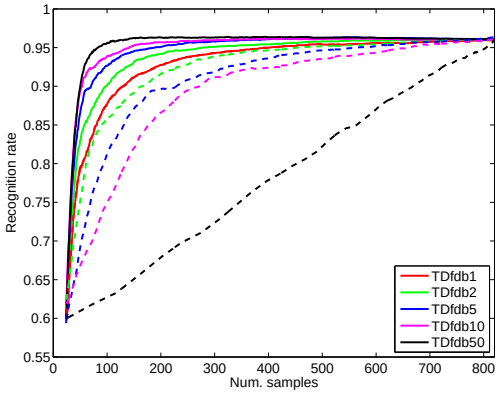


Figure 4. Recognition rate for different number of exam instances: 1, 2, 5, 10, and 50

TDrnd method. For the *TDfdb* strategy both results are shown; how many training instances have been used for updating the knowledge, as well as how many training instances have been labeled (when showing the exam instances). The figure shows that the *LDeeSel* and *LDieSel* active learning strategies clearly outperform the teacher-driven strategies.

In addition to these experiments, we have also verified the influence of the *number of exam instances* (N_{exams}) on performance of the *TDfdb* strategy. This number determines how many learning instances are used for intermediate evaluations of the learner's knowledge. When N_{exams} equals 1, *TDfdb* strategy operates identical to random sampling. As the value of N_{exams} increases, the probability that at least one instance will be incorrectly classified – meaning that we will be able to take such an instance as a new training instance – also grows. The results are shown in Fig. 4. The experiments have been carried out with the odKDE learning method on the color data set with 820 training instances. Introducing more exam instances to the learner obviously speeds up the learning process. However, it has to be noted that for *TDfdb* to achieve certain recognition rate there have to be (considerably) more labeled instances available as not only training instances but also exam instances have to be labeled (denoted as the dashed curves in Fig. 4). *This experiment nicely demonstrates that by an optimal selection of the training instances one can speeds up the learning process enormously.*

6.4. Experiment on the UCI Letter database

We have evaluated the strategies also on a considerably more extensive data set – UCI Letter recognition. This set is a high-dimensional data set with 16 attributes and 26 classes, where the data may not be fully visualized. There are 16000 training and 4000 test instances available in separate sets. In Fig. 5 results of testing the strategies on the UCI Letter

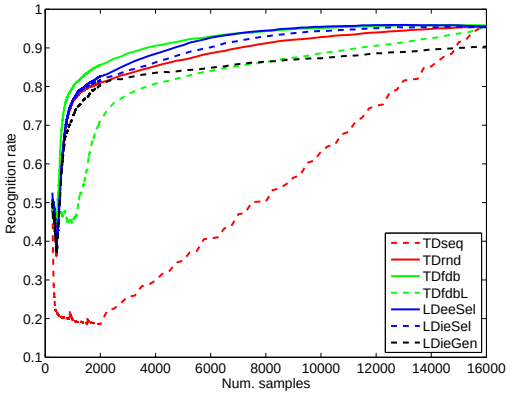


Figure 5. Recognition rate in the alphabet letters domain.

recognition data set using the odKDE learner are presented. The results have been averaged over 15 runs. It is important to notice that the results are analogue to the results on lower-dimensional color data set. These results, therefore, also confirm superior performance of the active learning strategies.

7. Conclusion

In this paper we addressed the problem of acquiring categorical knowledge from the active learning perspective. We focused on the problems of knowledge gap detection and communication. For both of these problems we proposed introspective and extrospective solutions, and based on these solutions, we presented four variants of active learning strategies.

The experimental results show that the analyzed active learning strategies effectively reduce the amount of training data that have to be introduced to the learner when reaching the required recognition rate. The results also demonstrate that the concerns about incorrect labelings in the proposed *LDieGen* strategy have came true. Further thorough test will be conducted on different domains, however, for the time being, this strategy can not be successfully employed to the problem of knowledge gap detection and communication.

References

[1] M. Cakmak, C. Chao, and A. L. Thomaz. Designing interactions for robot active learners. *IEEE T. Autonomous Mental Development*, 2(2):108–118, 2010. 2

[2] M. Cakmak, N. DePalma, R. I. Arriaga, and A. L. Thomaz. Exploiting social partners in robot learning. *Autonomous Robots*, 29(3-4):309–329, 2010. 2

[3] C. Chao, M. Cakmak, and A. L. Thomaz. Transparent active learning for robots. In *Proceedings of the 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 317–324, 2010. 2

[4] S. Chernova and M. Veloso. Confidence-based policy learning from demonstration using Gaussian mixture models. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, 2007. 2

[5] J. de Greeff, F. Delaunay, and B. T. Human-robot interaction in concept acquisition: a computational model. In *Proceedings of the 2009 IEEE 8th International Conference on Development and Learning*, pages 1–6, Washington, DC, USA, 2009. 2

[6] C. Diehl and G. Cauwenberghs. Svm incremental learning, adaptation and optimization. In *Proceedings of the 2003 International Joint Conference on Neural Networks*, page 26852690, 2003. 5

[7] Y. Fu, X. Zhu, and B. Li. A survey on instance selection for active learning. *Knowledge and Information Systems*, 2012. 2

[8] X. Hu, L. Wang, and B. Yuan. Querying representative points from a pool based on synthesized queries. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pages 1–6, 2012. 2

[9] M. Kristan and A. Leonardis. Online discriminative kernel density estimation. In *International Conference on Pattern Recognition*, pages 581–584, Istanbul, Turkey, 23-26 August 2010. 5

[10] S. M. Nguyen and P.-Y. Oudeyer. Interactive learning gives the tempo to an intrinsically motivated robot learner. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, 2012. 2

[11] F. Olsson. A literature survey of active machine learning in the context of natural language processing. Technical report SICS Technical report T2009:06, Swedish Institute of Computer Science, 2009. 2

[12] P.-Y. Oudeyer, F. Kaplan, and V. V. Hafner. Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation*, 11(2):265–286, 2007. 2

[13] B. Settles. Active learning literature survey. Technical report Computer sciences technical report 1648, University of Wisconsin-Madison, 2010. 2

[14] D. Skočaj, M. Majnik, M. Kristan, and A. Leonardis. Comparing different learning approaches in categorical knowledge acquisition. In *Proceedings of the 17th Computer Vision Winter Workshop*, pages 65–72, 2012. 2, 5, 6

[15] A. L. Thomaz and M. Cakmak. Learning about objects with human teachers. In *Proceedings of the 4th ACM/IEEE International Conference on Human-Robot Interaction*, pages 15–22, 2009. 2

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Top-down 3D Tracking and Pose Estimation of a Die Using Check-points

Fuensanta Torres and Walter G. Kropatsch
PRIP, Vienna University of Technology, Austria
http://www.prip.tuwien.ac.at

Abstract.

Real-time 3D pose estimation from monocular image sequences is a challenging research topic. Although current methods are able to recover 3D pose, they require a high computational cost to process high-resolution images in a video sequence at high frame-rates. To address that problem, we introduce the new concept of check-points. They are the minimum number of points needed to detect a 3D object motion. Our method tracks the 2D projections of the check-points over a 2D maximum pyramid. To handle large displacements of the object, our approach evaluates the projection of the check-points at highest levels of the pyramid. Moreover, it refines the pose localization by utilizing the check-points at lowest levels of the hierarchy. We show that just checking a few cells per frame, our method estimates the 3D pose of the tracked object. This early version of the method works with a specific type of object a 3D cube, with six well distinguished faces and which salient features in all the faces are dots, a die.

1. Introduction

Tracking and pose estimation of an object in a video sequence means continuously identifying all six degrees of freedom that define the object position and orientation relative to the camera. This is used in robotic applications, augmented reality systems, human computer interaction, automatic surveillance, etc.

After more than thirty years of research, real-time tracking for high-resolution images in an video sequence at high frame-rates is still a challenging research topic.

Here we focus on model-based 3D tracking using a single camera [11]. Model-based tracking methods use the prior knowledge of the shape and the appearance of the tracked object. The link between perception and the prior knowledge improves the robustness and performance of the method. We can divide the current tracking techniques into three main categories: the marker-based techniques, the natural feature-based approaches and the tracking-by-detection methods.

Tracking-by-detection methods identify and match points in successive images, in a non-recursive way [15], [1], [12]. These are strong methods. However, they analyze the whole frame to detect the features, which is computationally expensive.

The marker-based techniques and the natural feature-based methods match individual features across images, in a recursive way, which means that they use the last calculated position as an estimate for the current position. The marker-based techniques use either point fiducial or planar markers ([2], [8]). They are fast, robust and accurate [11]. Their main drawback is the difficulty to introduce these marks in all the environments.

The natural feature-based methods use the surface properties present in the nature. The natural features are either the edges (strong gradients or straight line segments) [20], [5] or the information provided by pixels inside the object (optical flow, template matching or interest point correspondences) [7], [18].

Inspired by [13] we propose a top-down tracking method. "A top-down method incorporates the prior knowledge about the objects in the lowlevel image processing" [13]. In particular, our algorithm uses the prior knowledge about the 3D shape in order to find the check-points. The check-points are the minimum set of 3D points around a salient feature, which detects and estimates the changes in the object movement (we assume a textured object with continuous and smooth motion). Once we have the check-points of the object, the tracking method requires a low computational cost.

Furthermore, our algorithm is based on a pyrami-

dal representation which allows large view changes. The use of a hierarchical approach for tracking has been widely used in the literature [4], [19], [9], [14]. Contrary to these methods, we implement a top-down feature extraction instead of a bottom-up process [10], which is less time consuming. Nevertheless, the main drawback of these approaches have been the high computational cost to build a pyramid per frame. To overcome this weakness, we can use the increasing programmability and computational power of the graphics processing unit (GPU) present in modern graphics hardware. It provides great scope for acceleration of computer vision algorithms which can be parallelized [16].

The rest of the paper is organized as follows: Sec.2 describes the different structures and processes of this approach. Sec. 3 presents the top-down 3D tracking and pose estimation method. The experimental results revealing the efficacy of the method are shown in Section 4. Finally, the paper concludes along with discussions and future work in Section 5.

2. Definitions

In this section we define the new concepts of maximum pyramid and check-points. Moreover, we describe in detail our recursive predictor-corrector tracking algorithm.

2.1. Maximum pyramid

A regular pyramid is a hierarchy (Fig. 1). Each level λ contains an array of cells. A cell of a regular pyramid is determined by its position (i, j, λ) in the hierarchy, (i, j) are its coordinates within the level λ. The cells on $\lambda = \emptyset$ (base level) are either directly the pixels of the input image or the result of any local computation, like filters, on the image. We obtain each pyramid level recursively by processing the level below.

The reduction window gives us the children-parent relationships. Each cell in level λ +1 has a reduction window of N×N children at level λ.

We can extend the parent-child relationship, defined by the reduction window, until the base level. The receptive field (RF) of a cell is the set of its linked pixels on the base level. The RF defines the embedding of a cell on the original image.

We use the reduction function to compute the value of each parent from the set of values of its children. The maximum pyramid uses the maximum as reduction function. Every cell stores the maximum gray value of its receptive field. And all the gray values in the receptive field of a cell are equal or smaller than the gray value of this one. The top of the pyramid receives the maximum gray value of the base image. The ratio between the number of cells at level λ and the number of cells at level λ+1 is the reduction factor (q).

The reduction factor and the reduction window (N×N/q) define a regular pyramid [3].

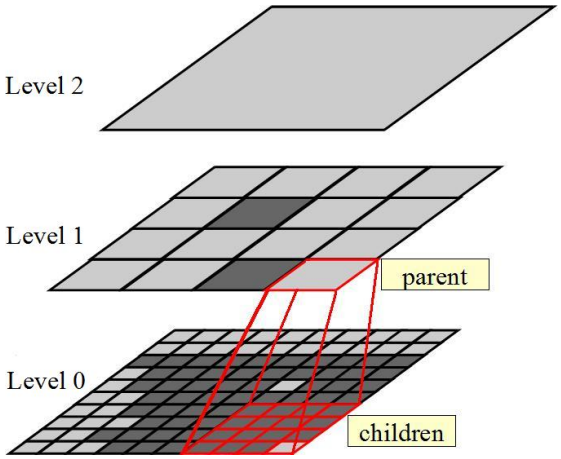


Figure 1. Example of a 4×4/2 maximum pyramid

Let D_λ be a simply connected non-maximum region (pixels with higher gray values surround the region) at level λ of an one-dimensional N/2 pyramid. $D_\lambda$can survive until the level $\lambda + 1$ if the receptive field of at least one cell at this level is completely inside of this region. We can construct $D_{\lambda+1}$ by erosion (morphological operation) of D_λ. The erosion applies a structuring element to D_λ. In this case the structuring element is the reduction window (W= $|N|$) (Erosion of D_λ by W is denoted $D_\lambda \ominus W$). Furthermore, a sub-sampling reduces the number of cells at $\lambda + 1$ from λ by the reduction factor 2. Sub-sampling with a factor of 2 corresponds to choosing every second cell (Sub-sampling X by a factor q is denoted X ↓ q):

$$D_{\lambda+1} = (D_\lambda \ominus W) \downarrow 2 \tag{1}$$

Using (1) and replacing all the factors by their sizes (d_λ is the size of D_λ):

$$d_{\lambda+1} = \frac{d_\lambda - N + 1}{2} \tag{2}$$

Theorem 1 (1D non-maximum region) *Let D_λ be a simply connected non-maximum region at level λ of*

an one-dimensional N/2 maximum pyramid. The size (d_λ) of D_λ, is exponentially decreasing by the reduction factor 2 as the level λ grows to higher pyramid levels:

$$d_\lambda = \frac{d_0 + (N-1)}{2^\lambda} - (N-1) \tag{3}$$

Proof: We prove the theorem by induction. The size of the non-maximum region at the base level is correct: $d_0 = \frac{d_0+(N-1)}{2^0} - (N-1) = d_0$. We assume that (3) is true for $\lambda = \alpha$. Using the recursion (2) we derive the formula for $\lambda + 1$:

$$\begin{aligned} d_{\lambda+1} &= \frac{d_\lambda - N + 1}{2} && (4)\\ &= \frac{(\frac{d_0+(N-1)}{2^\lambda} - (N-1)) - N + 1}{2} && (5)\\ &= \frac{d_0 + (N-1)}{2^{\lambda+1}} - \frac{N-1+N-1}{2} && (6)\\ &= \frac{d_0 + (N-1)}{2^{\lambda+1}} - (N-1) && (7) \end{aligned}$$

□

In a similar way, let B_λ be a connected maximum-region (pixels with lower gray values surround the region) at level λ of a N/2 pyramid. B_λ can survive until the level $\lambda+1$ if the receptive field of one cell at this level has at least one child inside of this region. We can construct $B_{\lambda+1}$ by the dilation of B_λ. The structuring element is the reduction window W. As mentioned before, a sub-sampling reduces the number of cells at $\lambda + 1$ from λ by the reduction factor 2:

$$B_{\lambda+1} = (B_\lambda \oplus W) \downarrow 2 \tag{8}$$

Using (8) and replacing all the factors by their sizes (b_λ is the size of B_λ):

$$b_{\lambda+1} = \frac{b_\lambda + N - 1}{2} \tag{9}$$

Theorem 2 (1D maximum region) *Let B_λ be a connected bright region at level λ of an one-dimensional N/2 maximum pyramid. The size (b_λ) of B_λ, is exponentially decreasing by the reduction factor 2 as the level λ grows to higher pyramid levels:*

$$b_\lambda = \frac{b_0 - (N-1)}{2^\lambda} + (N-1) \tag{10}$$

Proof: *Proof.* We proof the theorem by induction. First, the size of the maximum-region at the base level is correct: $b_0 = \frac{b_0-(N-1)}{2^0} + (N-1) = b_0$. We assume that (10) is true for $\lambda = \alpha$. Using the recursion (9) we derive the formula for $\alpha + 1$:

$$\begin{aligned} b_{\lambda+1} &= \frac{b_\lambda + N - 1}{2} && (11)\\ &= \frac{(\frac{b_0-(N-1)}{2^\lambda} + (N-1)) + N - 1}{2} && (12)\\ &= \frac{b_0 - (N-1)}{2^{\lambda+1}} + \frac{N-1+N-1}{2} && (13)\\ &= \frac{b_0 - (N-1)}{2^{\lambda+1}} + (N-1) && (14) \end{aligned}$$

□

Fig. 2 illustrates the appearance of a bright region (Theorem 2) at different levels of a 4/2 maximum pyramid. When we have a maximum-region of size b_λ=6, $b_{\lambda+1}$ will be either 5 or 4. $b_{\lambda+1}$ depends on the alignment of W in B_λ. If b_λ is odd, $b_{\lambda+1}$ does not depend of the alignment of W in B_λ because B_λ appears in the same number of reduction windows independently of the alignment. When b_λ=3, $b_{\lambda+1,+2,+3...}$=3. Furthermore, maxima-regions converge to size 3 cells being smaller or larger than 3.

We can extend the results of the theorems above to any dimension by the cross product. For instance, considering a 4×4/2 maximum pyramid and b_λ=6×6, $b_{\lambda+1}$ will be equal to 4×4, 4×5, 5×4 or 5×5.

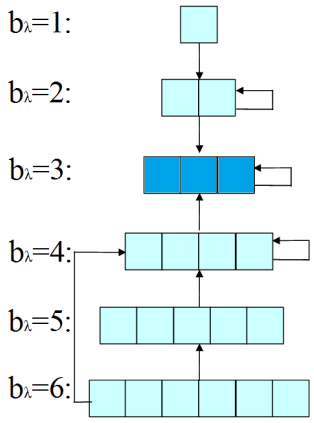


Figure 2. Appearance of the maximum-region at different levels of the 4/2 maximum pyramid. The direction and the sense of the arrows indicate the evolution of B_λ to $B_{\lambda+1}$

2.2. Check-points

Check-points are the minimum set of 3D points around a salient feature (the dots or the whole die),

which detects and estimates the changes in the object movement. We assume a textured object with continuous and smooth motion. We consider translation, rotation and uniform scaling transformations.

Four points around of one dot detect any possible translation and uniform scaling transformations. And one point inside alerts us when we lose the object (Fig. 3).

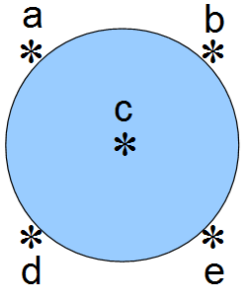


Figure 3. Check-points

However, we need at least two groups of check-points to detect a rotation change with a circular shape (Fig. 4).

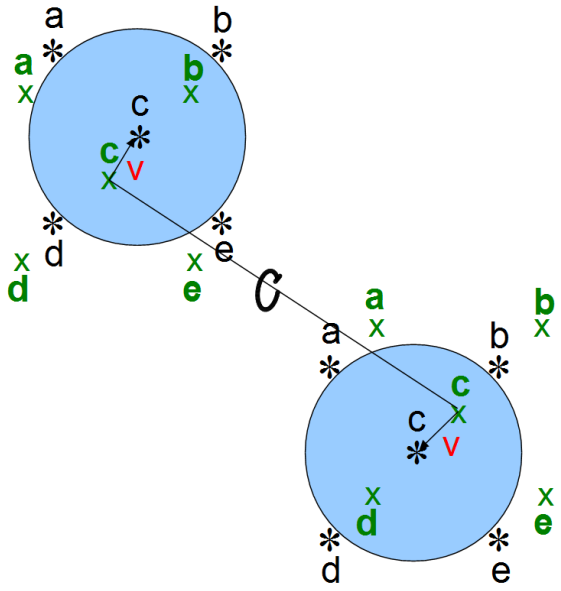


Figure 4. Predicted check-points (x) and the correct positions of these check-points (*)

2.3. Prediction-Estimation-Correction

This section defines the Prediction-Estimation-Correction method (PEC). We track the check-points with a prediction-correction method. The procedure is as follows. A motion model (the 3D affine motion model) predicts forward the check-points. We project the predicted check-points positions into the current frame. First, it checks whether **a**, **b**, **d** and **e** are outside of the saliency and **c** is inside. Otherwise, the prediction is incorrect. Tab. 1 considers all the possible errors and estimates their correction vectors. In the table, the zero means that the check-point is outsize of the saliency and the one appears when it is inside. The "-" means that this check-point does not have any effect on the calculated correction vector. The direction and the sense of the arrows describe the correction vector. For instance, the case of Fig. 5 corresponds to the box in Tab. 1, a, c, d are equal to 1 while b and e are equal to 0. Therefore, we translate the prediction to the left.

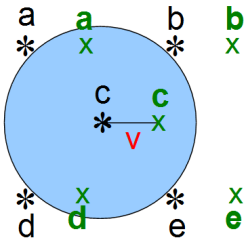


Figure 5. Translation error

Table 1

values at prediction					
a	b	c	d	e	v
0	0	–	0	1	↘
0	0	–	1	0	↙
0	0	–	1	1	↓
0	1	–	0	0	↗
0	1	–	0	1	→
0	1	–	1	1	↘
1	0	–	0	0	↖
1	0	–	1	0	←
1	0	–	1	1	↙
1	1	–	0	0	↑
1	1	–	0	1	↗
1	1	–	1	0	↖
0	0	1	0	0	s
1	1	1	1	1	$1/s$

Finally, the correction step calculates the relationship between the predicted and the estimated check-points's positions. This least-squares problem in the 3D space is solved by using Horn [6]. Horn returns the uniform scale factor (s), the rotation matrix (R_{3x3}) and the translation vector (T_{3x1}) needed to get the correction (x) from the prediction (x')(15).

$$x = (s \cdot R_{3\times3} + T_{3\times1}) \cdot x'; \tag{15}$$

3. Top-down tracking and Pose Estimation method

Our method tracks a die and estimates its 3D pose in a monocular video sequence. This is a model-based and top-down tracking method. We have a 3D model of the tracked die, this model is just the coordinates of its corners and the positions of its checkpoints in each face. Moreover, our approach uses the maximum pyramid to do a top-down feature extraction.

The method can be divided in three stages: The first step is the localization of the die. The second stage operates in the 2D space, it extracts the position. Finally, the third step operates in the 3D space, which estimates the 3D pose (Fig. 6).

Target localization: We use Theorem 2 to find the necessary height of the pyramid. If we know approximately the size of the die in the current frame, we can find the top-level where the whole die has a size at most of 6 cells. At this level the whole die has approximately homogeneous color. The method selects the cells where the object is placed. Hence the die is localized.

Object position: We go top-down along the pyramid until a level where the number of cells of the die is at least 20 (we use Theorem 2 to find this level). In this level, our approach estimates the position of the die with the PEC method and one group of checkpoints. Then we refine the position by going top-down in the hierarchy until the dots appear on the level (obtain this level with Theorem 3).

3D pose estimation: Finally, the method refines the 3D pose of the die by using two groups of checkpoints and the PEC method.

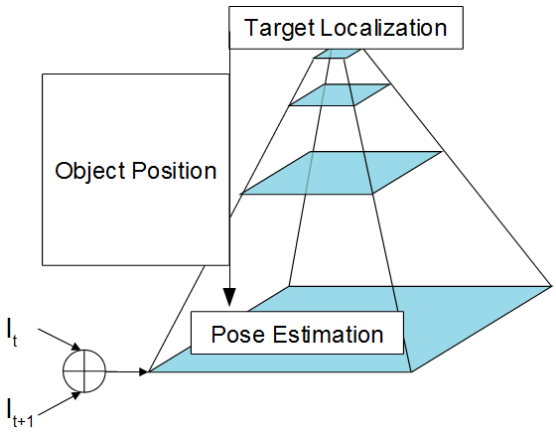


Figure 6. Illustration of the Top-down tracking and 3D pose estimation algorithm

4. Testing and results

All experiments are carried out with a 4×4/2 maximum pyramid.

Tab. 2 shows the experiments with two different images of a die: a die whose face area is 1600 pixels (40×40 pixels) (Fig. 7) and a higher resolution image with face area 96100 pixels (310 ×310 pixels).

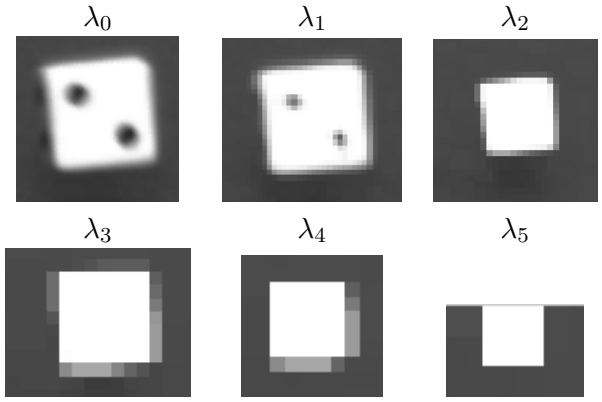


Fig. 7. Maximum pyramid of a die (face area 40×40)

Tab. 2 shows the side length of the die (σ) and the diameter of its dots (δ) at every level of the pyramid. ε is the error between the theoretical (10), (3) and the experimental results, $\varepsilon_{\sigma_\lambda}$=$|\sigma_\lambda - b_\lambda|$ and $\varepsilon_{\delta_\lambda}$ =$|\delta_\lambda - d_\lambda|$. The biggest error is $\varepsilon_{\sigma_\lambda}$=0.69 pixels and $\varepsilon_{\sigma_\lambda}$= 1.37 pixels in the lower and in the higher resolution image respectively.

Furthermore, the face size of the high resolution die decreases from 96100 pixels (310×310 pixels) at the base level to 24649 pixels (157×157 pixels) at the level 1 which is a 74.35%. The reduction of the die's size allows to use smaller correction vectors which accelerate the PEC method.

For the rest of our experiments we use a monocular video sequence of a die. Figs. 8 and 9 have the same nine frames (I_1, I_2..., I_9) of the video sequence. They show the prediction (green points) and the correction (red-points) of the check-points's positions at the levels 0 and 3 respectively of the maximum pyramid.

The diagram below (Fig. 10) shows the error between the obtained and the real y-coordinate of the center point of the die for 22 frames. The blue line is the real y-coordinate and the red points are the results of our method. The average error for 50 frames (until the die stops) is 0.58 pixels.

Finally, the strengths of our method is proven with different experiments:

- Robustness to illumination changes: We

Table 2. Maximum pyramid and the errors in pixels from the theoretical results

face area 40×40 pixels

Level λ	σ_λ	$\varepsilon_{\sigma_\lambda}$	δ_λ	$\varepsilon_{\delta_\lambda}$
0	40	0	8	0
1	21	0.5	3	0.5
2	12	0.25	0	0
3	8	0.37	0	0
4	6	0.69	0	0
5	4	0.16	0	0

high resolution image (310×310 pixels)

Level λ	σ_λ	$\varepsilon_{\sigma_\lambda}$	δ_λ	$\varepsilon_{\delta_\lambda}$
0	310	0	60	0
1	157	0.5	28	0.5
2	81	1.25	12	0.75
3	40	1.37	4	0.87
4	23	0.81	0	0
5	13	0.4	0	0
6	8	0.2	0	0
7	6	0.6	0	0

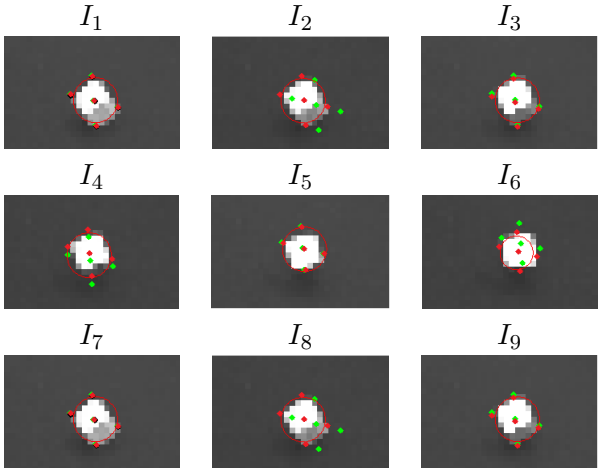


Fig. 8. PEC method at level 3 of the maximum pyramid (object position)

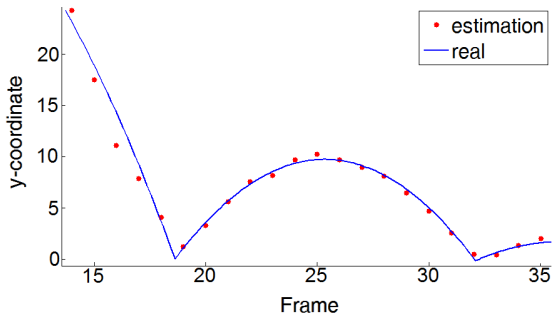


Figure 10. Center-point trajectory (y-coordinate)

changed the illumination in the training se-

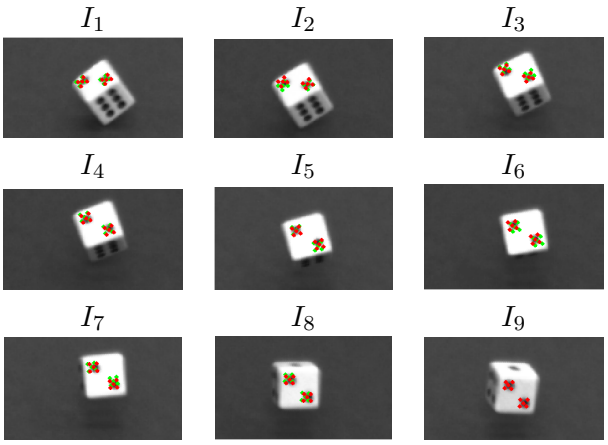


Fig. 9. PEC method at level 0 of the maximum pyramid (3D pose estimation)

quence Fig. 11. As can be seen in the bottom row the checkpoints handle a very abrupt lighting changes.

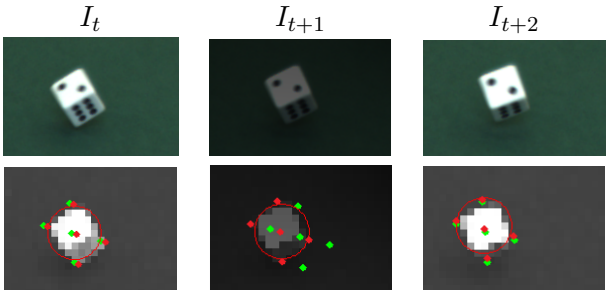


Fig. 11. Robustness to illumination changes

- Insensitivity to large view changes: The evaluation of the check-points at highest levels of the pyramid can handle large view changes. Moreover, it also updates the motion model. Fig. 12 shows in the top row the frames I_t, I_{t+12}, I_{t+13} and I_{t+14} of a video sequence. As can be seen in the bottom row, it localizes the die in the frame I_{t+12} and updates the motion model. Therefore, the prediction in the frame I_{t+14} is more accurate than in frame I_{t+13}.

- Computationally Cheap: The pyramid of the current frame I_{t+1} is the same pyramid as the previous frame I_t, where only the differences between I_{t+1} and I_t have been updated. Fig. 13 shows two consecutive frames and their differences on the top row, while the row below shows the differences at level 1 of the maximum pyramid. In this particular example, the dimensions

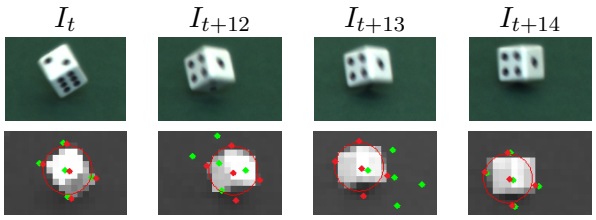


Fig. 12. Insensitivity to large view changes

of I_t and I_{t+1} are equal to 640x480= 307200 pixels, there are 5020 different cells at the base level 0, 17 at level 1, 10 at level 2 , 2 at level 3 and 0 in the rest of levels.

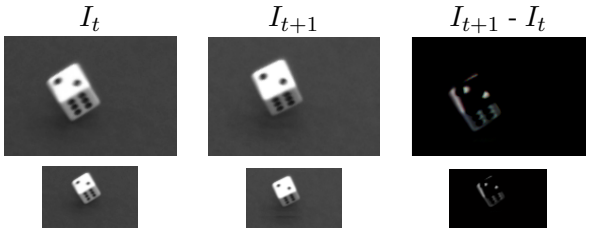


Fig. 13. Computationally Cheap.

5. Conclusion

This paper has proposed a method for 3D tracking high-resolution images in a video sequence at high frame-rates. It extends the process employed in our SSPR paper [17]. As in the previous version, this is a marker-less 3D tracking. It checks the cells where the 2D projections of the check-points should be and calculates the prediction error with a top-down method. The main novelty of this proposal is that, instead of searching an optimal level along of the pyramid to test the check-points, it now uses the properties of the maximum pyramid to know the sizes of the salient features at all levels of the pyramid. The method is robust to illumination and large view changes, computationally cheap and does not require large amount of memory. To demonstrate the new concept we have chosen a die because of its simple structure: six well distinguished faces and 21 dark dots. Future developments of our method will extract automatically the check-points from any possible salient feature shape.

Acknowledgements

This work has been granted by Doctoral College on Computational Perception (Vienna University of Technology, Austria)

References

[1] H. Bay, A. Ess, T. Tuytelaars, and L. J. Van Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding (CVIU) 110 (3)*, 2008. 1

[2] R. Bencina, M. Kaltenbrunner, and S. Jorda. Improved topological fiducial tracking in the reactivision system. *Computer Vision and Pattern Recognition (CVPR)*, 2005. 1

[3] L. Brun and W. G. Kropatsch. Construction of combinatorial pyramids. *Hancock, E.R., Vento,M. (eds.) GbRPR 2003. LNCS, vol. 2726, pp. 112. Springer*, 2003. 2

[4] D. Conte, P. Foggia, J. M. Jolion, and M. Vento. A graph-based, multi-resolution algorithm for tracking objects in presence of occlusions. *Pattern Recognition 39(4)*, 2006. 2

[5] P. David, D. DeMenthon, R. Duraiswami, and H. Samet. Simultaneous pose and correspondence determination using line features. *Computer Vision and Pattern Recogn (CVPR)*, 2003. 1

[6] B. K. P. Horn. Closed-form solution of absolute orientation using unit. *J. Optical Society of America, 4(4):629642*, 1987. 4

[7] Frederic Jurie and Michel Dhome. Real time 3d template matching. *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001. 1

[8] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. International symposium on augmented reality. *Virtual object manipulation on a table-top AR environment*, 2000. 1

[9] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. *In Proc. of ISMAR 2007*, 2007. 2

[10] W. G. Kropatsch. When pyramids learned walking. *The 14th International Congress on Pattern Recognition, CIARP 2009, volume 5856 of Lecture Notes in Computer Science, pp. 397414, Berlin Heidelberg*, 2009. 2

[11] V. Lepetit and P. Fua. Monocular model-based 3d tracking of rigid objects: A survey. *Foundations and Trends in Computer Graphics and Vision, vol. 1, no. 1*, 2005. 1

[12] D. Lowe. Distinctive image features from scale-invariant keypoints. *Computer Vision and Image Understanding 20*, 2004. 1

[13] K. Lu and Theo Pavlidis. Detecting textured objects using convex hull. *Machine Vision and Applications*, 2007. 1

[14] R. Marfil, L. Molina-Tanco, and S. F. Rodrguez. Real-time object tracking using bounded irregular pyramids. *Pattern Recognition Letters*, 2007. 2

[15] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast keypoint recognition using random ferns. *IEEE Transactions on Pattern Analysis and Machine Intelligence 32*, 2010. 1

[16] S. N. Sinha, J. M. Frahm, M. Pollefeys, and Y. Genc. Gpu-based video feature tracking and matching. *EDGE, Workshop on Edge Computing Using New Commodity Architectures*, 2006. 2

[17] F. Torres and W. G. Kropatsch. Top-down tracking and estimating 3d pose of a die. *Proceedings of Workshops on Structural and Syntactic Pattern Recognition (SSPR)*, 2012. 7

[18] Luca Vacchetti, Vincent Lepetit, and Pascal Fua. Stable real-time 3d tracking using online and offline information. *Unknown Journal*, 2004. 1

[19] D. Wagner, D. Schmalstieg, and H. Bischof. Multiple target detection and tracking with guaranteed framerates on mobile phones. *Proceedings of Int. Symposium on Mixed and Augmented Reality*, 2009. 2

[20] Heng Yang, Yueqiang Zhang, Xiaolin Liu, and Ioannis Patras. Coupled 3d tracking and pose optimization of rigid objects using particle filter. *International Conference on Pattern Recognition (ICPR)*, 2012. 1

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Registering 2D Imagery with 3D Range Scans

Brittany Morago
University of Missouri-Columbia
bagth5@mail.missouri.edu

Giang Bui
gdb338@mail.missouri.edu

Ye Duan
duanye@missouri.edu

Abstract. *Range scans provide detailed 3D structural information about stationary objects but lack continuous surfaces. Images and videos, on the other hand, contain cohesive, high resolution color information about scenes. The 2D imagery can also show how buildings and foliage change over time and record human motion. When we fuse these two valuable sources of information together, a new medium is created for viewing and studying data. In this paper, we discuss a fully automatic approach for creating 3D scenes by registering images and videos with range scans. Information obtained from images and videos taken at different times from different cameras can be combined with the full 3D structure of a scene provided by a LIDAR range scan. Fusing the 2D and 3D information provides a unique perspective not available when viewing either mode separately. This can be useful for easily constructing 3D models and creating simulations of athletic, transportive, artistic, or criminal activity. The data we work with includes a large variety of objects, including architecture, foliage, and people. We evaluate our system by computing the reprojection error for each calculated camera position. Our results are presented by showing 2D and the 2D-3D fused information for qualitative validation.*

1. Introduction

Photography and range scans often provide differing but complementing information about the same subject. In this paper, we propose a method for combining these two modalities by projecting 2D data onto 3D point clouds obtained with a LIDAR range scanner, as is shown in Figure 1. The 2D data consists of ordinary photographs and videos. We create a unique model allowing the 2D imagery to be viewed from different areas of interest with its correct 3D structural information in a single space. 2D imagery obtained at varying times and perspectives from arbitrary cameras can be fused with a single range scan. Data compiled in this manner can be utilized to review events that have occurred in an area such as vehicles driving around or a group of people congregating. All that is required is that the location of interest be scanned once. For instance, footage from a security camera can be projected onto a scan of a store or a jail to help law enforcement agencies better their method of visualization of the scene of interest. When information from different perspectives is combined in one system, new information may become available to viewers as they navigate and watch the scene from different angles. This program also paves a way toward creating a large database of range scans, images, and videos to be used for image localization. An individual can take a photo or video with their personal camera, upload it to the system which will match it to the appropriate scans, and show the person their location. This could help users navigate a new area or gain information about their current location. For instance, local and architectural history can be added into the database and be-

Figure 1. Three photographs projected onto scan of Jesse Hall.

come accessible whenever the user takes a picture of a particular building. 3D models constructed using our method may be beneficial for artists and the entertainment industry. Modeling architecture and foliage can be extremely time consuming and complex. However, our method facilitates the user obtaining a detailed model in the relatively short amount of time required to scan a scene and take several photographs.

1.1. Literature Review

A common goal is to register 2D images taken with arbitrary cameras with a 3D point cloud. The literature discusses solutions to this problem using a variety of data types. The methodologies described here are developed based on the information inherent in specific data sources and address difficulties posed by particular situations. The modality of the 3D data may be a range scan as used in the work of Stamos [26] and Neumann [18] or a structure from motion (SfM) point cloud as seen in Snavely's work [24] [25]. Various types of features (image-based [14], structural [26], etc.), certain system constraints (limiting camera motion to rotation [3]) and/or environmental information (GPS, GIS information, etc.) are always assumed to be present to help solve this problem [18] [9]. Often times, the proposed algorithm will call for finding keypoint matches [14] between the camera images [19]. Images can be matched to other regular cameras to create a new 3D relationship or to a set of images that have been preregistered with the 3D scan, as will be described in more detail below [26] [11] [5] [20].

Much of the work in this field focuses on 3D representations of urban environments that contain repetitive, linear structures and features. Linear features refer to the straight, identifiable "edges" in most architecture, such as windows and door frames. Schindler [21] handles this situation by identifying structural patterns in both the 2D and 3D data and then matching the patterns across dimensions. This approach avoids the confusion presented when using feature matching algorithms such as SIFT on repetitive data. The relative camera location to the 3D data is pinpointed using this relationship. Sinha [22] approaches the urban scene registration problem by identifying planes in an SfM point cloud. Matching planes in the 2D data are located by identifying vanishing points, creating 3D line segments, and constructing 3D planes. After registration, the planes are texture mapped. Planes in urban settings are also identified in Micusik's [17] and Gallup's [8] work. Gallup et al. perform multiple sweeps through a scene to identify planes with different normals.

Stamos et al. [26] and Li et al. [12] both fuse 2D photographs with range scans to produce photo-realistic models of urban scenes. These methods identify linear features and vanishing points in the 2D images. Stamos uses this information to estimate intrinsic camera parameters whereas Li relies on these calculations to rectify images. Wan et al. [27] also take advantage of the linear features inherent in urban scenes to estimate camera locations from vanishing lines. The estimates from this single-view geometry approach are combined to create a structural estimate of the entire scene.

Several groups have also worked on extracting information about video streams in relation to 3D structures. Nister [20] estimates camera motion in a video sequence, or a stereo camera setup. Features are extracted between images, the camera location is estimated, and an image trajectory is constructed. The entire camera motion estimate is based on visual information alone. In [29], Zhao et al. develop a method that works with continuous video as well. They project video onto a 3D point cloud obtained from a 3D sensor. Their method constructs an intermediary SfM point cloud from the video sequence using a dense stereo matching algorithm. Registration between the SfM and range scan point clouds is obtained using the iterative closest point algorithm is manually initialized [2]. This method is focused on airborne data similarly to the works of both Mastic [16] and Alharthy [1] have who have produced systems for matching 2D images to aerial LIDAR data.

There are several general methods for matching images to 3D models in order to obtain the camera location. The underlying goal is to solve for the camera parameters as discussed in Yang [28]. One method involves matching linear or circular structures between the 2D and 3D models. This approach is especially useful for urban models with highly regularized features as seen

in Friedman [7] and Schindler [21]. Another common methodology is to match keypoint features between camera images and a 2D image that is pre-registered with the 3D model. This can be used to perform the eight-point and five-point algorithms, as described in Ma [15] and Nister [19] in order to estimate the fundamental matrix, to solve for the projection matrix, to carry out 3D registration, and to perform object recognition. According to Yang [28], keypoint matches can be inaccurate when the images differ in qualities such as illumination and viewpoint. This greatly limits the datasets on which this type of approach can be reliably used. Irschara et al. [11] propose a solution to this problem. Their method uses SIFT matching between new images and images with known camera locations to find the locations of new images in relation to an existing 3D point cloud. Synthetic images from uncaptured viewpoints are constructed in such a way the most general viewpoints of a scene are represented with either a real or synthetic photo that can be matched to new images. Durand's work [5] demonstrates a similar idea to these papers by pinpointing the exact camera location and orientation of a historical photograph. Two new photographs are captured and used to construct a 3D model. This model is referenced to calculate the camera parameters for the historical photograph.

Several of these papers, though with similar results as those presented in this paper, require some form of user interaction when performing registration. Many methods rely on the user marking matching planes and line segments between 2D and 3D data [12][23]. Marking edges may also be used to calculate locations of vanishing points [12]. Minimal user interaction is required in [13] during the matching stage as well to ensure that the 3D data and the 2D image of interest are being viewed from similar perspectives.

1.2. Contributions

Our first main contribution is an automatic method for registering images, videos, and range scans using a minimal amount of information. Our method is quite flexible as we only require data from a scanner that any individual can operate. Our registration is performed using SIFT features [14] in the 2D imagery and the calibration information of the LIDAR camera. By relying on local feature descriptors and not looking to structural (linear, circular, etc.) features, we are able to work with datasets heavy in foliage and branch out from purely urban scenes. We treat the accurate LIDAR data as ground truth for our estimated 3D structure when determining camera locations. We also present a novel approach for viewing videos and 3D models. Unlike many methods mentioned above, we do not simplify the 3D geometry to a series of primitives or require a 3D polygonal mesh for texture mapping. By using a high resolution point cloud and projecting an image onto the structure, a 3D model is created using basic geometry without sacrificing fine details in architecture and foliage.

2. Methodology

Our registration method is based on the assumption that a point cloud and photographs with known 2D-3D correspondences are available. Ideally, the entire 3D point cloud should have matching 2D information in the form of photographs. We use a Leica C10 HDS LIDAR scanner which provides a high resolution, regularly spaced point cloud of a scene and 2D images of the scanned subject using a built-in camera. The LIDAR images have a resolution of 1920x1920 pixels. The output data from the scanner also consists of files containing the internal and external camera parameters for each image. This provides all the necessary information to establish a link between the 2D LIDAR images and the LIDAR range scan. The photographs and videos were taken with a variety of cameras including a Nikon D80 and various camera phones. We specify the image resolutions used in each test with our results. When handling videos, individual frames are extracted and treated as still images. Our data was collected on the University of Missouri-Columbia campus. For all of our tests, we assume the images and range scan used all involve the same subject.

We use this data to achieve our goal of projecting 2D imagery onto 3D range scans. A key issue in accomplishing the final projection is solving for the camera locations of all obtained 2D data in relation to the range scan. In order to solve this problem, our algorithm calculates the pro-

jection matrix of all camera images that have a high number of reliable keypoint matches with a subset of the LIDAR images. The 3D locations of the range scan points that match the 2D feature points are treated as the absolute locations for these points. This set of 2D-3D correspondences is used to calculate the projection matrix of the camera. The projection matrix represents the intrinsic and extrinsic camera parameters, yielding the camera's relationship to the range scan. Throughout this process, we consider the 3D LIDAR to be ground truth. Similarly to the manner that the GPS locations of the cameras are treated in [4], we want to minimize the distance between the LIDAR points and the 3D points inferred from the solved for camera position.

An exact mapping between the LIDAR camera images and the 3D point cloud is known from a provided file giving the camera's focal length in pixels and extrinsic parameters (rotation and translation). The point cloud can be projected onto the various image planes using this information. Each point in the point cloud is projected onto each image plane to find it's corresponding 2D points using Equations 1-3.

$$\hat{a} = R * a + t \tag{1}$$

$$x = f * \frac{\hat{a_x}}{\hat{a_z}} - \frac{w}{2} \tag{2}$$

$$y = f * \frac{\hat{a_y}}{\hat{a_z}} - \frac{h}{2} \tag{3}$$

where f is the focal length of the LIDAR camera in pixels, R is the rotation, t is the translation, w is the image width, and h is the image height. These parameters are particular to the individual image corresponding to the image plane in question. If the solved for x and y fall within the boundaries of the image dimensions, the 3D point has a corresponding 2D point in that image. The entire point cloud is projected onto each image once as a pre-processing step and the 2D-3D correspondences are saved. This step need only be performed once for any scan and is detailed in Algorithm 1.

Each camera image that is to be projected onto the range scan is matched against the LIDAR camera images using SIFT matching. At least six 2D-3D matches are needed to solve for the camera's projection matrix as is described in the Gold Standard Algorithm in [10]. The list of

Algorithm 1 Calculate 2D-3D Correspondence

```
procedure PROJECT(pts, camera parameters)
    n, m ← 0
    while n < number of pts do
        a ← pts[n]
        â ← R * a + t
        x ← f * â_x/â_z − w/2
        y ← f * â_y/â_z − h/2
        if 0 < x < image width then
            if 0 < y < image height then
                list[m] ← x, y, a
                m ← m + 1
            end if
        end if
        n ← n + 1
    end while
    return list
end procedure
```

2D-3D correspondences from the pre-processing stage is used to determine the 3D coordinates of the feature points in the LIDAR camera images. These 3D positions are considered to be extremely accurate representations of the matching 2D points in the regular camera image. Following this line of thinking, we use the set of 3D LIDAR point positions and 2D image points to solve for the projection matrix of the regular camera. This six-point algorithm is carried out using RANSAC [6] to find the set of matches that calculate the most accurate projection matrix. The most accurate matrix is defined as the one that yields the lowest average reprojection error for all of the feature correspondences. Matches with a reprojection error lower than a certain threshold are identified as inliers, and another round of RANSAC is preformed using only these correspondences. The final projection matrix is once again the one that produces the lowest average reprojection error.

Once the projection matrix, and thus the camera parameters, is known, the image is registered with the range scan. To display this registration, the scan is projected onto the image as described in Algorithm 1. The color assigned to each 3D point is the color of the pixel onto which it most closely projected. This 2D-3D correspondence is saved for each image that is projected onto the

scan. This image can then replace a LIDAR image in the steps previously described. This is extremely useful as a user's image or video may vary greatly in scale, illumination, and perspective from the set of LIDAR images. This can lead to the feature matching step producing erroneous correspondences. By creating a web of images each with gradual lighting and viewing location changes, more images can be accurately matched and added into the system. The system will perform much better if the initial images matched to the range scan are similar in perspective and illumination to the LIDAR images. Gathering images taken under similar conditions to the LIDAR images provides a strong base for creating this web. When displaying the final registration, the 3D points can be drawn in varying sizes depending on the density of the range scan. Larger points can be useful for filling in holes. The 3D model can then be manipulated for novel-view generation purposes.

Figure 2. Above: Image projected on Columns Scan. Below: Original camera image. Image resolution is 1936x1296.

3. Results and Discussion

We judge the accuracy of our results both qualitatively and quantitatively. When the 2D images are viewed next to the registration of the images and the range scan (Figures 2, 3, and 5), one can easily visually judge the accuracy of the projected data. We present the results from a variety of viewpoints in Figures 1 and 4 to show the usefulness of the method for novel-view generation. Figure 5 shows extracted frames from a video of a person walking in front of Cornell Hall projected onto a range scan. Regular photographs are also projected onto the scan to provide color information for portions of the scan that were not captured in the video. This example shows how our program can be used to view the full context of a scene when only a portion has been recorded.

Figure 3. Above: Two images projected onto Cornell Hall Scan. Bottom Left: Photograph that is projected onto left side of scan. Bottom Right: Photograph projected onto right side of scan. Image resolution is 1932x1294.

The reprojection error associated with a chosen camera position is also considered in judging the accuracy of our results. This error is a measure of how far a 3D point is from its corresponding 2D pixel when the 3D point is projected onto the image plane. The reprojection errors and number of feature matches and inlier matches associated with our presented results are shown in Table 1. The reprojection error is measured in image pixels in the coordinate system of the image being registered with the range scan. The errors for our results are well under a pixel, aligning with the accuracy observed in the pictorial registration results. We have obtained high accuracy on images ranging greatly in resolution (780x420 to 1936x1296). These errors can be further decreased by removing radial distortion in the images and performing bundle adjustment on both pairs of images and the set of projected images.

The performance of this system can be enhanced in the future by not projecting every point in the range scan onto every image plane as described in Algorithm 1. An approach can be developed for partitioning the range scan into sections based on location and narrowing down what regions of the scan are viewable from a certain camera location.

Image	RE	M	I
Columns (Fig. 2)	0.075	93	85
Cornell Left (Fig. 3)	0.046	197	171
Cornell Right (Fig. 3)	0.171	316	277
Jesse Hall Left (Fig. 4)	0.092	155	141
Jesse Hall Center (Fig. 4)	0.151	446	423
Jesse Hall Right (Fig. 4)	0.165	404	379
Video Frame 4 (Fig. 5)	0.078	143	122
Video Frame 26 (Fig. 5)	0.081	99	87
Video Frame 49 (Fig. 5)	0.046	316	277

Table 1. Reprojection Error(RE) of presented results, measured in pixels, with number of feature matches(M) and number of inlying correspondences(I).

4. Conclusion and Future Work

In this paper, a system has been presented that matches sets of ordinary 2D images and videos taken at different times, from different angles, and by different people to a single range scan. This information can be used to create 3D models of both architecture and foliage that may be observed from novel viewpoints. The registration process is completely automatic and the final output allows the user to view images from a variety of sources in one cohesive environment. This method can be easily expanded upon to work with large datasets of images. Once a set of images is registered with the scan, a SfM point cloud based on these images can be created to augment the range scan. The SfM point cloud can be used to fill in holes in the range scan or insert objects into the scene that were not originally viewed such as vehicles, people, and new foliage. A large set of matching images can also be used to perform a global optimization on all of the camera poses and increase the accuracy of the system.

Acknowledgements

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant Number 0943941 and the US Department of Education under Grant Number P200A100053.

References

[1] A. Alharthy[1] and J. Bethel. Detailed building reconstruction from airborne laser data using a moving surface method. 2004. 2

[2] P. Besl and N. McKay. A method for registration of 3-D shapes. *IEEE Transactions on pattern analysis and machine intelligence*, pages 239–256, 1992. 2

[3] D. Craciun, N. Paparoditis, and F. Schmitt. *Image-Laser Fusion for In Situ 3D Modeling of Complex Environments: A 4D Panoramic-Driven Approach.* InTech, 2012. 2

[4] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3001–3008. IEEE, 2011. 4

[5] F. Durand, A. Agarwala, S. Bae, F. Durand, et al. Computational Re-Photography. 29(3), 2010. 2, 3

[6] 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. 4

[7] S. Friedman and I. Stamos. Real Time Detection of Repeated Structures in Point Clouds of Urban Scenes. In *Proceedings of the 2011 International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, pages 220–227. IEEE Computer Society, 2011. 3

[8] D. Gallup, J. Frahm, P. Mordohai, Q. Yang, and M. Pollefeys. Real-time plane-sweeping stereo with multiple sweeping directions. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007. 2

[9] R. Grzeszczuk, J. Kosecka, R. Vedantham, and H. Hile. Creating compact architectural models by geo-registering image collections. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1718–1725. IEEE, 2009. 2

[10] R. Hartley and A. Zisserman. *Multiple View Geometry.* Cambridge University Press, Cambridge, United Kingdom, 2010. 4

[11] A. Irschara, C. Zach, J. Frahm, and H. Bischof. From structure-from-motion point clouds to fast location recognition. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2599–2606. IEEE, 2009. 2, 3

[12] Y. Li, Q. Zheng, A. Sharf, D. Cohen-Or, B. Chen, and N. Mitra. 2D-3D fusion for layer decomposition of urban facades. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 882–889. IEEE, 2011. 2, 3

[13] L. Liu and I. Stamos. A systematic approach for 2D-image to 3D-range registration in urban environments. *Computer Vision and Image Understanding*, 116(1):25–37, 2012. 3

[14] D. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 2, 3

[15] Y. Ma, S. Soatta, J. Kosecka, and S. S. Sastry. *An Invitation to 3-D Vision*. Springer, New York, New York, 2004. 3

[16] A. Mastin, J. Kepner, and J. Fisher. Automatic registration of LIDAR and optical images of urban scenes. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2639–2646. IEEE. 2

[17] B. Micusik and J. Kosecka. Piecewise planar city 3D modeling from street view panoramic sequences. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2906–2912. IEEE, 2009. 2

[18] U. Neumann, S. You, J. Hu, B. Jiang, and J. Lee. Augmented virtual environments (ave): Dynamic fusion of imagery and 3d models. 2003. 2

[19] D. Nistér. An efficient solution to the five-point relative pose problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(6):756–770, 2004. 2, 3

[20] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–652. IEEE, 2004. 2

[21] G. Schindler, P. Krishnamurthy, R. Lublinerman, Y. Liu, and F. Dellaert. Detecting and matching repeated patterns for automatic geo-tagging in urban environments. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–7. IEEE, 2008. 2, 3

[22] S. Sinha, D. Steedly, and R. Szeliski. Piecewise planar stereo for image-based rendering. In *Proc. ICCV*, pages 1881–1888, 2009. 2

[23] S. Sinha, D. Steedly, R. Szeliski, M. Agrawala, and M. Pollefeys. Interactive 3D architectural modeling from unordered photo collections. In *ACM Transactions on Graphics (TOG)*, volume 27, page 159. ACM, 2008. 3

[24] N. Snavely, S. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 835–846. ACM, 2006. 2

[25] N. Snavely, S. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *International Journal of Computer Vision*, 80(2):189–210, 2008. 2

[26] I. Stamos, L. Liu, C. Chen, G. Wolberg, G. Yu, and S. Zokai. Integrating automated range registration with multiview geometry for the photorealistic modeling of large-scale scenes. *International Journal of Computer Vision*, 78(2):237–260, 2008. 2

[27] G. Wan, N. Snavely, D. Cohen-Or, Q. Zheng, B. Chen, and S. Li. Sorting unorganized photo sets for urban reconstruction. *Graphical Models*, 74(1):14–28, 2012. 2

[28] G. Yang, J. Becker, and C. Stewart. Estimating the location of a camera with respect to a 3d model. In *3-D Digital Imaging and Modeling, 2007. 3DIM'07. Sixth International Conference on*, pages 159–166. IEEE, 2007. 2, 3

[29] W. Zhao, D. Nister, and S. Hsu. Alignment of continuous video onto 3D point clouds. *IEEE transactions on pattern analysis and machine intelligence*, pages 1305–1318, 2005. 2

Figure 4. Above: Three Images projected onto Jesse Hall Scan (two perspectives shown). Bottom: Left, center, and right original images that are projected onto the scan. Image resolution is 1549x1037.

Figure 5. Left: Three frames extracted from a video sequence. Video frame resolution is 720x480. From top to bottom, frames 4, 49, and 126. Right: Frames 4, 26, and 49 projected onto Cornell Hall Scan on top of previously shown images.

18^{th} *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Constrained Bundle Adjustment for Panoramic Cameras

Cenek Albl and Tomas Pajdla
Czech Technical University, Faculty of Electrical Engineering
Karlovo Namesti 13, Prague, Czech Republic
albl cene@cmp.felk.cvut.cz, pajdla@cmp.felk.cvut.cz

Abstract. *In this paper we propose, design and implement a method of Bundle adjustment (BA) incorporating constraints that describe a camera undergoing a circular motion. Using these constraints we are able to capture the physical properties of various rotating camera systems into the BA process, for example panoramic camera systems used on Mars rovers or turntables. We incorporated our method into the recently released Google non-linear least squares solver Ceres. By using constrained BA, we aim at improving the accuracy of reconstructing both the 3D points and the camera positions. The improvement in accuracy was experimentally verified on synthetic as well as real datasets.*

1. Introduction

There has been a lot of attention given to the Structure from motion reconstruction, resulting in many practical applications such as Photosynth and Bundler [13]. Current research aims to provide more precise reconstruction as well as the ability to handle larger datasets [1], [3].

Bundle Adjustment (BA) [14] is an important part of the structure from motion reconstruction as it optimizes the resulting estimates of 3D point coordinates and the position, orientation and calibration of cameras. Detailed analysis of BA optimization methods, parametrizations, error modeling and constraints has been given in [14]. An efficient and comprehensive algorithm that utilizes the sparsity of BA has been developed by Lourakis and Argyros [10] and the code is made freely available. This algorithm has been further used in [13], to build a full structure from motion pipeline. An extended version of [10] has been developed in [7] utilizing the sparsity even further in order to reduce computation time. Recently, the performance of BA on large datasets has been scrutinized [1]. The use of conjugate gradients and its effect on performance has been investigated in [2]. In [6], significant performance improvements using multiple techniques, such as embedded point iterations and preconditioned conjugate gradients were shown.

1.1. Constrained Bundle Adjustment

A general graph solver which can be used for BA is described in [12] and available online. Although [12] seems to be capable of handling camera constraints, authors have not investigated this possibility. BA with constraints on the structure has been used in [15] and [11]. Experiments with camera constrained BA designed specifically for stereo rig were carried out in [8]. The results showed that modeling stereo camera pair with proper constraints and incorporating it into BA can improve the precision of the reconstruction. In [4], authors incorporate constraints for a Pancam system consisting of two cameras mounted on a rotating shaft used on a Mars Exploration Rover mission. BA designed for a turntable has been presented in [15], where authors described the object as a set of points rotating around an axis with a fixed distance from the camera. They showed that this projection model brought an improvement in terms of precision of the reconstruction.

1.2. Contribution

In this paper we provide a general approach to describe a rotating camera and incorporate it into Bundle Adjustment optimization implemented in the open source solver Ceres from Google. In comparison to [4] and [15] our method is not limited to a specific Pancam system or a turntable even though it is capable of handling both of them. Using our approach we can optimize wide variety of scenarios, such as multiple panoramic cameras in one scene,

cameras undergoing circular motions with different radii, orientations and on different locations or turntables with more than just one camera position. This is done by adjusting the camera model, keeping only the circular motion constraints. The initialization does not require any previous knowledge about the scene as in the case of [4], instead we estimate the constraints from an initial unconstrained SfM result. In our experiments we confirm that using just enough constraints to keep the generality still leads to improvement of the reconstruction in terms of precision.

In section 2 we describe the basics of BA and introduce the notation we will use further in the paper. Section 3 presents the model used for rotating camera systems. Experiments and their results are shown in section 4 and the results discussed in section 5.

2. Bundle Adjustment

We build our method upon the Google non-linear least squares optimizer Ceres, which encompasses several methods for Bundle Adjustment, including the approaches described in [1] and [9]. We set Ceres to use a Levenberg-Marquardt algorithm [14] which iteratively solves the normal equation

$$(\mathbf{J^T J} + \frac{1}{\mu}\mathbf{D})\delta\mathbf{p} = \mathbf{J^T e} \tag{1}$$

where $\mathbf{J}$ is the Jacobian of the projection function, $\mathbf{D}$ is a matrix containing a diagonal of $\mathbf{J}$, $\mathbf{p}$ is the vector of parameters of cameras and 3D points and $\mathbf{e}$ is the error vector between the measured and predicted image points. The trust region radius μ controls the magnitude of sought updates $\delta\mathbf{p}$ to the parameter vector. Detailed description can be found in [5] and [14]. $\mathbf{J}$ and $\mathbf{J^T J}$ have known sparse structure and do not have to be computed explicitly. Ceres can make use of the Schur complement [5] to solve (1) for the camera parameters first, which requires less computational effort.

The projection functions we use are based on the perspective camera projection

$$\lambda\mathbf{x} = \mathbf{PX} \tag{2}$$

where $\mathbf{X}$ is a 3D point and $\mathbf{x}$ the image point, both in homogeneous coordinates. $\mathbf{P}$ is the projection matrix according to [5] and it can be decomposed as

$$\mathbf{P} = \mathbf{KR}\,[\mathbf{I}|\,-\mathbf{C}] \tag{3}$$

where $\mathbf{K}$ is the calibration matrix, $\mathbf{R}$ is a rotation matrix from world to camera coordinates and $\mathbf{C}$ is the camera center in the world coordinates. According to [5], the calibration matrix can be described by 5 parameters as

$$\mathbf{K} = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & ar\alpha_x & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

Where

$$ar = \frac{\alpha_y}{\alpha_x} \tag{5}$$

is the aspcet ratio.

3. Camera systems with rotational constraint

When dealing with multiple camera systems, which have some fixed physical properties, it would be meaningful to utilize this additional information in the BA process. We did that by using an adapted projection function describing the relationships between cameras. Such systems can be a camera rotating around an axis, where our additional knowledge is represented by the fact that camera centers lie on a circular trajectory. Each camera center $\mathbf{C_i}$ can therefore be expressed by a function involving center of rotation, the orientation of the rotation plane in space, the radius of the rotation and a displacement angle of the camera. Such equation can be written as

$$\mathbf{C}_i = \mathbf{R_c}\begin{bmatrix} \rho\cos\phi_i \\ \rho\sin\phi_i \\ 0 \end{bmatrix} + \begin{bmatrix} C_{cx} \\ C_{cy} \\ C_{cz} \end{bmatrix} \tag{6}$$

where $\mathbf{R_c}$ is a rotation matrix representing the orientation of the panoramic rotation plane in space, vector $\mathbf{C}_c = [C_{cx}, C_{cy}, C_{cz}]^T$ represents the center of rotation, ρ the radius and ϕ the camera displacement angle. Having $\mathbf{C}_i$ we can construct $\mathbf{P}_i$ followingly

$$\mathbf{P}_i = \mathbf{R}_i\,[I|\,-\mathbf{C}_i] \tag{7}$$

When we express the rotation matrices using quaternions, we obtain the following parameter vector

$$\mathbf{a}_j = [\mathbf{k}, \mathbf{q}, \mathbf{C}_c^T, \mathbf{q}_c, \rho] \tag{8}$$

where

$$\begin{array}{rcl} \mathbf{k}_i & = & [f_x, x_0, y_0, ar, s] \\ \mathbf{q}_i & = & [q_{i1}, q_{i2}, q_{i3}, q_{i4}, \phi_j] \\ \mathbf{C}_c & = & [C_{cx}, C_{cy}, C_{cz}]^T \\ \mathbf{q}_c & = & [q_{c1}, q_{c2}, q_{c3}, q_{c4}] \end{array} \tag{9}$$

so the blocks denoted by index i belong to each individual camera and blocks denoted by j describe the circular constraints for a cluster of cameras. It is clear, that parameters describing the panoramic rotation (C_c, q_c, ρ) will be identical for all cameras from the same panorama. It is possible within the Ceres platform to make any of these blocks shared by multiple cameras. The reason why we chose such blocks is that we can set any combination of radius, circle orientation and circle position shared also between multiple camera clusters. As an example, Pancam in [4] could be described by two camera clusters with shared rotation axis position and orientation but different radii.

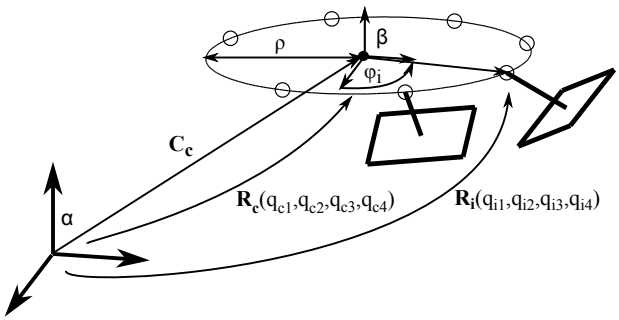


Figure 1. Visualization of parameters used to describe panoramic camera. Parameters C_c, R_c and ρ describe the circle of rotation, therefore they are shared among all cameras within this panorama. C_i and ϕ_i determine the positions of individual camera centers and R_i their orientations.

4. Experiments

To validate our method, we performed a set of tests on synthetic and real data. We compared our extended bundle adjustment (XBA) algorithm to SBA, upon which our algorithm was built.

4.1. Parametrization

Shared parameters in XBA were set to utilize all physically meaningful properties of the camera system. In the case of panoramic cameras, the camera centers were placed on circles and all circles shared the same radius. Stereo setup used a shared baseline and orientation of the second camera with respect to the first one over the whole sequence. Both SBA and XBA adjusted 3 internal camera calibration parameters (f_x,x_0,y_0) and kept the 2 remaining fixed (ar,s). We also estimated the radial distortion using the same model as [13]. The overall number of parameters describing a camera was 14 (five for intrinsics, four for camera orientation, three for camera center location and two for radial distortion) for SBA and 20 (as in (8) plus two for radial distortion) for XBA. Because two intrinsic were kept fixed, the numbers of parameters adjusted were 12 and 18 for SBA and XBA respectively. The XBA parameters were described in section 3.

Table 1. Datasets

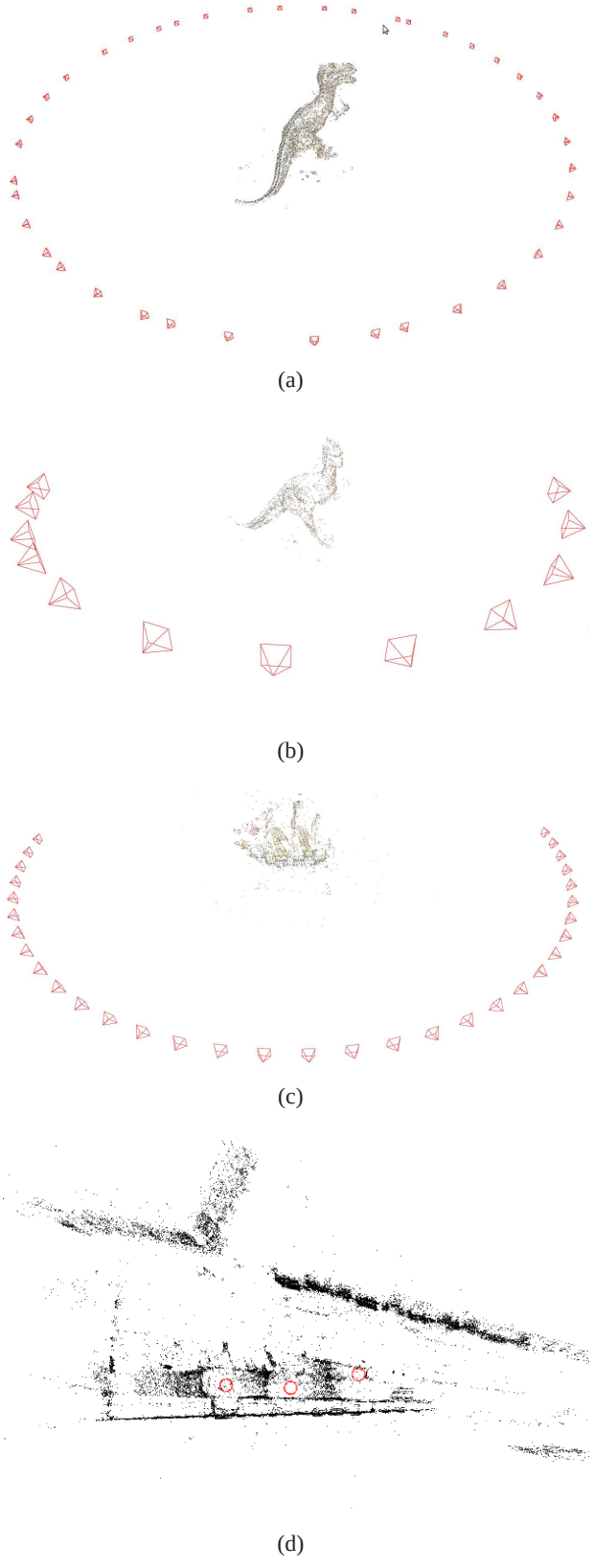
	cameras	points	projections
Dino1	40	14038	42241
Dino2	20	3080	7343
Ship	13	8851	29259
Street	54	34443	93348

4.2. Datasets

We examined the performance of XBA on four real world dataset captured using a panoramic camera setup. Pictures were acquired using a Nikon D60 and D3100 DSLR cameras. First dataset was simulating a panoramic observation of environment, using a camera rotating on a shaft mounted on a tripod. Three panoramic observations were made on three different locations, each consisting of 18 images of an urban environment. The data was then reconstructed using the SfM pipeline [13] and estimates for camera and structure parameters were obtained. The rest of the real world datasets was obtained using a turntable with two objects each with different shape properties. The turntable was used to capture the model from various viewpoints, while keeping the circular trajectory of the camera with respect to the object.

To evaluate our method, we needed some ground truth information about the scenes. For that purpose we took the outcome of the SfM pipeline, placed the cameras on circular trajectories (as they were in reality) and projected the 3D points into new image projections. Therefore, we obtained a perfect ground truth information in a real world scenarios.

In all datasets, the 3D points were perturbed by a uniformly distributed error $e \in [0; d/20]$, where d is the distance from the camera centers. The image projections were perturbed by a Gaussian noise with mean 0 and standard deviation of 0.7.



(a)

(b)

(c)

(d)

Figure 2. Perturbed initializations for datasets (a) Dino1, (b) Dino2, (c) Ship and (d) Street.

4.3. Analysis

In measurements we analyzed the following aspects

1. Error in reconstructed structure with respect to ground truth
2. Error in reconstructed camera positions with respect to ground truth
3. Evolution of the reconstruction error during iterations
4. Evolution of the mean reprojection error

To express the reconstruction error, we used a least square fit of either the resulting 3D point positions or the camera center positions onto the ground truth data. Camera fit means that a transformation was found between the reconstructed camera centers and the ground truth camera centers and this transformation was then applied to reconstructed camera centers as well as 3D points. Points fit applies the same principle but the transformation between reconstructed and ground truth 3D points was found in this case. After the fit, error in both camera centers and 3D points was measured as the Euclidean distance from the correct positions. These measurements express how accurately were the camera positions and 3D points reconstructed and also how accurately can we determine the camera positions knowing the 3D structure parameters and vice versa. This evaluation was also carried out for each individual BA step to show how the accuracy of the model develops during the process. For each dataset, the tests were run multiple times with different random initialization.

Results of reconstruction error of camera centers are in figure 4, which shows, how precisely we estimated the surrounding structure relatively to the camera positions, e.g. their relative scale. The precision of reconstructed structure with respect to the ground truth can be read from figure 5. Analogically to the camera centers' case, figure 5 now show how precisely we can determine the camera centers relatively to the structure.

To investigate the behavior of XBA versus SBA deeper, we show for each iteration on the dataset Dino2 the mean reprojection error and the reconstruction error in Figure .

5. Discussion

As shown in figure 3, more dimensions of freedom cause SBA to achieve lower image reprojection error,

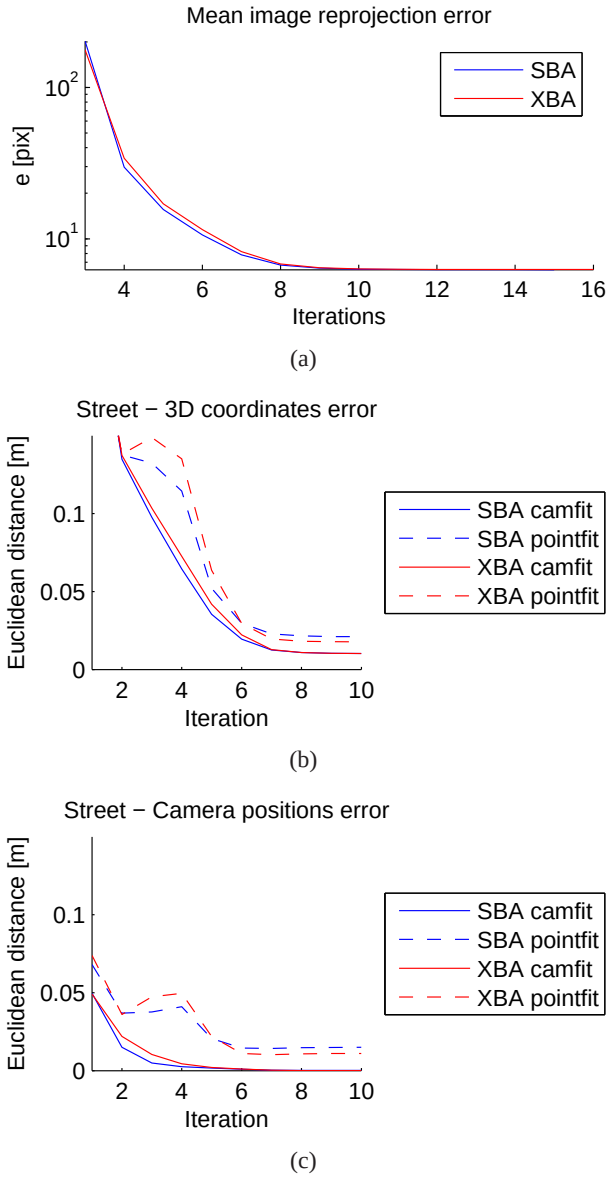


Figure 3. **Real PANCAM**. (a) The image reprojection error during iterations of SBA and XBA on the dataset Street. (b) Development in 3D point and camera position error during iterations of SBA and XBA on the dataset Street.

but the constraints ensure that the 3D reconstruction error becomes lower using XBA.

The results, summarized in table 2, show that XBA can in certain scenarios outperform SBA in terms of reconstruction precision. Particularly, in the dataset Street the reconstruction errors of XBA were much smaller than of SBA. As can be seen in the results, the difference between XBA and SBA varies with different initializations. On rare occasions, SBA achieves similar results as XBA or even slightly better. However, on average, the reconstruction using

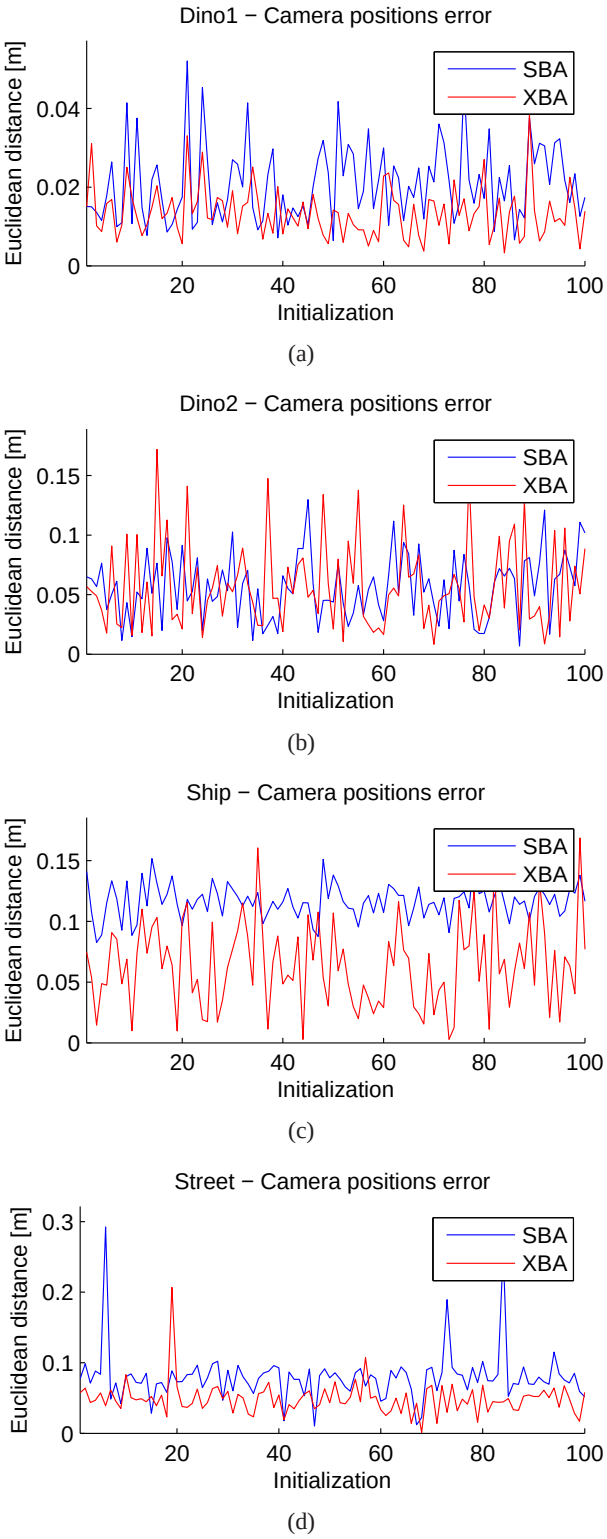
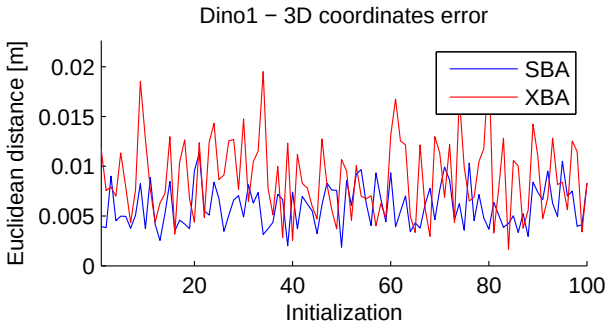


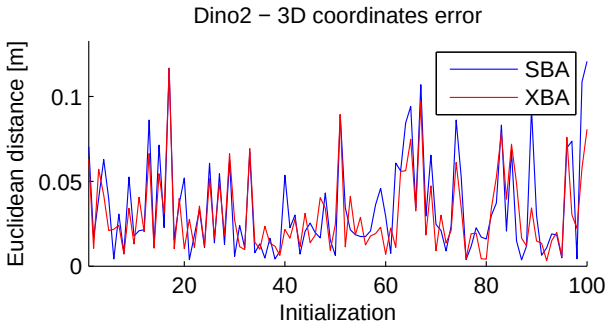
Figure 4. **Camera position error**. Error in camera center positions after reconstructing and fitting through 3D point positions for datasets (a) Dino1, (b) Dino2, (c) Ship and (d) Street.

XBA was more correlated with the ground truth.

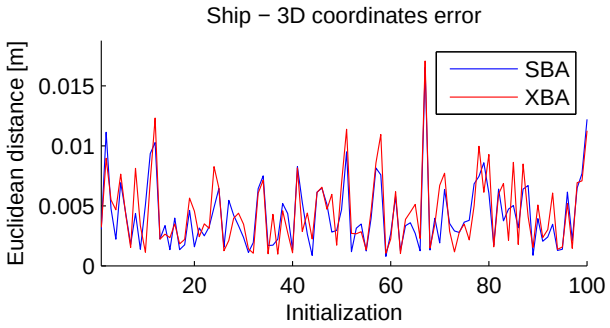
This could be useful in many applications, such as



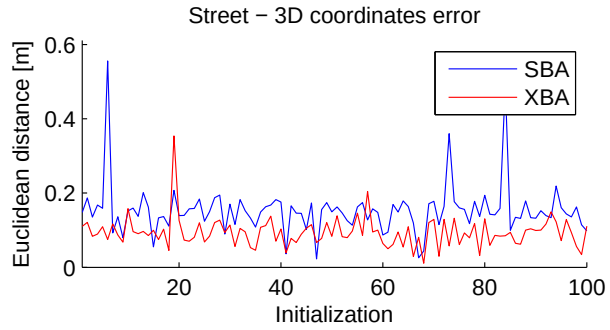
(a)



(b)



(c)



(d)

Figure 5. **Camera position error**. Error in camera center positions after reconstructing and fitting through 3D point positions for datasets (a) Dino1, (b) Dino2, (c) Ship and (d) Street.

localization and mapping. If one knows information about the dimension of either the structure or camera positions (for example the diameter of panoramic rotation), one can better estimate the dimension of the other.

Table 2. Summary of the results for all datasets

Mean reconstruction error [m]					
		SBA		XBA	
		cams	pts	cams	pts
Dino1	cfit	0.0207	**0.0060**	**0.0121**	0.0089
	pfit	0.0208	0.0035	**0.0135**	**0.0017**
Dino2	cfit	0.0166	0.0348	**0.0120**	**0.0306**
	pfit	**0.0564**	**0.0066**	0.0567	0.0075
Ship	cfit	0.0033	**0.0042**	**0.0023**	0.0045
	pfit	0.1166	0.0149	**0.0638**	**0.0082**
Street	cfit	0.0763	0.1528	**0.0456**	**0.0939**
	pfit	0.796	0.1528	**0.0489**	**0.0950**

For the datasets acquired using a turntable the performance of XBA was not always better than SBA as in the case of the Street dataset. In dataset Dino1 XBA handled better the reconstruction of the scene with respect to the points and in dataset Dino2 with respect to the cameras. Since Dino1 has far less cameras, points and projections, this could be interpreted as the ability of constrained BA to maintain the correct positions of cameras even with less observations and interconnectivity of the cameras and points in the scene.

Another situation occurs in the case of the Ship dataset. The points fit slightly better to the ground truth cameras using SBA, but points with respect to points and cameras with respect to either cameras or points are reconstructed better using XBA by a large margin. In this scenario, the camera circle is purposively incomplete and, as expected, XBA handled the camera positions much better. This could be useful if we are interested in precise poses of cameras, such as in the case of dense stereo reconstruction.

However, it can be concluded, that XBA does not bring such an improvement in the case of turntable scenarios, where all cameras are pointing towards a single object and the scene is heavily interconnected, i.e. cameras have high overlaps and each point is observed by many cameras. The potential of XBA seems to be more in the outdoor scenarios, where cameras are pointing outwards of the axis of rotation and have smaller overlap.

6. Conclusion

We proposed and implemented a new method of constrained BA (XBA) for cameras undergoing circular motion into Google optimization software Ceres. This method incorporates circular constraints for systems such as panoramic camera or a turntable. The method was validated on several real datasets and the reconstruction error in camera positions and 3D points was measured. The results show for certain scenarios improved quality of the reconstruction over the previous method. The advantages of this method and its possible applications were discussed.

Acknowledgements

The authors were supported by Grant Agency of the CTU Prague project SGS12/191/OHK3/3T/13.

References

[1] S. Agarwal, N. Snavely, S. M. Seitz, and R. Szeliski. Bundle adjustment in the large. In *ECCV (2)*, pages 29–42, 2010. 1, 2

[2] M. Byröd and K. Åström. Conjugate gradient bundle adjustment. In *Proceedings of the 11th European conference on Computer vision: Part II*, ECCV'10, pages 114–127, Berlin, Heidelberg, 2010. Springer-Verlag. 1

[3] D. Crandall, A. Owens, and N. Snavely. Discrete-continuous optimization for large-scale structure from motion. *IEEE Conference on Computer Vision and Pattern Recognition (2008)*, 286(26):3001–3008, 2011. 1

[4] K. Di, F. Xu, R. Li, B. Adjustment, and P. Image. Constrained bundle adjustment of panoramic stereo images for mars landing site mapping. *MMT*, 2003. 1, 2, 3

[5] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004. 2

[6] Y. Jeong, D. Nister, D. Steedly, R. Szeliski, and I.-S. Kweon. Pushing the envelope of modern methods for bundle adjustment. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 1474–1481, 2010. 1

[7] K. Konolige. Sparse sparse bundle adjustment. In *British Machine Vision Conference*, Aberystwyth, Wales, 08/2010 2010. 1

[8] C. Kurz, T. Thormählen, and H.-P. Seidel. Bundle adjustment for stereoscopic 3D. In A. Gagalowicz and W. Philips, editors, *5th International Conference on Computer Vision/Computer Graphics Collaboration Techniques (MIRAGE 2011)*, volume 6930 of *Lecture Notes in Computer Science*, pages 1–12, Rocquencourt, France, October 2011. Inria, Springer. 1

[9] A. Kushal. Visibility based preconditioning for bundle adjustment. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, pages 1442–1449, Washington, DC, USA, 2012. IEEE Computer Society. 2

[10] M. I. A. Lourakis and A. A. Argyros. Sba: A software package for generic sparse bundle adjustment. *ACM Transactions on Mathematical Software*, 36:1–30, 2009. 1

[11] C. McGlone. *Bundle Adjustment with Geometric Constraints for Hypothesis Evaluation*, pages 529–534. 1996. 1

[12] H. S. K. K. Rainer Kuemmerle, Giorgio Grisetti and W. Burgard. g2o: A general framework for graph optimization. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. 1

[13] N. Snavely. Bundler: Structure from motion (sfm) for unordered image collections. `http://phototour.cs.washington.edu/bundler/`, 5 2011. 1, 3

[14] B. Triggs, P. Mclauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment a modern synthesis. In *Vision Algorithms: Theory and Practice, LNCS*, pages 298–375. Springer Verlag, 2000. 1, 2

[15] K. H. Wong, M. Ming, and Y. Chang. 3d model reconstruction by constrained bundle adjustment. In *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 3 - Volume 03*, ICPR '04, pages 902–905, Washington, DC, USA, 2004. IEEE Computer Society. 1

Author Index

ISBN: 978-3-200-02943-9