Medical Image Analysis 34 (2016) 82–100

Contents lists available at ScienceDirect

Medical Image Analysis

journal homepage: www.elsevier.com/locate/media

Real-time localization of articulated surgical instruments in retinal microsurgery

Nicola Rieke[a,*], David Joseph Tan[a], Chiara Amat di San Filippo[a,c], Federico Tombari[a,d], Mohamed Alsheakhali[a], Vasileios Belagiannis[b], Abouzar Eslami[c], Nassir Navab[a]

[a] Computer Aided Medical Procedures, Technische Universität München, Germany
[b] Visual Geometry Group, Department of Engineering Science, University of Oxford, Great Britain
[c] Carl Zeiss Meditec AG, München, Germany
[d] DISI, University of Bologna, Italy

ARTICLE INFO

Article history:
Received 14 January 2016
Revised 3 May 2016
Accepted 3 May 2016
Available online 13 May 2016

Keywords:
Pose estimation
Visual tracking
Retinal microsurgery

ABSTRACT

Real-time visual tracking of a surgical instrument holds great potential for improving the outcome of retinal microsurgery by enabling new possibilities for computer-aided techniques such as augmented reality and automatic assessment of instrument manipulation. Due to high magnification and illumination variations, retinal microsurgery images usually entail a high level of noise and appearance changes. As a result, real-time tracking of the surgical instrument remains challenging in *in-vivo* sequences. To overcome these problems, we present a method that builds on random forests and addresses the task by modelling the instrument as an articulated object. A multi-template tracker reduces the region of interest to a rectangular area around the instrument tip by relating the movement of the instrument to the induced changes on the image intensities. Within this bounding box, a gradient-based pose estimation infers the location of the instrument parts from image features. In this way, the algorithm does not only provide the location of instrument, but also the positions of the tool tips in real-time. Various experiments on a novel dataset comprising 18 *in-vivo* retinal microsurgery sequences demonstrate the robustness and generalizability of our method. The comparison on two publicly available datasets indicates that the algorithm can outperform current state-of-the art.

1. Introduction

Retinal Microsurgery (RM) is a delicate surgical procedure, which requires high handling precision for the utilized instruments under limited visual feedback. In routine surgery such as membrane peeling, the surgeon has to manipulate anatomical features (*layers*) which are less than 10 \mu m thick. One of the main difficulties is caused by the fact that the surgeon can only observe the procedure in an indirect way through a microscope. The interpretation of the perceived depth then becomes quite challenging and the high magnification leads to lens distortions such that, in most cases, only a portion of the observed scene is focused. Furthermore, the haptic feedback is weak. All these problems limit the possibilities for the surgeon to identify or grasp surgical targets and consequently may increase operating time and the risk for retinal damage.

Recently, new microscopes introduced on the market provide additional intraoperative imaging information to the surgeon, by visualizing subretinal structure information via Optical Coherence Tomography (OCT). OCT imaging has become widespread in ophthalmology over the past 20 years because of its ability to visualize ocular structures at high resolution (Gabriele et al., 2011). Its new intraoperative version (iOCT) has opened up new research fields and paved the way to new applications (Ehlers et al., 2014), thanks to the fact that depth information within the tissue can be obtained in real-time during the procedure. In the current workflow, these devices have to be manually positioned on the region of interest which further increases the complexity of the device handling for the surgeons, who already have to manipulate the surgical tools, the manual light source and the microscope.

Recent works have been aiming at introducing specific computer vision algorithms in order to overcome the current technical limitations and support the surgeon in a direct or indirect way. For instance, the visual data acquired from the microscope can be processed for stitching different frames together, in order to create a wider field of view of the retina, e.g., by Cattin et al. (2006);

* Corresponding author.
E-mail address: nicola.rieke@tum.de (N. Rieke).

http://dx.doi.org/10.1016/j.media.2016.05.003
1361-8415/

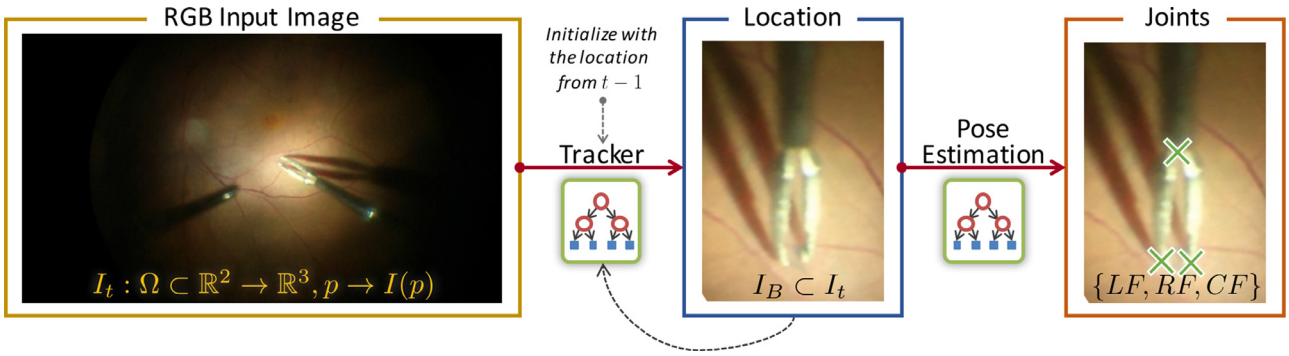


Fig. 1. Pipeline: The figure illustrates an overview of the algorithm. Given a frame of a video sequence at time t, the temporal tracker determines the region of interest around the tool tip. It takes the bounding box from the previous frame and iteratively refines its location. Thereafter, the bounding box is used as input to the pose estimation to find the positions of the three joints – left joint (LF), right joint (RF), central joint (CF).

or, for providing landmarks on the retina, but also simultaneously tracking surgical tool and its shadow as presented by Yigitsoy et al. (2015). Within this field, the capability of robustly tracking the surgical instrument at each frame represents a key component for various applications.

There are different applications that can benefit from efficient and robust tool tracking such as the one proposed in this paper. First, the algorithm allows tracking the position of the tools and estimate its trajectory, in order to compare the movements of an expert physician to a non-expert for training and for assessing the quality of the surgeries, as introduced by Blum et al. (2007). Secondly, the pose estimation of the articulated object is a particularly crucial step for further applications as automatic grasp-counting. Note that the number of grasps is fundamental in retinal surgery (Pavlidis et al., 2015) and should be minimized because the tissue can be easily damaged. Third, we consider also the usability of the microscope. Since the physician has to operate with both ocular and food pedal to navigate the microscope options, smoothing the workload can lead to quicker and better surgical result, as specific movements of the tool can be linked with the (de-)activation or modification of functionalities such as zooming, lighting or iOCT automatic positioning (Rieke et al., 2016). Moreover, the position of the tool is also the missing link to advanced augmented reality applications, such as providing the surgeon with additional information regarding the proximity of the tool tip to the retina. In Roodaki et al. (2015), the distance of the instrument from the retina can be calculated with the help of a tool tracker, and can visually inform the operator in case of risk of retina damage. Importantly, for the integration into the surgical workflow, it is necessary that the tracking algorithm achieves real-time efficiency.

1.1. Benefit of the instrument pose in addition to the instrument position

The position of the instrument in real-time is a valuable information during RM. However, the surgeon's center of attention is usually close to the surgical tool tips, which are more challenging to detect due to its opening and closing movement. Providing the position of the instrument's tips in real-time rather than the position of the central joint (e.g., as done in the work of Li et al., 2014 or Sznitman et al., 2012) can pave the way for advanced computer-aided support. One example is the positioning of the intraoperative OCT (iOCT) during membrane peeling. Usually, the surgeon needs the distance and depth information of the layers of the retina at the point where the tool tips grasp the membrane. In the current workflow, the iOCT position has to be adjusted manually, which is time consuming and also increases the complexity of handling the various instruments during a procedure. The ability of extracting the location of the surgical tool tips allows carrying out the positioning of the iOCT in an automatic way (Rieke et al., 2016). Furthermore, additional information such as proximity information of the tool tips to the retina can be visualized close to the instrument tips, so that the surgeon does not have to switch between different visualization modalities. By knowing the pixel distance of the tool tips to the joint point of the forceps in an image, the physical distance can be inferred and characteristics of anatomical structures can be measured directly in the visual data. The location of the instrument parts relative to each other can also provide important information about the state of the surgical workflow. All these aims are not achievable by measuring only the location of the center joint of the forceps.

1.2. Contributions

Despite the importance of estimating the location of the tool tips, most existing methods recover only the center joint of the instrument and are tested on synthetic data or only on a small dataset of RM sequences. In this work, we go beyond phantom data and propose a method for real-time tracking and pose estimation of surgical instruments in *in-vivo* microsurgery images, which estimates not only the position of the forceps central joint, but also the position of the instrument tips. Preliminary results of this work appeared in Rieke et al. (2015). The algorithm is inspired by state-of-the-art computer vision approaches and handles the aforementioned difficulties by modelling the problem as two different tasks: tracking and pose estimation (see Fig. 1). First, the tracking algorithm reduces the considered image information to a rectangular region containing the tool tip. In the second step, the pose estimation algorithm estimates the location of the instrument parts inside the bounding box. Both algorithms employ random forest in order to cope with noisy and incomplete data which result from the various appearance and illumination changes, but rely on different image information. By combining these two different algorithms, we use both color and gradient information for predicting the positions of the instrument parts.

In contrast to the work of Rieke et al. (2015) where only the grayscale information was used, the tracker in this work is now based on the entire RGB space. Another main contribution is the introduction of a novel, extended dataset of *in-vivo* RM sequences, which allows us to perform various detailed experiments evaluating the performance of the proposed algorithm regarding generalization to different tool types and different background conditions. These experiments could not be carried out extensively with the original dataset presented in Rieke et al. (2015), as it included only one sequence per tool type. In addition, we compared the performance of our method on this novel dataset to two methods: the online tracker TLD and the offline learned FPBC for retinal microsurgery by Sznitman et al. (2014). Due to the variations in instrument size in our new dataset, the performance of the algorithm is no longer comparable across sequences via the standard

performance measure which is based on pixel distances. Therefore, we also introduce a new metric for evaluating the prediction for the forceps joint which takes into account the variation of instrument shapes and different image resolutions. Furthermore, an additional contribution is the extension of the annotations on a public available laparoscopic dataset to pose information. Thereby, we can compare our algorithm to state-of-the-art methods on two published dataset – the RM sequence dataset and the laparoscopic instrument sequence.

2. Related work

Despite recent advances, the vision-based tracking of *in-vivo* sequences remains challenging – the strong illumination changes and the noise level of the images are the most prominent difficulties while the various appearances changes of the surgical instrument complicate the task further.

Prior work addressing these challenges has considered the use of geometric models such as (Baek et al., 2012) proposed an approach to track the forceps by generating a database of the projected contours of a 3D CAD model of the robotic forceps. The likelihood to the projected contour of the microscopic image is measured and finally the full state of the forceps is estimated via particle filtering. They evaluate this approach and demonstrate its robustness and efficiency on synthetic data of a simulated surgical environment. Reiter et al. (2012) presented a tracking method, which relies on the appearance of natural landmarks. They trained an efficient multi-class classifier and as the location of the natural landmarks are known in the tool's CAD model, they are used to compute the final pose. The algorithm was tested on five endoscopic sequences. Color-based approaches were presented by Allan et al. (2013, 2015) for the related field of laparoscopic tool tracking. Other relevant work (e.g. Richa et al., 2011) presents results on both phantom and *in-vivo* data, using a two stage procedure: brute-force search of the tool tip in the surroundings of the instrument coordinates in the previous frame; and, weighted mutual information to optimize the initial guess.

Most recent works build on learning based approaches like (Chen et al., 2013) who use natural features of surgical instruments for tracking and adopt a spiking neural network to recognize the instrument tip in laparoscopic surgeries. (Li et al., 2014) proposed an online learning approach for tool tracking in RM. The system starts with a manual initialization and gradually builds the database for tracking by adding new positive and negative tool samples, which are collected by a filtering process. The algorithm provides an accurate bounding box around the forceps' central point, but does not localize the two tips of the instrument. In Pezzementi et al. (2009), a phantom is employed, using a half-sphere, painted to resemble the retinal surface. This learning-based method is based on creating a model by hand-segmenting the instrument, where experiments have shown that usually one or two frames are sufficient. Rigid tool tracking is performed over two steps: first via appearance modelling, which computes a pixel-wise probability of class membership (foreground/background), then filtering, which estimates the current tool configuration. The proposed method of Sznitman et al. (2011) utilizes a parametrization of the surgical tool considering the following three criteria: the location of the insertion point, the angle between image boundary and tool, and the tool length. Afterwards, tracking is considered as a Bayesian filtering estimation problem. To compute the necessary posterior distribution, they use a strategy based on active testing (ATF). Their dataset consists in two sequences on a retinal phantom using a needle. In their most recent work, (Sznitman et al., 2014) proposed a robust and efficient algorithm which uses a multi-class classifier based on boosted regression trees. Each class represents a different part of the instrument (e.g. center, insertion point, shaft) or background. In order to provide both accuracy and good frame rate, an early stopping method was also implemented using a probabilistic model, which evaluates the reliability of current classification, and stops in case no more computation is necessary.

Many different works have been proposed in the RM field, but most of them are constrained and cannot uphold to real-world applications. For instance, some methods (Baek et al., 2012; Sznitman et al., 2011; Pezzementi et al., 2009) are only evaluated on synthetic data. In others, the (CAD-) model of the surgical instrument was given (Reiter et al., 2012). But this is not possible in RM because the tool are often changed and 50+ models are available on the market. Moreover, in Pezzementi et al. (2009), the instrument needs to be hand segmented in the first frame, and we fear this approach can increase surgery time. The works of Sznitman et al. (2011); 2014) use the insertion point as parameter to define the tool. However, in many cases when evaluating on our novel dataset (see Section 5.2), the point is not well-defined and their performance showed it to be the most challenging identifiable point of all classes. Allan et al. (2015) utilize the optical flow technique, which did not provide reliable results on the novel dataset due to the strong illumination changes in RM. Moreover, most of the works are not focusing on the exact coordinates of the tool tips (e.g. Richa et al., 2011; Sznitman et al., 2012; Li et al., 2014; Sznitman et al., 2014), which is a crucial step as discussed in Section 1.1.

3. Method

In this section, we present details of the proposed algorithm. The overall goal is the location of the instrument parts for every frame of an *in-vivo* RM sequence in real-time. As previously mentioned, due to the challenging nuisances normally present on the images, the independent tracking of the tool parts proves to be difficult. Furthermore, appearance changes of the instrument and strong illuminations variations result in incomplete and noisy data. Random forests (see Section 3.1) have shown to be able to handle these problems and even generalize to unseen situations. Therefore, we propose an algorithm relying on random forests, but tackling the task by means of two separate steps, so to focus the algorithms in exploiting different available information that are essential to solve the distinct problems in tracking and pose estimation. The overall pipeline is as follows (see Fig. 2): we assume, as input, an RGB-valued image $I : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3, p \rightarrow I(p)$ along with the initial localization of the tool. For every new frame I_t, the tracking algorithm (see Section 3.2) estimates the transformation that updates the location of a bounding box $I_B \subset I_t$ containing the entire instrument tip. As a RGB-based frame-to-frame tracker, it exploits spatio-temporal information of the template in the previous frame and relies on the contrast between the instrument and the retina. The pose estimation (see Section 3.3) then regresses the locations of the points of interest within this region I_B, which define the articulated pose of the instrument. In contrast to the tracker, it employs gradient information and is completely independent from the results of previous frames. The specific details of our algorithm will be given in the following sections.

3.1. Random forest

A crucial part of our method is the random forest, which is a machine learning method used in the tracking and in the pose estimation stage of our algorithm. Considering an input **X** and output **Y**, the random forest is used to learn the relation of **X** and **Y** such that, given the input **X**, the forest can predict the output **Y**. A forest itself consists of an ensemble of decision trees which can output a class (classification) or real numbers (regression) as in our case. In particular, random forests are used to correct the tendency of trees

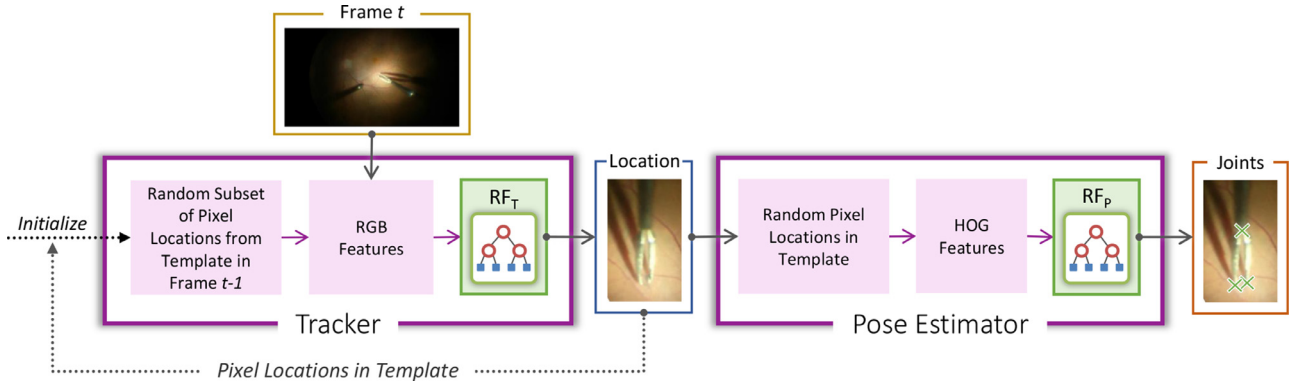


Fig. 2. Information processing: The figure illustrates the information processing of the algorithm. The algorithm relies on random forest and addresses the problem in two stages: tracking and pose estimation. The tracking random forest employs RGB intensity information whereas the pose estimation forest uses HOG features for estimating the locations of the joints.

to overfit the training set. Moreover, while predictions of a single tree are highly sensitive to noise in the training set, the average of many trees is not, under the hypothesis that they are independent. Each classifier at each node of the tree represents a "weak learner", while the ensemble of all such weak classifiers make the forest more confident to predict $\mathbf{Y}$. Further information on random forest can be found in the original work of Breiman (2001).

A tree is composed of nodes that can either be a branch which has two children (i.e., left and right) or a leaf which is the terminal node. Training a tree requires a learning dataset $P = \{(\mathbf{X}_d, \mathbf{Y}_d)\}_{d=1}^{n_d} = \{(\mathbf{X}_1, \mathbf{Y}_1), (\mathbf{X}_2, \mathbf{Y}_2), \ldots, (\mathbf{X}_{n_d}, \mathbf{Y}_{n_d})\}$ with the input $\mathbf{X}_d$ and its corresponding ground truth observation $\mathbf{Y}_d$. During *training* at each node, we split the data into two subsets to be passed down to its children (P_l, P_r), based on a splitting criterion θ. The splitting criterion is a pool of random tests and has the goal, at each step, to find the best split of the set. In our work, we employ the information gain for evaluating the best split, which is given as

$$g(\theta) = H(P_n) - \sum_{i \in \{l,r\}} \frac{|P_i(\theta)|}{|P_n|} H(P_i(\theta)), \tag{1}$$

where P_n is the set of samples that reached the node n, $|P|$ is the number of samples in the set P and $H(\cdot)$ evaluates the randomness of P. Since we consider regression forest, $H(\cdot)$ can be estimated by standard deviation of the multi-variable Gaussian distribution. Starting from the root node, the dataset is iteratively split into two subsets and passed down to the node's children until one of the following stopping criteria is true:

1. the maximum depth of tree is reached;
2. the number of samples that reached the node is insufficient to split; or,
3. the information gain of the best split is too small.

Then, the leaf stores the distribution of the parameters of $\mathbf{Y}$ that typically employ a normal distribution with its mean and standard deviation. As a result of learning, each branch of the tree stores the parameters of the splitting function with respect to the input $\mathbf{X}$ while each leaf stores a distribution of the output $\mathbf{Y}$. To enfoce the independence of the trees in the forest, each random tree selects a random subset of elements in $\mathbf{X}$ or a random subset of the learning dataset.

During *testing*, a new sample of $\mathbf{X}$ traverses the tree. At each branch, it moves to the left or right child node depending on the splitting function, eventually ending up at a leaf node which contains the prediction to be associated with such sample. Finally, the results of the leaf nodes at different trees are aggregated in order to robustly obtain the final prediction.

3.2. Tracker

In this work, a template is defined as a rectangular region that encloses the three keypoints at the tip of the tool which are used for the pose estimation. The region is axis-aligned to the shaft and the tool tip is on the upper third of the bounding box. In this way, the rectangular region is large enough such that all the keypoints are visible for the pose estimation. In practice, the tracker is initialized by enclosing a bounding box around the tool on the first frame of a given video sequence. The objective then is to propagate the transformation parameters from one frame to the next in order to keep track of the region of interest.

Mathematically, a template is described by the RGB intensity values at n_s sample points, written as $\{\mathbf{x}_s\}_{s=1}^{n_s}$, which are 2D points that are randomly selected within the rectangular region. Thus, given a sequence of images $\{I_t\}_{t=0}^{n_t}$, the tracker determines the transformation $\mathbf{T}_t$ of the sample points and, in effect, locates the rectangular region such that the tool tip is enclosed in the bounding box. Based on this, the tracker uses the random forest to learn the relation of the RGB intensity vector at the sample point locations of the previous frame $\mathbf{X} = [I_t(\mathbf{T}_{t-1} \cdot \mathbf{x}_s)]_{s=1}^{n_s}$ and the corresponding translation vector $\mathbf{Y} = \delta\boldsymbol{\mu}$ that aligns the bounding box at the position of the tool tip through

$$\mathbf{T}_t = \mathbf{T}_{t-1}\mathbf{T}(\delta\boldsymbol{\mu}). \tag{2}$$

The algorithm is inspired by the work of Tan and Ilic (2014). Similar to them, our tracker runs at less than 2 ms per frame with a single CPU core. The fast tracking time with a very small computational cost is the primary advantage of this tracker.

In general, the cost function of most template tracking algorithms (e.g. Baker and Matthews, 2004; Jurie and Dhome, 2002; Holzer et al., 2012; Tan and Ilic, 2014) follow the pixel-wise difference

$$E(\mathbf{x}_s) = \|I_t(\mathbf{T}_{t-1} \cdot \mathbf{x}_s) - I_{t_0}(\mathbf{x}_s)\| \; \forall \mathbf{x}_s, \tag{3}$$

derived from image registration where $I_{t_0}(\mathbf{x}_s)$ is the given template. Based on Tan and Ilic (2014), they converted Eq. 3 as the feature vector $\mathbf{X} = [I_t(\mathbf{T}_{t-1} \cdot \mathbf{x}_s) - I_{t_0}(\mathbf{x}_s)]_{s=1}^{n_s}$ of a random forest. In tracking, each node of the tree thresholds one element of $\mathbf{X}$ to determine whether to traverse to the left or right child until a leaf is reached. Due to the given $I_{t_0}(\mathbf{x}_s)$, both the cost function and feature vector illustrate the limitation of these methods to track a single template. However, we observed that, when using pixel-wise splitting, $I_{t_0}(\mathbf{x}_s)$ is always constant for each element of the feature vector. Thus, we can incorporate $I_{t_0}(\mathbf{x}_s)$ as part of the threshold and simplify the feature vector as $\mathbf{X} = [I_t(\mathbf{T}_{t-1} \cdot \mathbf{x}_s)]_{s=1}^{n_s}$. In this formulation, the tracker learns and tracks based on the intensity values instead of the intensity difference. As a consequence, we can

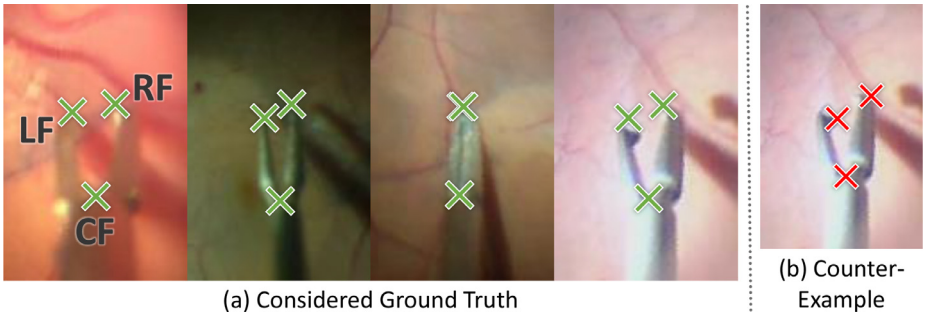


Fig. 3. Considered ground truth in instrument dataset: Due to the variation of instrument shapes, the ground truth has to be set in more detail in order to be well-defined. (a) shows examples for the considered ground truth. The tool tips (*LF* and *RF*) are set as the points on the tip which are closest to the microscope and would touch in case of a closed forceps. The center joint (*CF*) is the point connecting the two parts of the forceps. In (b), the annotation is not selected as the top visible part the tool tip points but on the part which is closer to the retina.

take a step further to alleviate the limitation of learning only one template and learn based on the intensity values of multiple templates. Therefore, in contrast to other template tracking algorithms where they learn and track an individual template, we generalize our algorithm to utilize multiple correlated templates to handle the visual changes due to the articulated deformation of the tool and different instrument structures, and to be robust against various environmental factors such as illumination changes and photometric distortions that are common in such working conditions as shown in Fig. 3.

Learning. Considering that the algorithm is a temporal tracker, it predicts the update parameters that refines the location of the tool from the previous frame to the current frame through Eq. 2. The forest then learns to predict the movement from an erroneous position of the tool to its ground truth. To create the learning dataset of the forest, we enforce this movement by randomly transforming the template from its ground truth location. Given n_i templates to learn and the individual ground truth transformation $\mathbf{T}_i$ of the ith template, we impose n_r random transformations on the ith template by transforming the sample points by $\mathbf{T}_i\mathbf{T}_r^{-1}$ for all $\{\mathbf{T}_r\}_{r=1}^{n_r}$ where $\mathbf{T}_r = \mathbf{T}(\delta\boldsymbol{\mu}_\mathbf{r})$. The objective of introducing $\mathbf{T}_r^{-1}$ is to mimic the transformation from the previous frame such that a transformation of $\mathbf{T}_r$ updates the position of the template from an erroneous location $\mathbf{T}_i\mathbf{T}_r^{-1}$ to its ground truth location as $(\mathbf{T}_i\mathbf{T}_r^{-1})\mathbf{T}_r = \mathbf{T}_i$.

As a consequence, each random transformation generates a combination of samples and labels written as $(\mathbf{X}_r,\ \mathbf{Y}_r) = ([I_i(\mathbf{T}_i\mathbf{T}_r^{-1}\cdot\mathbf{x}_s)]_{s=1}^{n_s}, \delta\boldsymbol{\mu}_r)$, which are accumulated as the set $P = \{(\mathbf{X}_d, \mathbf{Y}_d)\}_{d=1}^{n_i\cdot n_r}$ that are used to learn one forest per transformation parameter. Our goal in learning is to divide the learning dataset P, as the tree gets deeper, into subsets with similar transformation parameters such that, in tracking, the subset is located and uses the mean of the parameters as the prediction.

When learning a tree, each node splits the the learning dataset into two subsets which are inherited by the left and right child. Given the subset of the learning dataset P_n that arrived on the node n, an index β of the vector $\mathbf{X}_r$ is selected to threshold the values across the dataset. In order to find the best split, multiple indices and thresholds are tested and the selection of the best pair is measured based on the information gain in Eq. 1 where

$$H(P) = \frac{1}{|P|}\sqrt{\sum_{\mathbf{Y}_d\in P}\|\mathbf{Y}_d - \delta\bar{\boldsymbol{\mu}}(P)\|^2} \tag{4}$$

is the standard deviation of the transformation parameter and

$$\delta\bar{\boldsymbol{\mu}}(P) = \frac{1}{|P|}\sum_{\mathbf{Y}_d\in P}\mathbf{Y}_d \tag{5}$$

is the mean vector of all parameters in P. This implies that the result of the split is two subsets with a more homogeneous transformation parameter.

With the splitting of P enforced in each node, the tree continuously grows until one of the stopping criteria in Section 3.1 is satisfied. Consequently, the node is considered as a leaf and stores the mean and standard deviation of the parameters based on the subset of the learning dataset that arrives on the node, which is similar to Eqs. 4 and 5. Here, the mean from Eq. 5 is the predicted transformation parameter in tracking while the standard deviation from Eq. 4 acts as a weight that measures the homogeneity of the parameters within the subset.

Tracking. When evaluating the trees in tracking, $\mathbf{X}$ is computed using the transformation from the previous frame. Through the splitting function at the nodes of a tree, $\mathbf{X}$ manoeuvres from the root node to a leaf where the mean and standard deviation of the predicted parameter is stored. Considering the possibility that, when learning the tree, some subsets of the learning dataset does not converge to a homogeneous transformation parameters, only the 15% of the best predictions multiple trees of the forest with the lowest standard deviation are aggregated in finding the parameters. Using Eq. 5, the best predictions are aggregated by computing the average parameter $\delta\bar{\boldsymbol{\mu}}$. Thereafter, the predictions constructs $\mathbf{T}(\delta\bar{\boldsymbol{\mu}})$ and updates the transformation with Eq. 2 from $t-1$ to t. Notably, the forest is used iteratively to refine the previous estimate.

Since the standard deviation measures the confidence of the predictions, we deem tracking successful (i.e., the tracker converges to a confident solution) if the average standard deviation in the final iteration is less than a threshold; otherwise, we avoid updating the transformation parameters and utilize the previous location of the tool for the succeeding frames.

3.3. Pose estimation

The final task of the presented method is the localization of the instrument parts in every frame, which similarly to the tracking stage, also has to be carried out in real-time. The main idea behind our approach is to interpret the surgical tool as an articulated object and to employ parametric models similar to those successfully proposed in the field of human pose estimation (as can be seen in (Belagiannis et al., 2014). Specifically, by defining the set of joints as the left tip of the instrument (LF), the right tip of the instrument (RF) and the center joint (CF) connecting these two parts, we can integrate this methodology in our approach. Since we investigate the performance of the algorithm on different instrument shapes, we have to emphasize that the points on the tips are defined as the inner-most and top visible point of the part, which

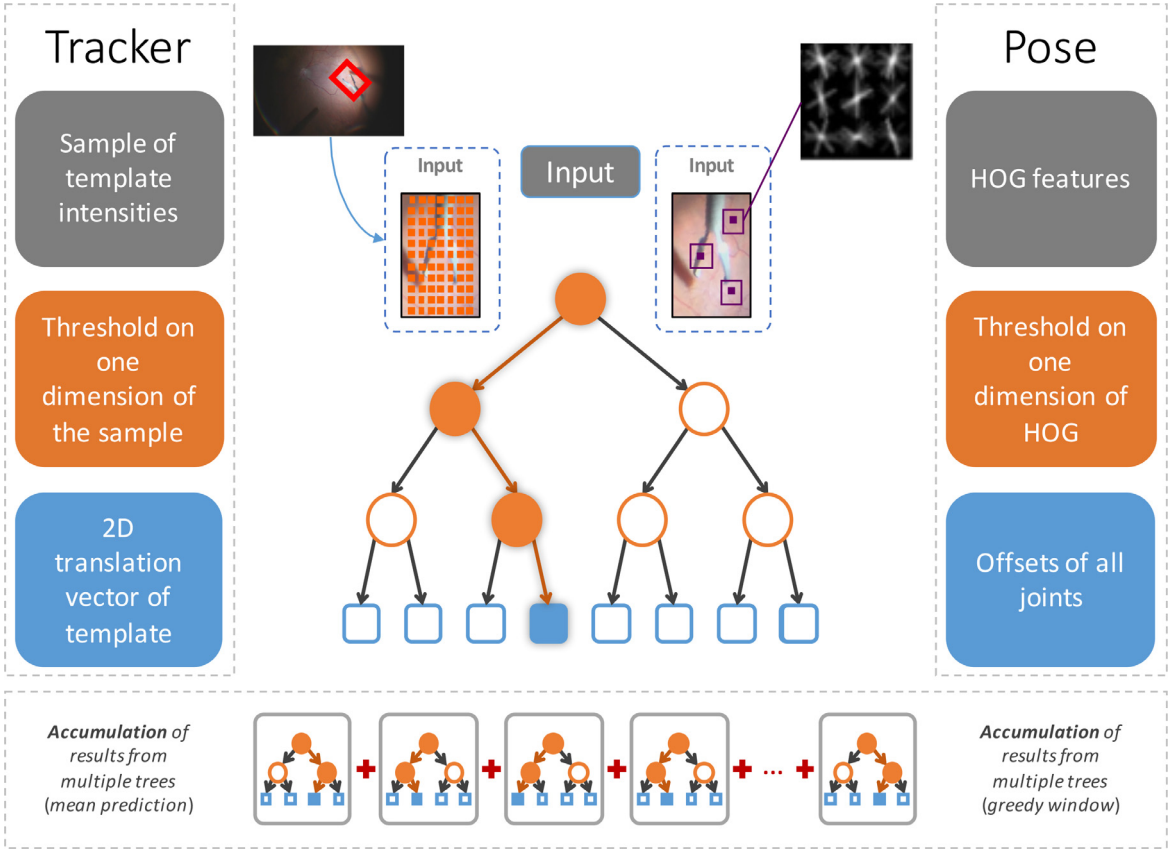


Fig. 4. Overview random forests: The input **X** for *Tracker* are the intensity values of the current frame I_t at the sampling positions x_s from previous frame I_{t-1}. The binary split function θ divides the samples by thresholding on one dimension of **X**. The output **Y** is the 2D translation of the template I_B. Finally, the results are accumulated by taking the mean of the predictions with the smallest standard deviation from the forest of a parameter. For the *Pose Estimation*, the input **X** consists of the HoG features extracted at randomly selected points within the bounding box $q \in I_B$ and the binary test θ is performed on one dimension of the HoG feature. The output **Y** is the offset of the joints of the instrument. The final estimation is aggregated by a greedy dense-window algorithm.

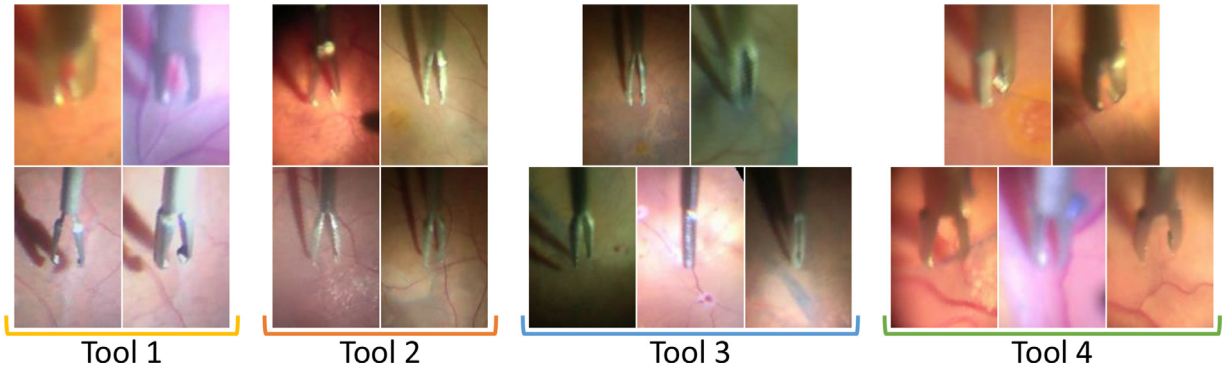


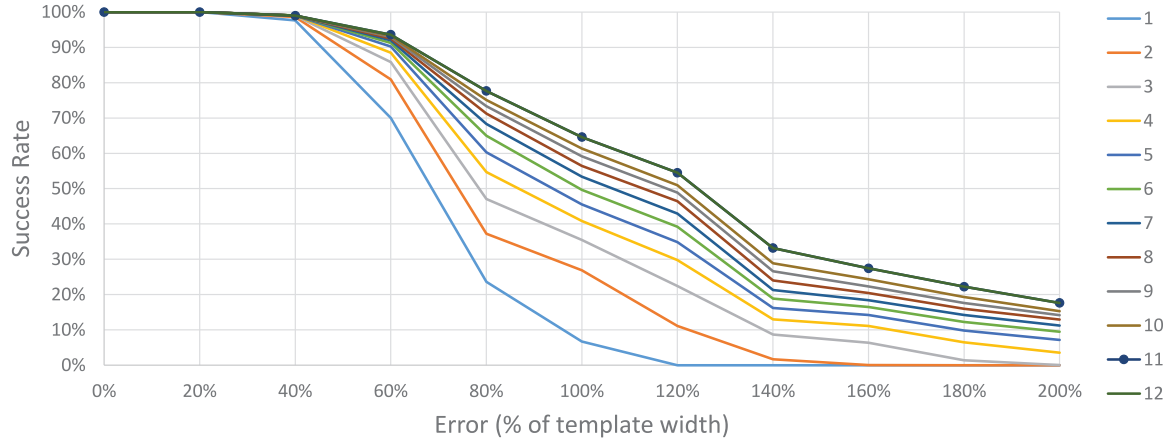
Fig. 5. Instrument dataset representative frames: Different types of instruments are present in our dataset. They show different shapes, for example **Tool 1** and **Tool 4** are bulky, the first being more rounded close to the center joint. **Tool 2** and **Tool 3** are smaller, with Tool 3 showing a extremely thin attachment to the shaft. In addition, different illuminations, reflections, blur and colouring can be observed.

tend to touch each other in case of a closed forceps (see Fig. 3), in order to have a well-defined ground truth.

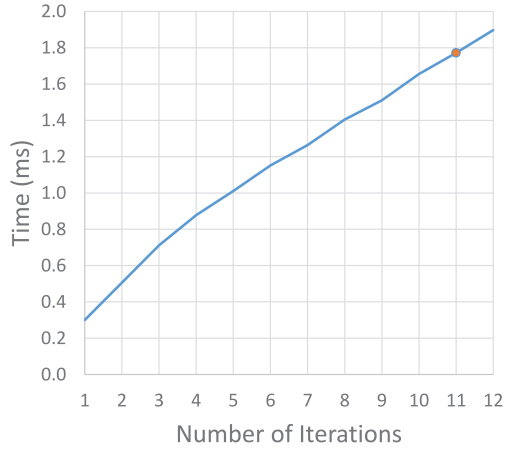
The goal is now to infer the location of the instrument parts from the extracted image features. In contrast to part-based methods, the holistic approach aims at predicting the joints at one step. Instead of considering the image information of the entire frame, the tracker simplifies the problem by limiting the region of interest to a bounding box $I_B \subset I_t$, and therefore drastically reduces the computational cost. This is an important observation since in this second step of the pipeline, we make use of the computationally more expensive gradient information, which tends to be highly reliable in these kind of challenging scenarios. More precisely, we employ Histogram of Oriented Gradients (HoG) features (Dalal and Triggs, 2005), which have shown their robustness in fields such as object detection (Felzenszwalb et al., 2010), image retrieval (Eitz et al., 2011) and classification (Nilsback and Zisserman, 2008). Here, a key aspect is that the tracked template as defined in Section 3.2 yields a bounding box around the tool tip which is aligned with the direction of the tool shaft at the time of the initialization. During the tracking, only the translation parameters are updated and I_B is not necessarily aligned with the instrument shaft any more. However, the insertion point of the

Table 1
Overview of the novel dataset introduced in this work. Representatives of the Tool Types are shown in Fig. 5. Example frames of every sequence are displayed in Figs. 8, 9, 10, 11. For determination the average RGB color, all pixels in the range (0,0,0) to (10,10,10) were excluded from the computation. Significant scaling is present, if the ratio of minimum size to maximum size of instrument tip is higher than 0.5. Translation is considered as significant, if the overall translation distance of the central joint is higher than 1000 pixels.

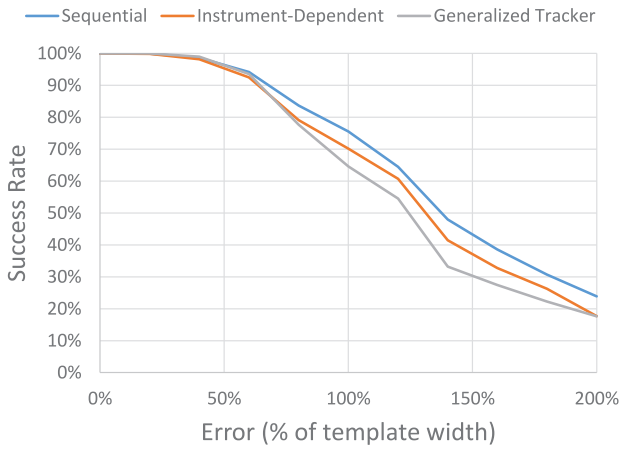
Sequence	Resolution (pixel)	Tool type	# open-close	Average RGB color	# frames	Average (*h*, *w*) for KBB (pixel)	Lightsource in focused area	Significant Scaling	Significant Translation
1	1920 × 1080	Tool 1	2	[103, 59, 35]	200	(53,70)	✓	✓	✓
2	1920 × 1080	Tool 1	3	[74, 50, 70]	200	(83,43)	✓	✓	✓
3	1920 × 1080	Tool 1	2	[68, 57, 55]	200	(67,82)	×	×	✓
4	1920 × 1080	Tool 1	4	[84, 68, 66]	200	(100,115)	✓	×	✓
5	1920 × 1080	Tool 2	2	[91, 41, 31]	200	(98,178)	✓	✓	✓
6	1920 × 1080	Tool 2	2	[71, 55, 31]	200	(131,79)	✓	✓	✓
7	1920 × 1080	Tool 2	3	[26, 29, 37]	200	(87,33)	✓	✓	×
8	1920 × 1080	Tool 2	3	[32, 45, 61]	200	(126,69)	×	✓	✓
9	1920 × 1080	Tool 3	2	[65, 48, 34]	200	(226,74)	×	×	✓
10	1920 × 1080	Tool 3	2	[52, 53, 30]	200	(94,31)	×	×	×
11	1920 × 1080	Tool 3	2	[49, 48, 27]	200	(121,60)	✓	×	✓
12	1920 × 1080	Tool 3	1	[100, 68, 67]	200	(173,109)	✓	×	✓
13	1920 × 1080	Tool 3	3	[60, 44, 36]	200	(104,91)	✓	✓	✓
14	1920 × 1080	Tool 4	2	[133, 77, 55]	200	(104,173)	✓	✓	✓
15	1920 × 1080	Tool 4	1	[83, 50, 30]	200	(77,141)	✓	✓	✓
16	1920 × 1080	Tool 4	3	[123, 62, 46]	200	(76,98)	✓	✓	✓
17	1920 × 1080	Tool 4	2	[96, 57, 74]	200	(93,61)	✓	✓	✓
18	1920 × 1080	Tool 4	1	[115, 73, 54]	200	(93,130)	✓	×	✓



(a) Number of Iterations



(b) Tracking Time



(c) Different Learning Dataset

Fig. 6. Parameter evaluation for the tracker: The experiment was evaluated on every sequence of the Instrument Dataset (IDS). The depicted results show the success rate of the tracker and the timings associated with them. With 100 trees, we evaluate the number of iterations required to achieve convergence in (a) with its corresponding tracking time in (b). In addition, we compare the success rate when learning an increasing number of templates (i.e., 100 for sequential, 400-500 for instrument-dependent and 1800 for the generalized tracker).

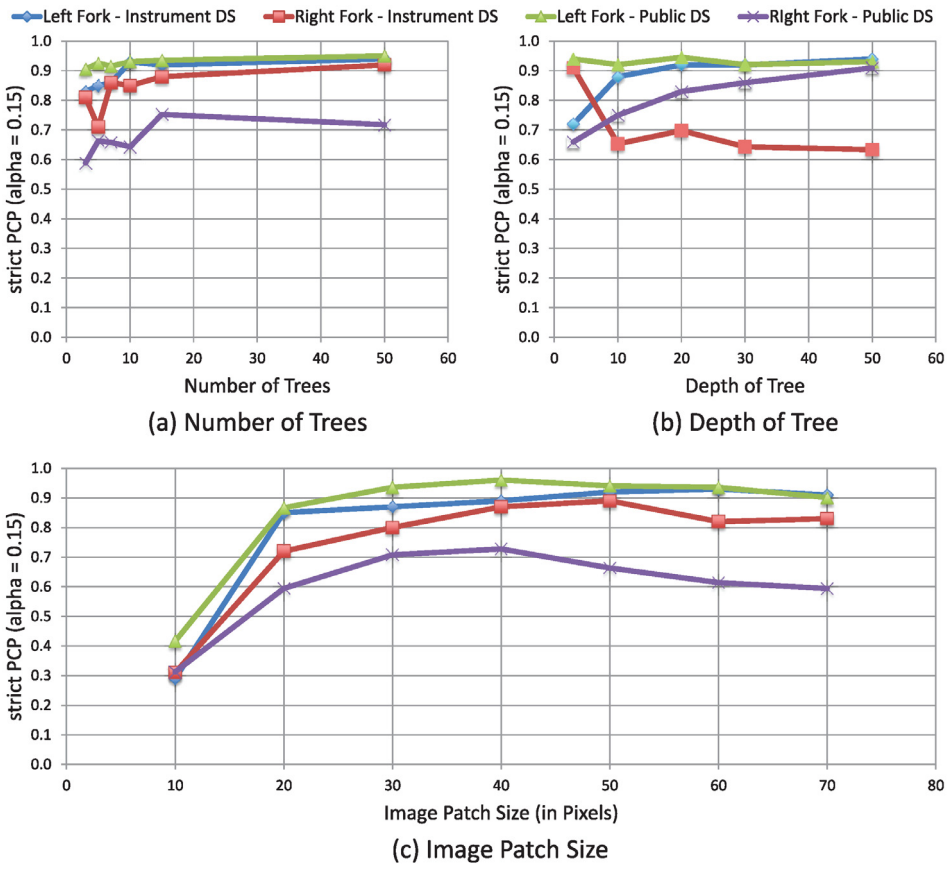


Fig. 7. Parameter evaluation for pose estimation: The experiment was evaluated on one sequence each of the Instrument Dataset (IDS) and Public Dataset (PDS). The depicted results are the strict PCP scores for the alpha value of $\alpha = 0.15$. It should be considered that in human pose estimation, a common choice is $\alpha = 0.5$. Therefore, the low α value in our case constrains that only very precise predictions are accepted.

instrument to the eye is fixed by trocars during a procedure and consequently the orientation of the tool shaft remains in a limited range. Therefore, we consider the set of templates as defined in Section 3.2 together with the ground truth annotation of the joints as base learning dataset and augment the data by applying n random similarity transformations with parameters in the range of $\pm$ 0.3 for the scale, $\pm$ 30 pixel for the translation in x and y direction and $\pm$ 30 degrees for the rotation from the ground truth homography. All images are rescaled to a fixed pixel size. This yields an extended dataset which improves the robustness of the pose estimation and tackles the problem that HoG features are not rotation invariant.

Within the bounding box I_B, the HoG features are computed on image patches around randomly selected points and are employed as an input $\mathbf{X}$ for the trees. The binary split function θ divides the input sample regarding a threshold on one dimension of the HoG feature.

The function $H(\cdot)$ is based on the Sum-of-Squared-Distances (SSD)

$$H(P) = \sum_{i \in I} \sum_{j} \| o_{i,j} - \mu_j \|_2^2, \tag{6}$$

where I denotes the image patch, the 2D vector $o_{i,j}$ contains the offset of the joint $j \in J$ from the image patch center and μ_j is the mean for each joint offset. The leaves store the corresponding offsets $o_j = \mathbf{Y} \subset \mathbb{R}^2$ of all instrument joints $j \in J = \{LF, RF, CF\} \subset \mathbb{R}^2$. In order to find the most probable outcome, the votes of the separate trees are accumulated by a greedy dense-window algorithm, similar to the work of Belagiannis et al. (2014). For this purpose, the 2D predictions for every joint $j \in J$ are discretized on a fixed grid, whereas the grid cells contain the number of votes that lie within it. To aggregate its votes, an integral matrix is created for every cell and all the cells form an integral image. Then, the final estimation corresponds to the region with the maximum number of points, which is found by sliding a window over the integral image.

In this way, a direct mapping between the extracted HoG features and the location of the instrument joints is modelled. Due to the characteristics of the random forest, this relation can also be inferred for unseen instrument poses or varying lighting conditions. An overview of the random forests is visualized in Fig. 4.

4. Material

4.1. Description of the datasets.

The experimental validation of the proposed algorithm is carried out on two different RM datasets: a new dataset, in the following called *Instrument Dataset* and the dataset published by Sznitman et al. (2012), in the following referred to as *Public Dataset*. For comparison, the performance of the algorithm was also evaluated on a published laparoscopic instrument sequence.

Instrument dataset: This consists of 18 sequences of *in-vivo* retinal surgery and is an extended version of the *appearance* dataset presented in the work of Rieke et al. (2015), which only contained 4 of the 18 sequences. The images are acquired by a Carl-Zeiss Lumera 700® operating microscope with a resolution of 1920 $\times$ 1080 pixels at 25 fps progressive scans with 24-bit RGB color format. For each sequence, we selected 200 subsequent frames in which the instrument is always visible and at least one

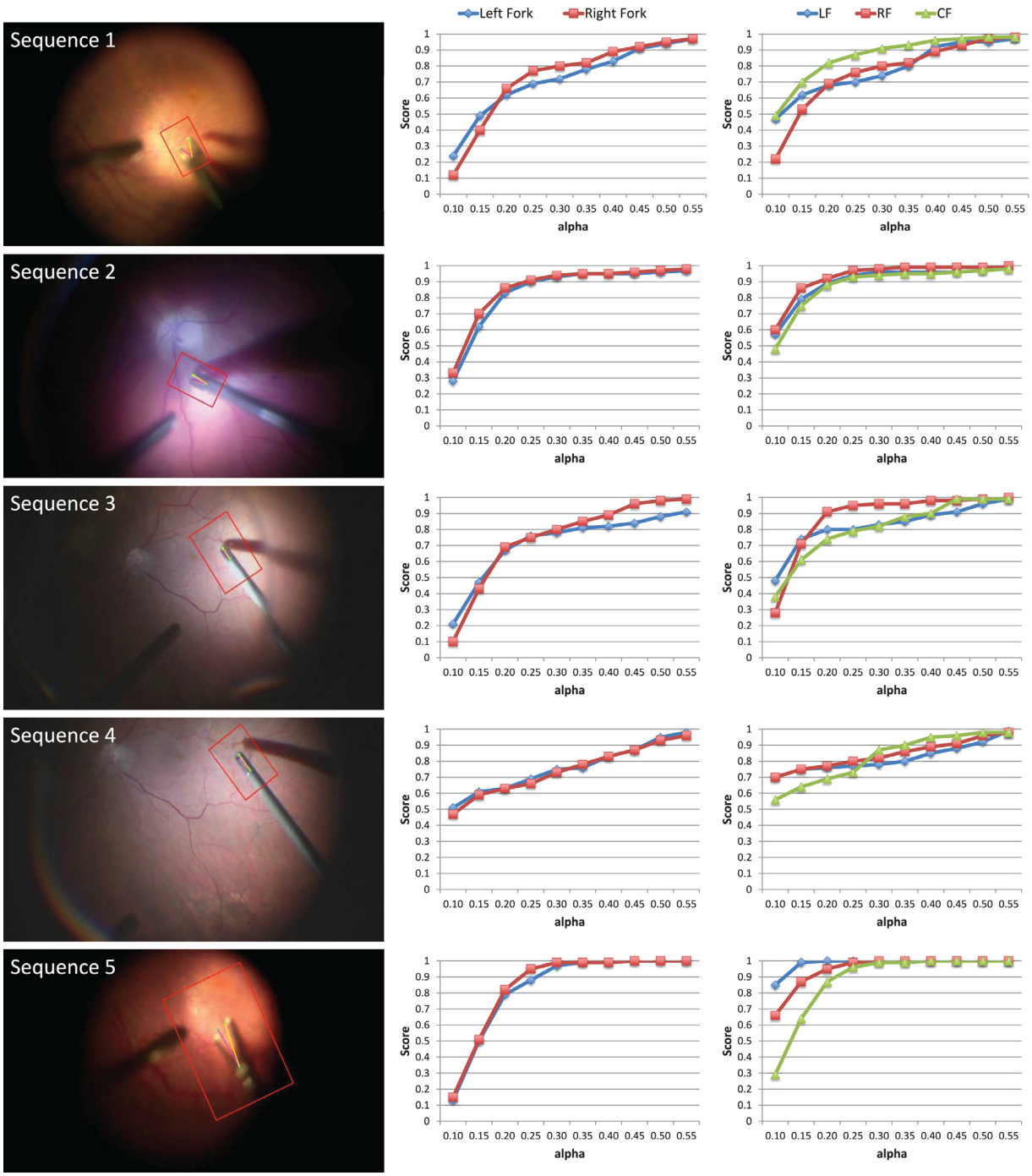


Fig. 8. Part I of the sequential evaluation of instrument dataset: Results for sequences 1 to 5. Left column: Example of the sequence. Middle column: strict PCP score for the left and the right fork. Right column: *KBB* score for evaluating the prediction of the keypoints.

movement of opening and closing is present. Each frame was annotated manually, following the definition of the ground truth given in Section 3.3. In comparison to the sequences in Rieke et al. (2015), the dataset was considerably extended and allows us to perform instrument dependent experiments. Furthermore, additional lightning variations and microscope zoom factors are present, increasing the complexity of learning. In total, four different types of instrument can be observed as depicted in Fig. 5. Therefore, depending on the type of tool present in the sequences, we divided the dataset in four smaller subsets, containing respectively 4, 4, 5 and 5 videos. An overview of characteristics of the sequences of the novel dataset can be found in Table 1.

Public dataset [1]**:** This is a fully annotated dataset of three different sequences of *in-vivo* vitreoretinal surgeries. It comprises of 1171 images with a resolution of 640 × 480 pixels with respectively 402, 222 and 547 frames for the first, second and third sequence. The main challenge of this dataset is variations of lighting as well as the presence of noise and shadows. Notably, the same instrument is utilized in all sequences. The key component of this dataset is the dominant blue and green colouring of the sequences, on which the dependence of an algorithm regarding its color reliance can be evaluated.

[1] https://sites.google.com/site/sznitr/code-and-datasets

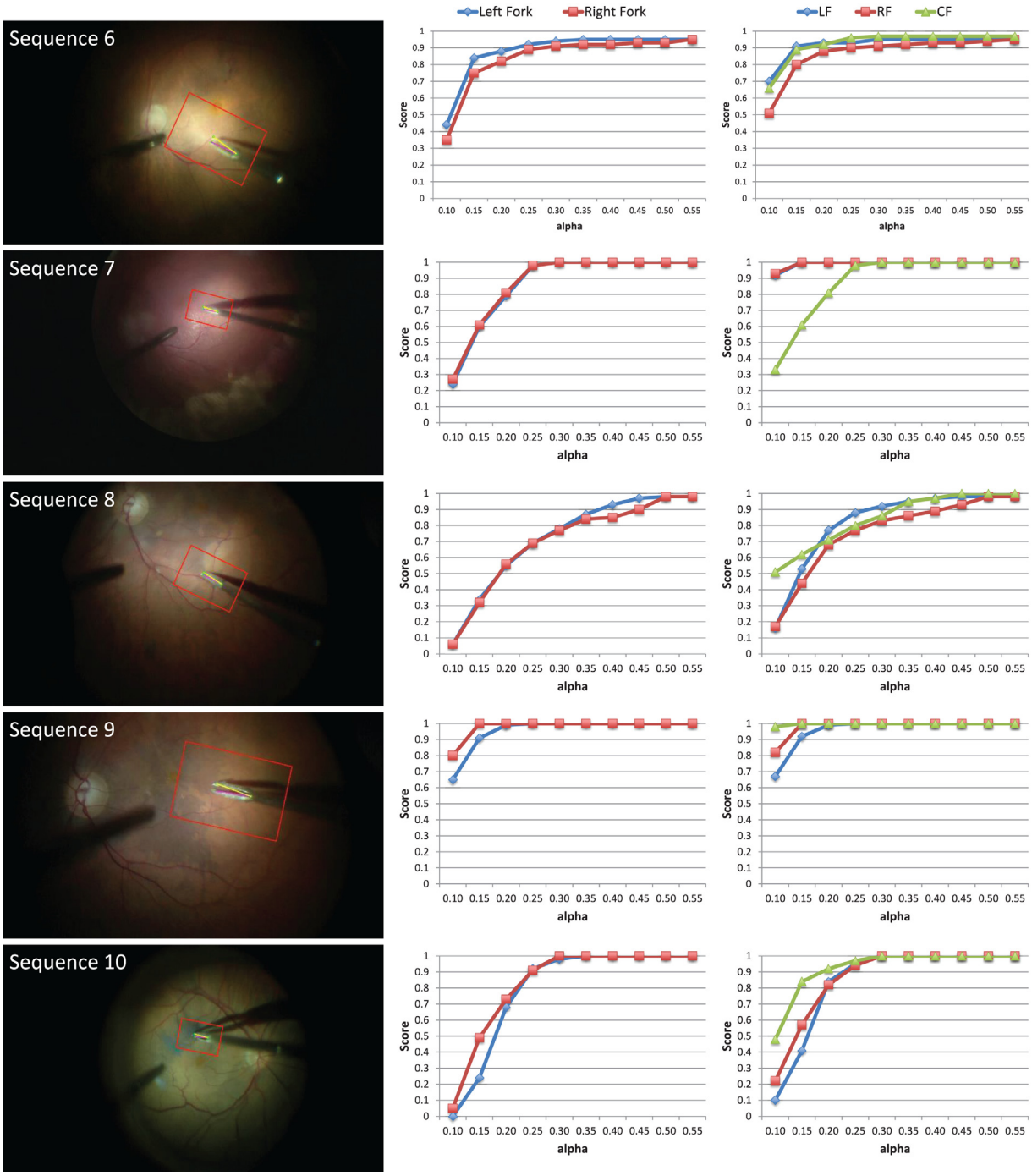


Fig. 9. Part II of the sequential evaluation of instrument dataset: Results for sequences 6 to 10. Left column: Example of the sequence. Middle column: strict PCP score for the left and the right fork. Right column: *KBB* score for evaluating the prediction of the keypoints.

Laparoscopic sequence[1]**:** This is an annotated and publicly available laparoscopic instrument sequence. It consists of 1000 frames and shows two surgical instruments. The location of the central joint is labelled for every visible instrument. We focus on the more challenging instrument (in previous works referred to as Tool 1 Li et al., 2014) and extend the provided labels by manually annotating the location of the tool tips. For pose estimation, the main difficulties are the partial occlusions when the instrument enters the tissue and the presence of smoke. Furthermore, the sequence is recorded with large variations regarding the distance between the instrument and the camera.

4.2. Description of the metrics

The performance of our method was evaluated by means of four different metrics which are presented in this section, including standard metrics and a newly proposed metric addressing the variation of the scales of the instruments and image resolutions in the Instrument Dataset.

Strict percentage of correct pose (strict PCP): This addresses the quality of the prediction for a part of an articulated object and is a standard metric in human pose estimation (Pickering et al., 2009). A prediction for a part connected by two joints $j_1, j_2 \in \mathbb{R}^2$

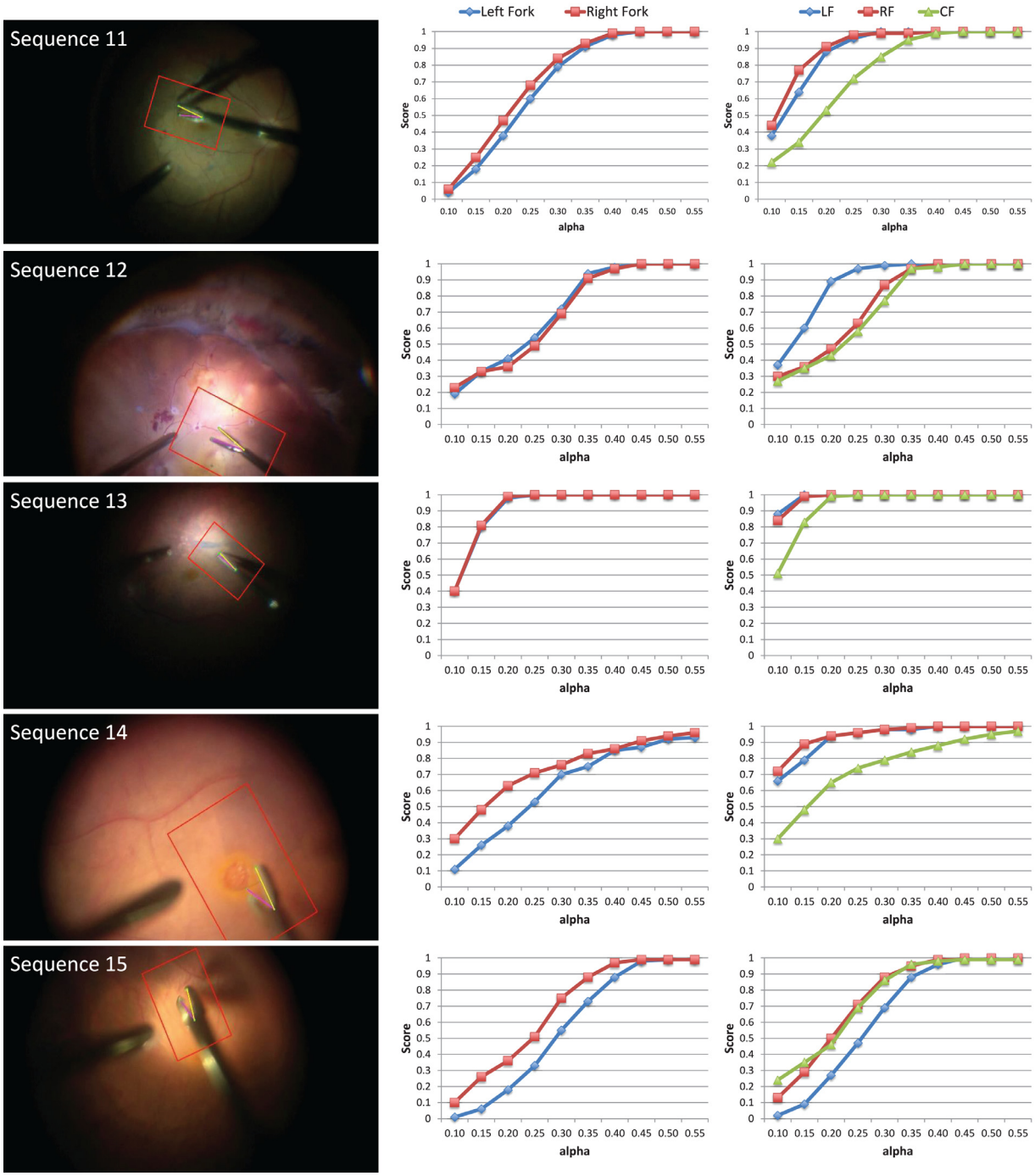


Fig. 10. Part III of the sequential evaluation of instrument dataset: Results for sequences 11 to 15. Left column: Example of the sequence. Middle column: strict PCP score for the left and the right fork. Right column: *KBB* score for evaluating the prediction of the keypoints.

is evaluated as correct only if both the euclidean distances of the predicted joints j_1, j_2 to its ground truths $\hat{j}_1, \hat{j}_2$ are lower than a threshold as a function of the ratio $\alpha \in \mathbb{R}$ times the ground truth length of the part, e.g. both of the following equations have to be fulfilled:

$$\|j_1 - \hat{j}_1\| < \alpha \cdot \|\hat{j}_1 - \hat{j}_2\|$$
$$\|j_2 - \hat{j}_2\| < \alpha \cdot \|\hat{j}_1 - \hat{j}_2\|.$$

For human pose estimation, the threshold value is usually set to $\alpha = 0.5$ (compare Pickering et al., 2009).

Keypoint Threshold (KT): This was employed by Sznitman et al. (2012) and addresses the quality of the keypoint predictions as a pixel-wise measure. Estimated joint locations $j \in \mathbb{R}^2$ are evaluated as correct if the euclidean distance to the ground truth annotation $\hat{j} \in \mathbb{R}^2$ is lower than a fixed pixel threshold $T \in \mathbb{R}$:

$$\|j - \hat{j}\| < T. \tag{7}$$

Therefore, it yields a separate evaluation for every keypoint $j \in J$.

Keypoint threshold bounding box (KBB): The *KT* metric indirectly assumes that the frames have the same resolution and show the same type of instrument. However, the selection of a reasonable threshold is difficult for different zoom factors and instru-

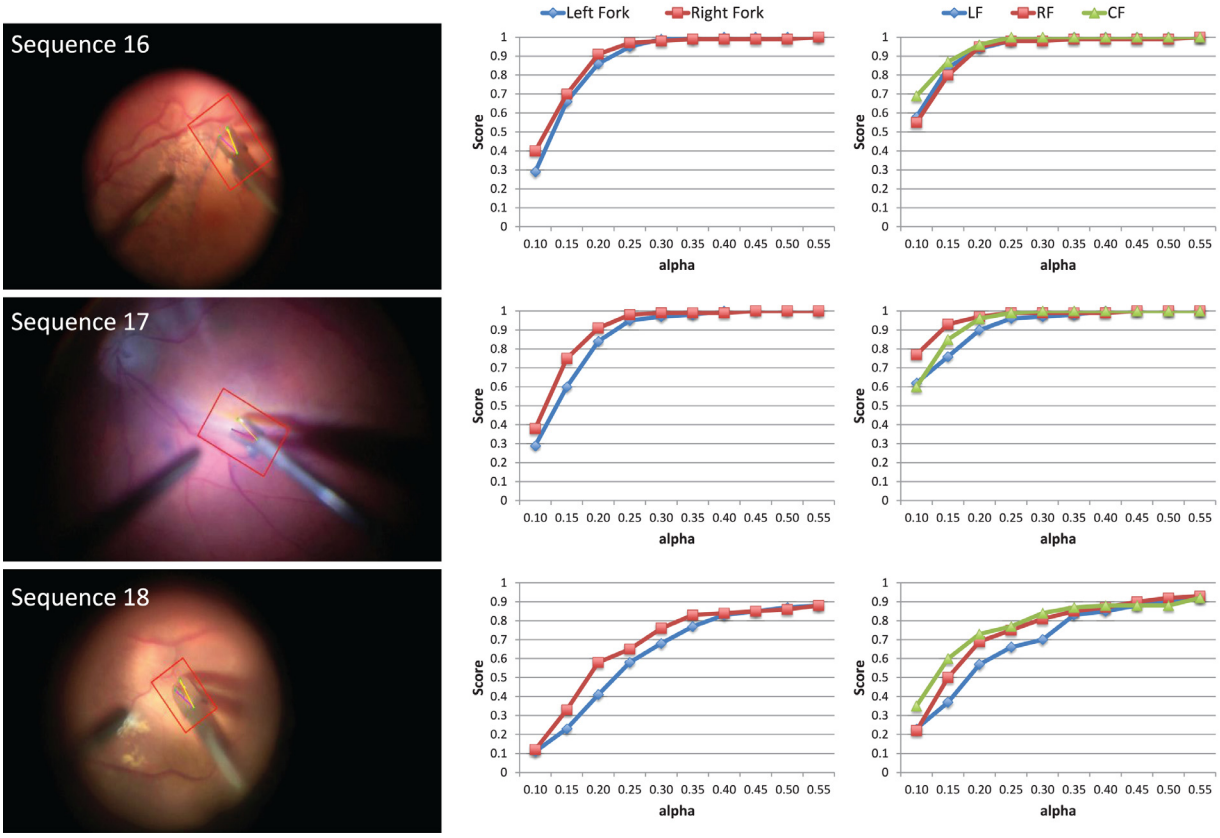


Fig. 11. Part IV of the sequential evaluation of instrument dataset: Results for sequences 16 to 18. Left column: Example of the sequence. Middle column: strict PCP score for the left and the right fork. Right column: *KBB* score for evaluating the prediction of the keypoints.

ments, leading to the problem that sequences are not directly comparable. Inspired by the metric introduced by Yang and Ramanan (2013) in the field of human pose estimation, we propose a novel metric for retinal microsurgery which addresses this problem. Instead of using a fixed pixel threshold, the accepted distance depends on the size of the instrument tip. In this way, a higher resolution of sequences and a change in the distance of the instrument from the retina does not automatically lead to a higher error for the keypoint evaluation. For this, we consider a tightly cropped, axis-aligned bounding box which contains all ground truth joints of the instrument in the respective frame. We define a joint j to be located correctly if

$$\|j - \hat{j}\| < \alpha \cdot \max(h, w), \tag{8}$$

where $\hat{j}$ is the ground truth annotation of the joint, w and h are the width and height of the bounding box around the instrument given by the ground truth, and $\alpha \in \mathbb{R}$. It should be noted that this metric is only computable if the ground truth of all joints is given. However, the evaluation of a keypoint is pose-independent and also applicable if only the joint point *CF* is estimated.

Success rate of the tracker: The tracker is evaluated by inducing random translation to the template to simulate the displacement of the bounding box from the previous frame to the current frame. Apart from the standard frame-to-frame tracking that is used to find the bounding box for the pose estimation, we also introduce this synthetic evaluation to numerically determine the range of translation error, which the tracker can handle. In this case, the maximum translation error is parameterized with respect to the percentage of the template's width. Here, a successfully tracked template is determined by asserting that all three joints must be within the bounding box, which is defined to be relatively tight around the tool tip. After applying the synthetic evaluation across multiple images, it follows that the success rate is defined as the percentage of successfully tracked templates over the total number of tests.

5. Experiments and results

In this section, we present the results of the experiments that are performed on the three different dataset presented in Section 4.1 and evaluated in terms of the metrics described in Section 4.2. First, the influence of the parameters for both the tracker and pose estimation is investigated (Section 5.1). We gradually evaluate the generalizability of our algorithm to unseen conditions in Section 5.2. The performance of the proposed method is compared to state-of-the-art methods on RM sequences in Section 5.3 and on a laparoscopic sequence in Section 5.4. The method is implemented in C++ and runs at 30 fps on an off-the-shelf computer.

5.1. Parameters experiments

For both the tracker and the pose estimation, several parameters have to be set during training. For this purpose, we evaluated the performance of the two algorithms independently as described in the following.

Parameter for tracker: Considering that the tracker is an iterative method, we evaluate its performance with respect to the number of iterations required to achieve convergence. After using the standard parameters of 100 trees with a maximum depth of 20, Fig. 6(a) illustrates the convergence rate of the tracker with respect to the success rate as the average translation error increases. Here, we show that the tracker performs equally well between 11

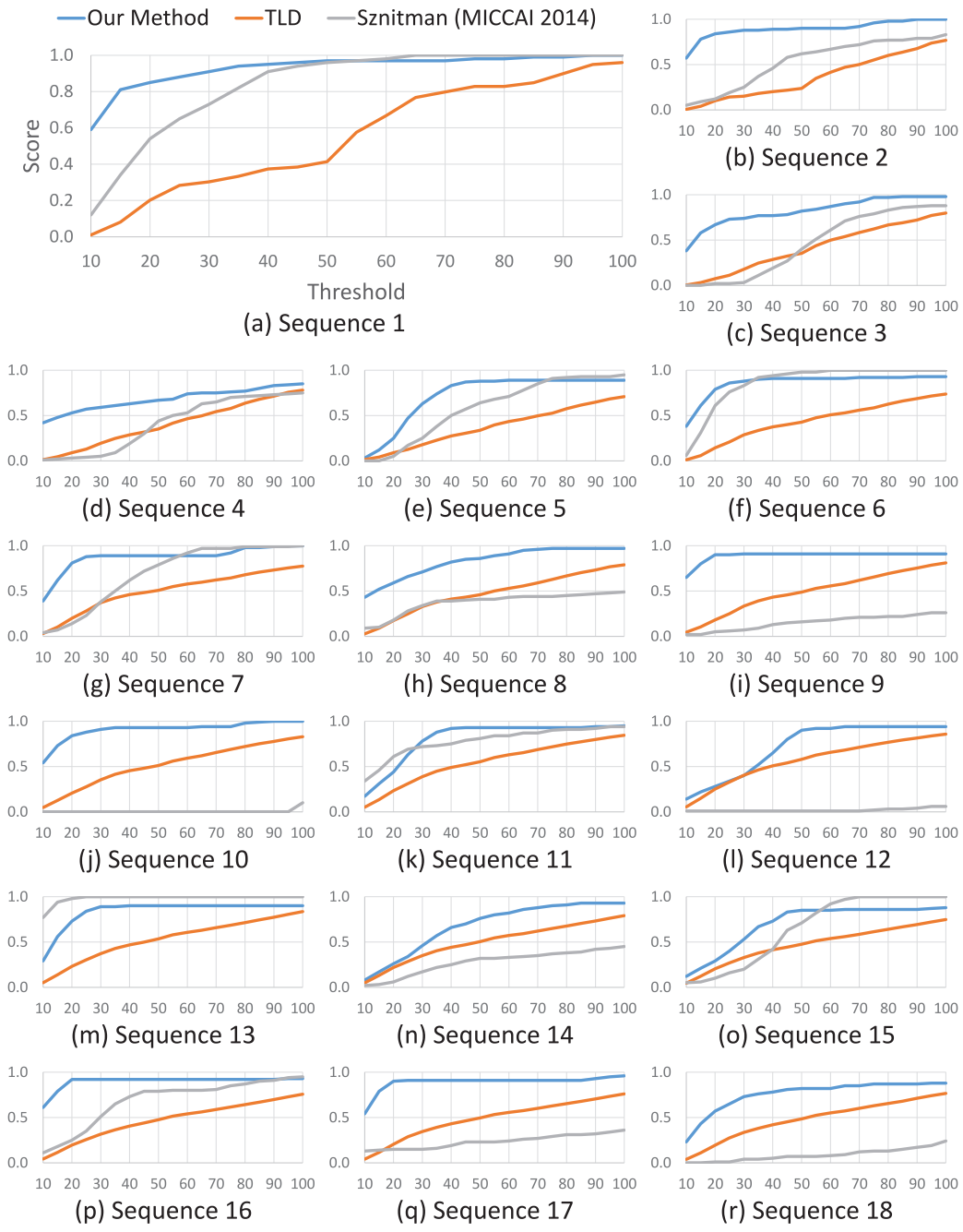


Fig. 12. Comparison sequentual evaluation of instrument dataset to TLD and Sznitman et al. (2014): The results of our method on the sequential evaluation of the instrument dataset is compared to the performance the TLD tracker from Kalal et al. (2012) and the tool tracker of Sznitman et al. (2014). The graphs show the scores for the estimation of the central joint (CF) by means of the pixel threshold metric *KT*.

and 12 iterations. With 11 iterations, the tracker runs at approximately 1.8 ms with one CPU core, see Fig. 6(b). In addition, Fig. 6(c) shows the performance of the tracker as the number of templates increases in learning the random forest. Notably, there is no significant decline in performance as the number of learned templates increases from 100 in the sequential evaluation to 400-500 in the instrument-dependent evaluation to 1800 in the generalized evaluation.

Parameters for pose estimation: We evaluate the performance of the pose estimation on one sequence of the Public Dataset and one sequence of the Instrument Dataset by varying the respective parameters. Based on the results depicted in the Fig. 7, we decided to use 15 trees with a maximum depth of 50 for the pose estimation in the following experiments. The depth of the trees are considerably high due to the high variation in terms of lighting conditions, appearance and motion of the surgical instrument. A patch size resolution of 50 pixels per dimension has proven to yield good results. For all the following experiments, a HoG features bin size of 9 is used and the resolution of the dense-window grid is set to 100×100 pixels.

5.2. Evaluations on the instrument dataset

With the introduction of this new dataset, we have the possibility of performing more detailed experiments regarding the ability of the algorithm to generalize for unseen conditions.

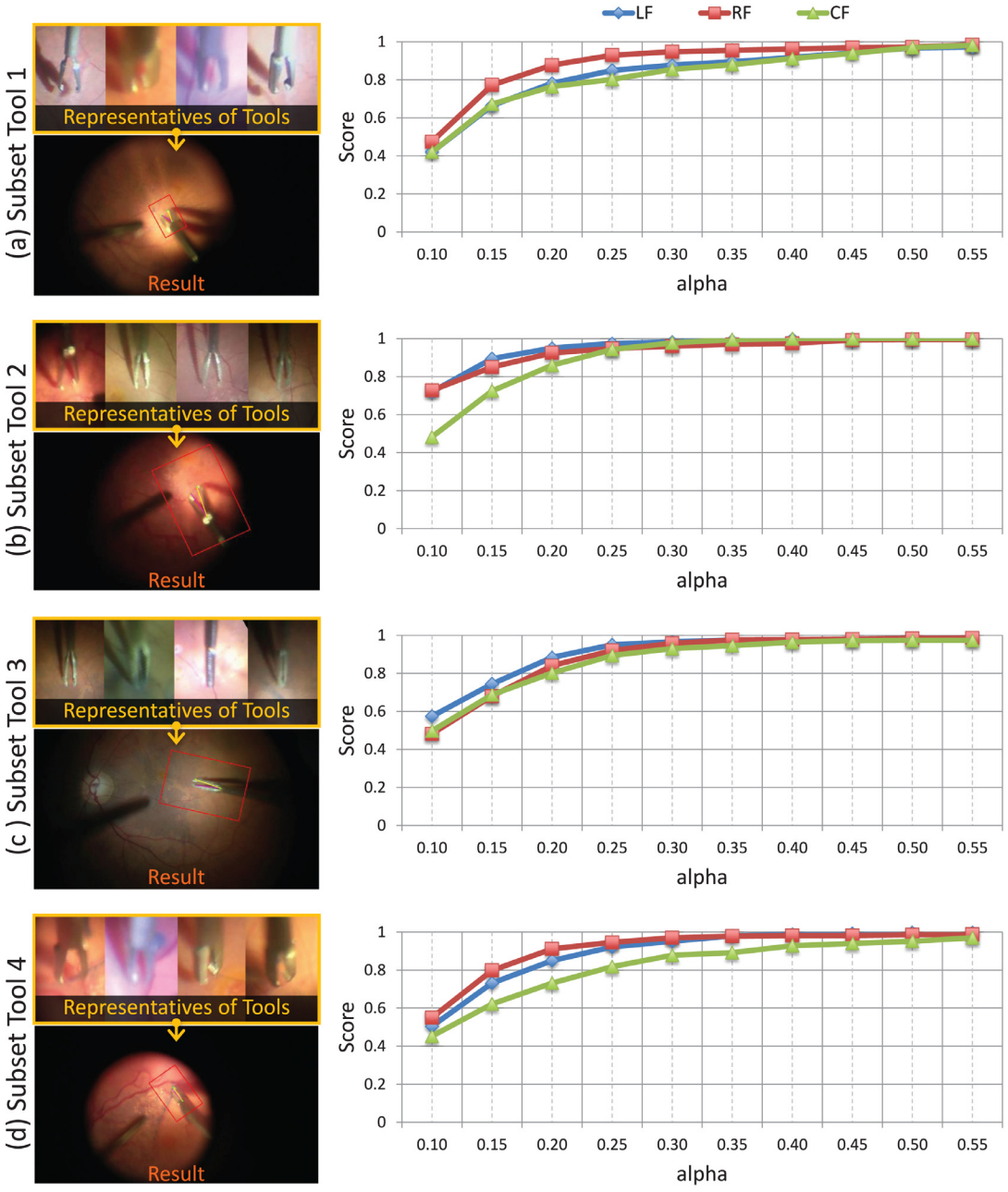


Fig. 13. Results for instrument dependent evaluation of instrument dataset: First stage of generalization. The experiment is performed separately for every subset of the Instrument Dataset by training on all the first halves of the respective sequences and evaluating on the remaining ones of the subset. The score in the right column shows the *KBB* score for all keypoints.

First, we perform a *sequence-wise* experiment on the dataset. The forests are trained on the first 100 frames of a sequence and evaluated on the remaining ones. As depicted in Figs. 8 to 11, the algorithm can reliably predict the joint positions for various blur levels and illumination changes reaching over 86% score for a strict PCP with $\alpha = 0.5$ in every sequence. The estimation of the joint positions seems to be more challenging in case of bulky instrument (compare Seq. 14, 15 and 18 – Tool 4). A reason for the comparatively worse result in Seq. 15 is also the higher amount of reflection and blur.

The performance of our method is exemplarily compared to the online learning algorithm TLD (Kalal et al., 2012) and to the offline method Fast Part-Based Classification (FPBC)[2] for RM sequences introduced by Sznitman et al. (2014) by comparing the estimation for the central joint (CF) by means of *KT*. TLD stands for tracking, learning and detection, whereas the tracker follows the object of interest in subsequent frames, the detector estimates the appearance changes and corrects the tracker and the learning step calculates the errors of the detector and updates it. The authors claim that TDL is successful for challenging videos and can handle frequent tracking failures. We initialized the bounding box around the central point using the ground truth annotation. As depicted in Fig. 12, our algorithm outperforms the baseline online tracker in every sequence.

The Fast Part-Based Classification (FPBC) algorithm represents an offline state-of-the-art tool tracking method for medical applications, which consists of the following steps: a multiclass classifier (Gradient Boosting) accelerated by an early-stopping scheme

[2] https://sites.google.com/site/sznitr/code-and-datasets

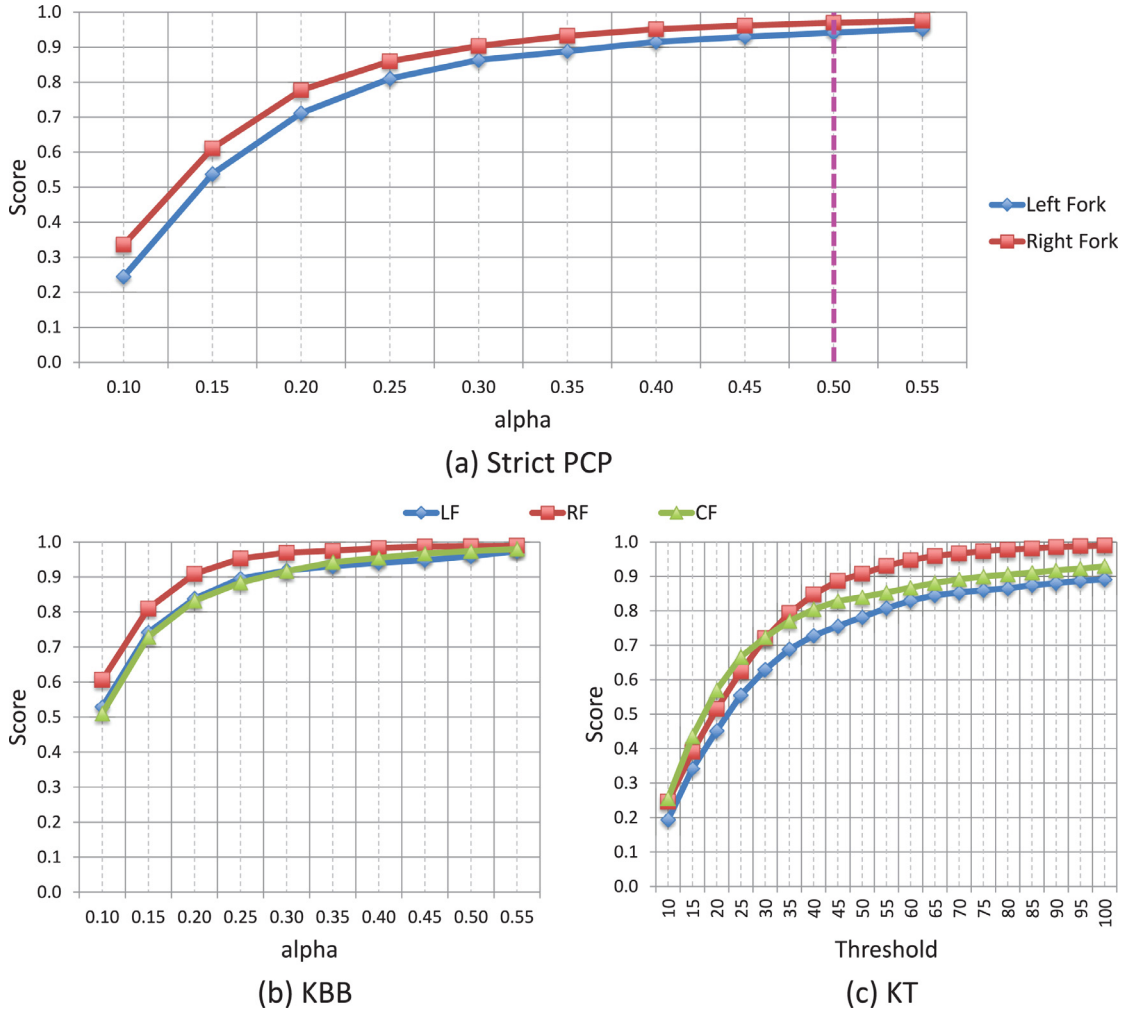


Fig. 14. Result for complete evaluation of the instrument dataset: The method was trainined on all the first 100 frames of the Instrument Dataset and evaluated on the second 100 frames. In (a), the strict PCP score for the instrument parts is visualized for different alpha values. (b) and (c) show the keypoint evaluation, whereas the former is the tool size dependent evaluation and the latter is the threshold metric.

(EDE) assigns each pixel to a class. Afterwards, a response map is generated and RANSAC is considered to obtain inliers. Finally, a weighted averaging estimates the pose of the instrument. For sake of completeness, it is important to mention that the following routine were implemented by ourselves: the patch extraction, where the original annotation were used as center of the $r \times r$ patch. For the background class, an algorithm was implemented to randomly select patches, which did not include the tool but the retina or the black background. For the Instrument Dataset, we downsampled the original images by a factor of 3 to increase evaluation speed (final image size 640 $\times$ 360) and considered four classes (background, insertion point, tool center and the tool shaft), whereas the tool center was defined as the middle point between insertion point and tool shaft. Patches were selected of size 48 $\times$ 48 pixel in order to include the instrument in all possible zoom factors of the different videos. We used the first 75 frames for training, the following 25 for the EDE early stopping criteria and finally the last 100 to test the procedure. The tree depth was set to 2, number of boosting iterations $T = 200$, RANSAC with 500 iterations, number of stopping criteria evaluation to $\delta = 10$ and the entropy threshold is set to $\gamma = 10^{-3}$. For details about the parameters, we refer the reader to the original paper by Sznitman et al. (2014). A graphical comparison can be seen in Fig. 12. It is noticeable that overall, our algorithm shows more stable performance results than FPBC. In the Seq. 10 and 12, the algorithm FPBC is confused by the presence of various vascular structures and high amount of black background, whereas the proposed method still produces reliable results.

In the next step, an *instrument dependent* experiment was performed on each subset of the Instrument Dataset. Within each subset, the shape of the instrument is similar. This allows us to investigate whether the method can generalize regarding changes in illumination and background. For this purpose, we include the first 100 frames of all the sequences of a subset in the training dataset and test on the remaining ones. Due to the differences in tool tip resolution, we now employ the newly introduced metric *KBB* for the performance evaluation. The results are summarized in Fig. 13. As already indicated by the sequential experiment, the localization is more challenging for the bulky tools (i.e., Tool 1 and Tool 4). However, the extension of the dataset to more sequences seems to increase the capture range and performance of the algorithm.

The next level is the generalization for various tool shapes. In the *complete* experiment, all the first halves of the 18 sequences are included in the training set. In this way, we can evaluate whether the algorithm can generalize not only for background, illumination changes and blurriness levels, but also for instrument shapes. Due to the more challenging scenario, we used 30 trees for the pose estimation. The results are visualized in Fig. 14. Although this experiment includes a high complexity for the learning algorithms, we reach a strict PCP score of 93.7% for the left instrument part and 95.54% for the right instrument part with respect

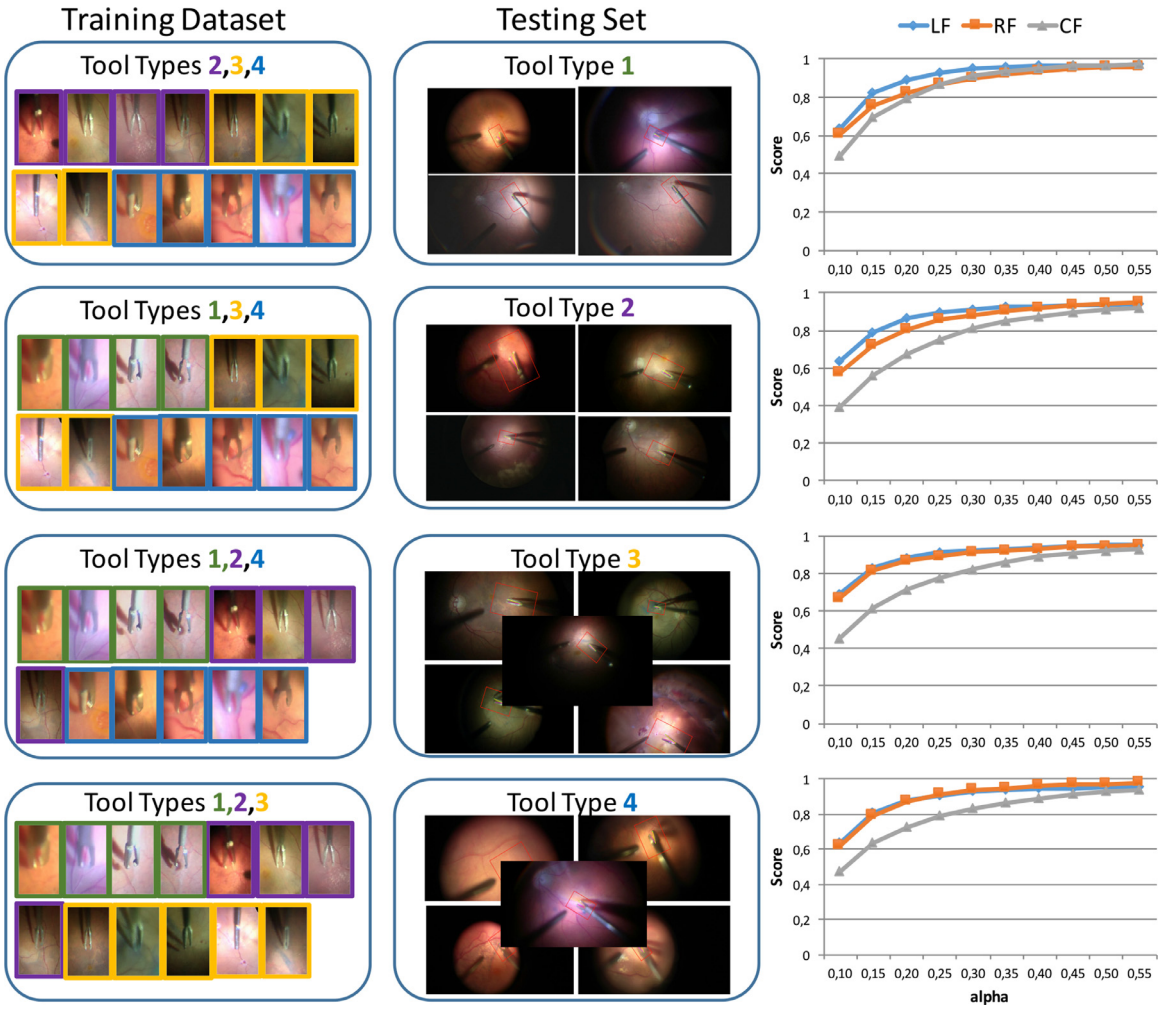


Fig. 15. Results for the cross validation of instrument dataset: The trees are trained on all sequences of three tool types and evaluated on all sequences on the remaining tool type. The score in the right column shows the *KBB* score for all keypoints.

to $\alpha = 0.5$. Regarding the keypoint scores, 83.7% for the LF, 90.4% for the RF and 80.5% for the CF of the predictions are evaluated as correct by means of the metric *KBB* with $\alpha = 0.2$. The metric *KT* indicates a worse performance because pixel thresholds are directly compared across sequences although the size of the instrument tip in pixel varies significantly.

The most challenging experiment is the *leave-one-out* validation on the subsets: the forests are trained on all sequences of three tool types and are tested on all sequences of the unseen tool type. The difficulty of this setting lies in the generalization to both an unknown geometry and unseen sequences. As depicted in Fig. 15, the proposed algorithm can build on the vast dataset and achieves at least 86.7% for the LF, 80.7% for the RF and 67.8% for the CF success rate by means of the *KBB* metric with $\alpha = 0.2$ in all four cross validations. Regarding the *PCP* score with $\alpha = 0.5$, the method predicts the instrument parts correctly in at least 88.2% for the left instrument part and 89.5% for the right instrument part.

5.3. Evaluation on the public dataset

On the Public Dataset, the performance of the proposed method is compared to state-of-the-art methods including the data-driven visual tracking (DDVT) by Sznitman et al. (2012), the visual tracking (MI) by Richa et al. (2011), a gradient-based image registration (SCV) by Pickering et al. (2009) and an online-learning approach (ITOL) by Li et al. (2014). In order to be consistent, the estimation of the position of the center joint (CF) is compared and the experiments are performed analogously to other works. The *sequential* evaluation was performed by training the random forests on the first half of a sequence and test on the remaining half (Fig. 16). In the *complete* experiment, the forest were trained on all the first halves of the three sequences and tested on the remaining halves (Fig. 17). Both experiments indicate that the proposed method outperforms the state-of-the-art methods.

5.4. Evaluation on the laparoscopy sequence

Analogously to works presented on this dataset (Sznitman et al., 2012; Li et al., 2014), we used the first 500 frames of the sequence as training dataset. For comparison, the performance was evaluated on the remaining frames by means of the pixel-wise measure *KT* for the center joint CF for thresholds between 15 and 40 pixels. Although two instruments are shown in the sequence, we perform the experiment only for Tool 1, which is more interesting for pose estimation due to grasping operations and various movements. Tool 2 remains relatively static and closed. As depicted in Fig. 18, the proposed method performs similar to the baseline algorithm DDVT (Sznitman et al., 2012) in terms of prediction of the keypoint CF. In contrast to the other methods, our algorithm did not need to be reinitialized and was able to track all three joints of the articulated instrument for the entire sequence.

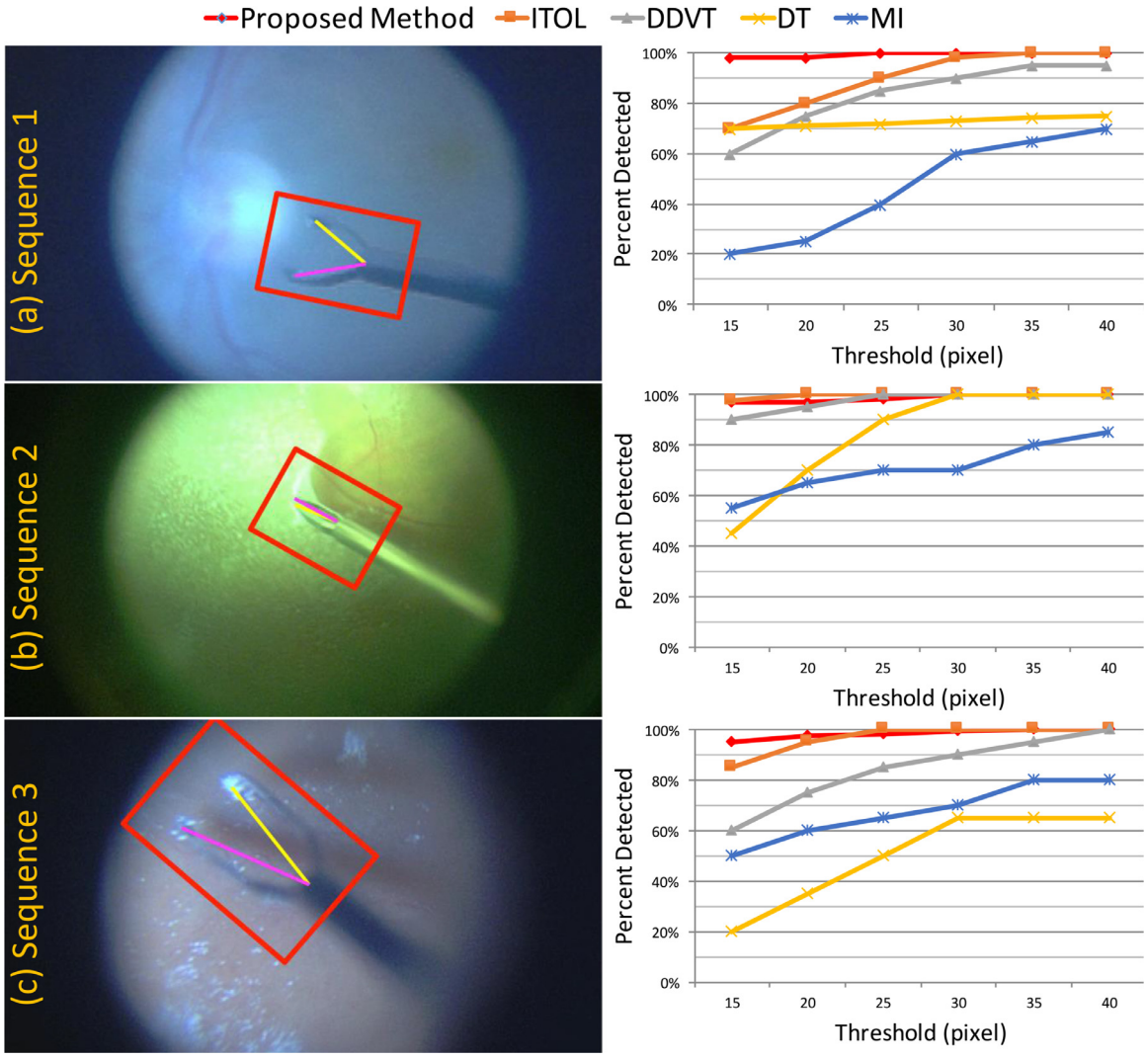
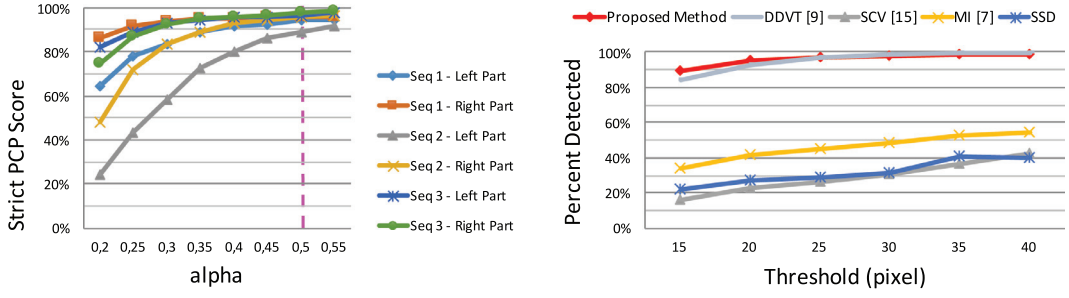


Fig. 16. Results for sequential evaluation of public dataset: For every sequence separately, the forests are trained on the first half and tested on the remaining half. The result for the central joint (CF) is compared analogously to the cited works by means of threshold distance in pixel (*KT*). The compared methods are ITOL and DT presented in the work by Li et al. (2014), DDVT by Sznitman et al. (2012) and MI by Richa et al. (2011).



(a) Public Set – Strict PCP Score

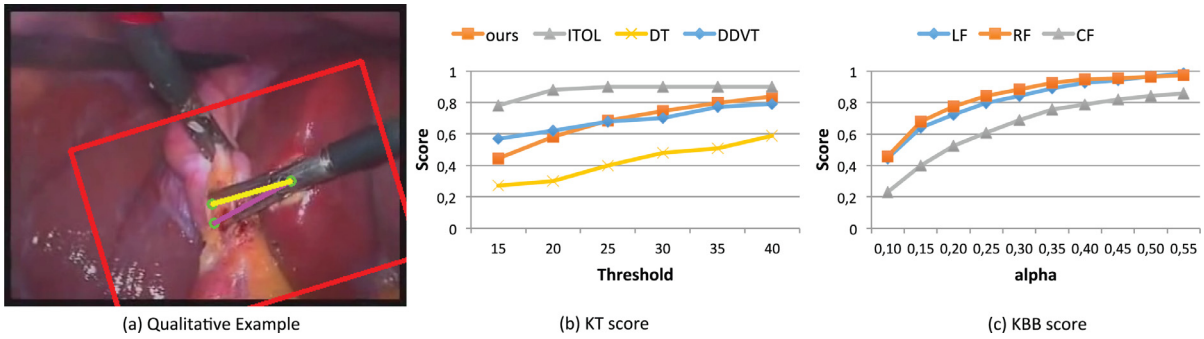
(b) Full Dataset – Threshold Score

Fig. 17. Results for public dataset: In (a), the strict PCP scores for learning and testing on the separate sequences is visualized. The vertical pink line represents the standard value for alpha in human pose estimation. (b) depicts the *KT* score for the estimation of the central joint (CF) when training the forest on all halves of the sequences and evaluating on the second halves. The compared methods are DDVT by Sznitman et al. (2012), SCV by Pickering et al. (2009), MI by Richa et al. (2011) and SSD. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

6. Discussion and concluding remarks

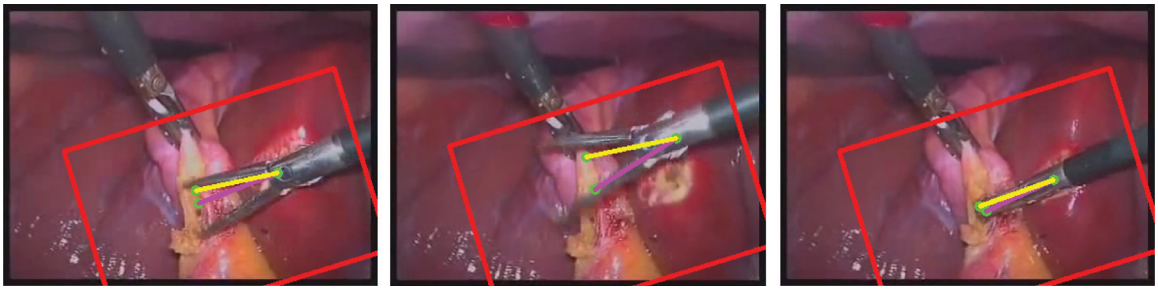
In this paper, we presented a robust framework for tracking a surgical instrument in *in-vivo* RM sequences. In contrast to other methods, which focus on estimating just the center joint of a forceps, we recover the tool's articulated pose in real-time. To withstand noisy and incomplete data, we have proposed to base both parts of the algorithm on random forest, which has shown to be a fast, flexible and robust machine learning tool for a variety of tasks.

The advantage of separating the problem into two tasks is twofold. On the one hand, the algorithm can make use of both color



(a) Qualitative Example (b) KT score (c) KBB score

Fig. 18. Results for laparoscopy sequence: In (a), a qualitative example of the estimation is visualized. Figure (b) depicts the KT metric for the central joint (CF) in comparison to the methods DDVT by Sznitman et al. (2012), DT and ITOL by Li et al. (2014). In (c), results for all joints by means of the metric *KBB* are shown.



(a) Wrong Pose (b) Scale Change (c) Shifted CF

Fig. 19. Failure cases for laparoscopy sequence: in some cases, the proposed method has problems with the localization of the keypoints. In (a) the position of the CF and RF are predicted correctly, but the left tool tip (LF) is distant to the ground truth, which results in an incorrect pose. In (b), the significant scale change of the size of the instrument leads to totally shifted localizations. In (c) the tool tips are inserted into tissue. Consequently, the geometric relation between the tool tips and the center joint is changed.

as well as gradient information. On the other hand, the computationally more expensive step of extracting gradient information is reduced to a smaller region of interest (i.e., for the sake of pose estimation only), thus making our algorithm particularly efficient. Please note that even if the prominent color of the background changes sensibly, the contrast between the metallic appearance of the instrument and the retina remains a valuable and easily accessible cue. With less than 2 ms of computation time using one CPU core, the color-based temporal tracker takes advantage of the contrast to efficiently localize the position of the instrument tip. However, color information tends to be less reliable for precise estimation in RM sequences, due to typically strong illumination and appearance changes. For this reason, we employed gradient information for pose estimation in the second step. Differently from the approach by Sznitman et al. (2014), we do not use gradient information from patches within the entire frame, but limit this computationally expensive step to the region of interest provided by the tracker. Another important difference is that the tracker relies on temporal information available from previous frames of the sequence, while the pose estimation stage only exploits the information available from the current frame.

The performance of the proposed methods was evaluated on three different datasets by means of four different metrics. The results show that our method can not only handle unseen changes within a sequence, but also generalize for various illumination and instrument appearance changes. In particular, the complete evaluation of the Instrument Dataset was one of the most challenging experiment including 18 sequences of four different instrument shapes. Our algorithm yielded a strict PCP score ($\alpha = 0.5$) of more than 95% for both the left and right parts of the forceps, and 84.1% for the *LF*, 90.8% for the *RF* and 83.2% for the *CF* in terms of *KBB* with $\alpha = 0.2$. In contrast to the *KT* metric, the newly introduced *KBB* metric takes into account the variations in instrument appearance size and image pixel resolution and thereby allows the performance evaluation across sequences. In the experiments on the laparoscopic instrument sequence, the pose estimation revealed difficulties regarding large scale changes of the instrument size (Fig. 19). This could be caused by the fact that the HoG features are extracted with a fixed patch size and can be tackled by extending the tracker so that it estimates the full rigid transformation update for the template. However, looking at Fig. 18, the performance of the proposed method is still comparable to state-of-the-art methods.

The proposed method is designed for one single instrument. However, the simultaneous tracking of several tools can easily be realized by initializing a separate thread of the algorithm for every instrument. An important observation is that our method is not confused by the presence of another tool in the image frame. In RM sequences, usually only one forceps is utilized. In contrast to the *Public* dataset, most parts of the sequences within the novel *Instrument* dataset include the intra-ocular light in the focused area. Being a metallic and rigid device, it is similar in appearance to the tracked forceps, and as such could be considered as a second tool. In some sequences (e.g. Seq. 5 and Seq. 15, see Figs. 9 and 10), the light source is very close to the tool tip. Even in this challenging situation, the proposed algorithm is not misled by this additional nuisance. Also in the laparoscopy dataset, the presence of a second forceps does not deteriorate the performance.

One limitation of our method is the lack of a recovery procedure in case of tracking failures. The pose estimation relies on the output of the tracker, which is the bounding box containing the tool. This region of interest does not necessarily have to be precise because the final prediction is produced by the pose estimation. However, if the instrument is not captured by the bounding box the pose estimation would also fail at inferring the instrument joints. Nevertheless, in all our experiments, this case did not occur

and therefore the tracker did not have to be re-initialized with the ground truth, if the tool was present in the frame and showed a continuous movement. A detector would further increase the robustness of our method and make it more suitable for the clinical practice.

An interesting future direction is represented by the use of the inferred pose as an additional input for the tracker, within a closed-loop framework where the pose estimation stage also provides feedback for the tracker. This brings the challenge of synergically combining the predictions from two forests to achieve a better performance.

Acknowledgments

This research is partially supported by the CARC grant (identification number IUK456/002) and Carl Zeiss Meditec AG, Munich.

References

Allan, M., Chang, P.-L., Ourselin, S., Hawkes, D.J., Sridhar, A., Kelly, J., Stoyanov, D., 2015. Image based surgical instrument pose estimation with multi-class labelling and optical flow. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015. Springer, pp. 331–338.

Allan, M., Ourselin, S., Thompson, S., Hawkes, D.J., Kelly, J., Stoyanov, D., 2013. Toward detection and localization of instruments in minimally invasive surgery. In: IEEE Transactions on Biomedical Engineering, 60, pp. 1050–1058.

Baek, Y.M., Tanaka, S., Kanako, H., Sugita, N., Morita, A., Sora, S., Mochizuki, R., Mitsuishi, M., 2012. Full state visual forceps tracking under a microscope using projective contour models. In: Proceedings of IEEE ICRA, pp. 2919–2925.

Baker, S., Matthews, I., 2004. Lucas-kanade 20 years on: a unifying framework. IJCV 56 (3), 221–255.

Belagiannis, V., Amann, C., Navab, N., Ilic, S., 2014. Holistic human pose estimation with regression forests. In: Perales, F.J., Santos-Victor, J. (Eds.), AMDO 2014. LNCS, 8563. Springer, Heidelberg, pp. 20–30.

Blum, T., Sielhorst, T., Navab, N., 2007. Advanced augmented reality feedback for teaching 3d tool manipulation.

Breiman, L., 2001. Random forests. Mach. Learn. 45 (1), 5–32. doi:10.1023/A:1010933404324.

Cattin, P.C., Bay, H., Van Gool, L., Székely, G., 2006. Retina mosaicing using local features. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2006. Springer, pp. 185–192.

Chen, C.-J., Huang, W.-W., Song, K.-T., 2013. Image tracking of laparoscopic instrument using spiking neural networks. In: ICCAS 2013, pp. 951–955.

Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, 1. IEEE, pp. 886–893.

Ehlers, J., Kaiser, P.K., Srivastava, S.K., 2014. Intraoperative optical coherence tomography using the rescan 700: preliminary results from the discover study. Br. J Ophthalmol. 1329–1332.

Eitz, M., Hildebrand, K., Boubekeur, T., Alexa, M., 2011. Sketch-based image retrieval: Benchmark and bag-of-features descriptors. Vis. Comput. Graph., IEEE Trans. 17 (11), 1624–1636.

Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D., 2010. Object detection with discriminatively trained part-based models. Pattern Anal. Mach. Intell., IEEE Trans. 32 (9), 1627–1645.

Gabriele, M.L., Wollstein, G., Ishikawa, H., Kagemann, L., Xu, J., Folio, L.S., Schuman, J.S., 2011. Optical coherence tomography: History, current status, and laboratory work. Invest. Ophthalmol. & Vis. Sci. 2425–2436.

Holzer, S., Pollefeys, M., Ilic, S., Tan, D.J., Navab, N., 2012. Online learning of linear predictors for real-time tracking. In: 12th European Conference on Computer Vision (ECCV).

Jurie, F., Dhome, M., 2002. Hyperplane approximation for template matching. PAMI 24 (7), 996–1000.

Kalal, Z., Mikolajczyk, K., Matas, J., 2012. Tracking-learning-detection. Pattern Anal. Mach. Intell., IEEE Trans. 34 (7), 1409–1422.

Li, Y., Chen, C., Huang, X., Huang, J., 2014. Instrument tracking via online learning in retinal microsurgery. In: Golland, P., et al. (Eds.), MICCAI 2014. LNCS, 8673. Springer, Heidelberg, pp. 464–471.

Nilsback, M.-E., Zisserman, A., 2008. Automated flower classification over a large number of classes. In: Computer Vision, Graphics & Image Processing, 2008. ICVGIP'08. Sixth Indian Conference on. IEEE, pp. 722–729.

Pavlidis, M., Georgalas, I., K÷rber, N., 2015. Determination of a new parameter, elevated epiretinal membrane, by en face oct as a prognostic factor for pars plana vitrectomy and safer epiretinal membrane peeling. J. Ophthalmol.

Pezzementi, Z., Voros, S., Hager, G.D., 2009. Articulated object tracking by rendering consistent appearance parts. ICRA 2009, pp. 3940–3947 (2009).

Pickering, M.R., Muhit, A.A., Scarvell, J.M., Smith, P.N., 2009. A new multi-modal similarity measure for fast gradient-based 2d-3d image registration. In: EMBC 2009, pp. 5821–5824.

Reiter, A., Allen, P.K., Zhao, T., 2012. Feature classification for tracking articulated surgical tools. In: Ayache, N., Delingette, H., Golland, P., Mori, K. (Eds.), MICCAI 2012, Part II. LNCS, 7511. Springer, Heidelberg, pp. 592–600.

Richa, R., Balicki, M., Meisner, E., Sznitman, R., Taylor, R., Hager, G., 2011. Visual tracking of surgical tools for proximity detection in retinal surgery. In: IPCAI, pp. 55–66.

Rieke, N., Duca, S., Navab, N., Eslami, A., 2016. Automatic ioct positioning during membrane peeling via real-time high resolution surgical forceps tracking. In: International Society of Imaging in the Eye Conference, Association for Research in Vision and Ophthalmology (ARVO 2016), To be published. Seattle, USA

Rieke, N., Tan, D.J., Alsheakhali, M., Tombari, F., Amat di San Filippo, C., Belagiannis, V., Eslami, A., Navab, N., 2015. Surgical tool tracking and pose estimation in retinal microsurgery. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015. Springer, pp. 266–273.

Roodaki, H., Filippatos, K., Eslami, A., Navab, N., 2015. Introducing augmented reality to optical coherence tomography in ophthalmic microsurgery. In: 2015 IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2015, Fukuoka, Japan, September 29 – Oct. 3, 2015, pp. 1–6.

Sznitman, R., Ali, K., Richa, R., Taylor, R.H., Hager, G.D., Fua, P., 2012. Data-driven visual tracking in retinal microsurgery. In: Ayache, N., Delingette, H., Golland, P., Mori, K. (Eds.), MICCAI 2012, Part II. LNCS, 7511. Springer, Heidelberg, pp. 568–575.

Sznitman, R., Basu, A., Richa, R., Handa, J., Gehlbach, P., Taylor, R.H., Jedynak, B., Hager, G.D., 2011. Unified detection and tracking in retinal microsurgery. In: Fichtinger, G., Martel, A., Peters, T. (Eds.), MICCAI 2011, Part I. LNCS, 6891. Springer, Heidelberg, pp. 1–8.

Sznitman, R., Becker, C., Fua, P., 2014. Fast part-based classification for instrument detection in minimally invasive surgery. In: Golland, P., et al. (Eds.), MICCAI 2014. LNCS, 8673. Springer, Heidelberg, pp. 692–699.

Tan, D.J., Ilic, S., 2014. Multi-forest tracker: A chameleon in tracking. In: CVPR 2014, pp. 1202–1209.

Yang, Y., Ramanan, D., 2013. Articulated human detection with flexible mixtures of parts. Pattern Anal. Mach. Intell., IEEE Trans. 35 (12), 2878–2890.

Yigitsoy, M., Belagiannis, V., Djurka, A., Katouzian, A., Ilic, S., Pernus, E., Eslami, A., Navab, N., 2015. Random ferns for multiple target tracking in microscopic retina image sequences. In: Biomedical Imaging (ISBI), 2015 IEEE 12th International Symposium on. IEEE, pp. 209–212.