Unified detection and tracking of instruments during retinal microsurgery

Raphael Sznitman, Rogerio Richa, Russell H. Taylor *Fellow, IEEE*, Bruno Jedynak and Gregory D. Hager, *Fellow, IEEE*

Abstract—Methods for tracking an object have generally fallen into two groups: tracking by detection, and tracking through local optimization. The advantage of detection-based tracking is its ability to deal with target appearance and disappearance, but does not naturally take advantage of target motion continuity during detection. The advantage of local optimization is efficiency and accuracy, but requires additional algorithms to initialize tracking when the target is lost.

To bridge these two approaches, we propose a framework for unified detection and tracking as a time-series Bayesian estimation problem. The basis of our approach is to treat both detection and tracking as a sequential entropy minimization problem, where the goal is to determine the parameters describing a target in each frame. To do this we integrate the Active Testing paradigm with Bayesian filtering, and this results in a framework capable of both detecting and tracking robustly in situations where the target object enters and leaves the field of view regularly. We demonstrate our approach on a retinal tool tracking problem and show through extensive experiments that our method provides an efficient and robust tracking solution.

Index Terms—Unified object detection and tracking, Active Testing, Instrument tracking, Adaptive Sensing, Retinal microsurgery.

1 Introduction

Visual tracking has been intensely studied in computer vision over the past two decades [1]. Informally, the objective of visual tracking is to provide an accurate estimate of the configuration of a target across time, where the term "configuration" denotes parameters describing the position, pose, or shape of an object. A general solution involves solving two subtasks: (i) detecting the target in the initial image in which it appears, and (ii) predicting and refining (*i.e.*, tracking) the configuration of the detected target in subsequent images [1], [2]. While extensive research in this area has produced excellent tracking systems, combining these two subtasks remains difficult when the target appearance is complex or when the target enters and leaves the field of view frequently.

Indeed, the initial detection of the target is often the most difficult aspect of a tracking system. This is particularly the case when object appearance is complex and many configuration parameters are involved [3]–[5]. Even more so, performing detection with accuracy and at frame rate for objects that have many pose parameters is often infeasible due to the enormous size of the search space *i.e.*, easily over a billion hypotheses. In addition, while object detection and localization algorithms, such as classifier cascades [6], [7] or branch-and-bound algorithms with SVM based cost functions [8] are expected to determine parameters that define an object (*i.e.*, a bounding box [6], or object segmentation [9]), incorporating prior object knowledge into these frameworks to improve detection in subsequent images is usually an ad-hoc adaptation of common filtering paradigms [10], [11].

Conversely, some tracking approaches have tried to place detection and tracking under the same umbrella. Approaches have included strategies for removing faulty detections by model validation and temporal non-maximal suppression [12]–[14]. However, in these cases only restricted regions of the image are considered when searching for the target, which often leads to tracking failures when motion models are violated. Other approaches have performed tracking by using cascades of detection processes that refine the possible object location [15], [16]. While these have generally been shown to be robust, the construction of these systems has been hand-crafted for each problem setting with no underlying principle. This in turn makes them challenging to implement in real-applications.

1.1 An Active Testing Approach

In this work, we propose an algorithmic solution for the task of detection and tracking. Our approach embodies and extends the *Active Testing* (AT) paradigm [3], [17] and allows both detection and tracking to be considered within the same framework. In particular, in both tasks, estimating the object parameters is achieved by using the same sequential entropy minimization procedure, and hence removes the need for two separate algorithms and the protocols necessary to join them. By using the AT framework and Bayesian filtering strategies, the entire search space of the object is always considered when searching for the object pose, and informative priors

R. Sznitman, R. Richa, R. H. Taylor and G. D. Hager are with the Dept. of Computer Science, B. Jedynak is with the Dept. of Applied Mathematics and Statistics Johns Hopkins University, Baltimore, MD, 21218, USA.
e-mail:{sznitman,richa,rht,bruno.jedynak,hager}@jhu.edu

Digital Object Indentifier 10.1109/TPAMI.2012.209

can effectively be used to weigh likely pose candidates in subsequent images. In addition, within the AT optimization, the parameters of the object are searched sequentially, requiring far fewer observation models when compared to [17]. Consequently, the learning stage of the framework is significantly simplified. Finally, central to the tracking problem, we detail how to incorporate traditional gradient-based tracking methods for this task [1], [18], [19] within our framework.

In summary, the unique aspects of our framework are: (i) an information-based heuristic is used to guide the search process during detection and tracking, allowing both to be solved using the same optimization strategy while considering the entire search space at all times, (ii) traditional local optimization tracking is incorporated into a larger class of image functions used to gather information regarding the location of the target and (iii) the process of learning observation models from training data is greatly simplified by introducing a new parametrization of these models.

1.2 Instrument Tracking in Surgery

We demonstrate our approach on the task of detecting and tracking a surgical instrument during retinal microsurgery. From a computer vision point of view, visual tracking of instruments is a challenging problem within a controlled environment. The instruments used during surgery are known *a priori*, making it possible to learn their appearance and geometry beforehand. However, the instruments are subject to a large variety of appearance changes during procedures, making tracking difficult. For example, an instrument may be partially blurred, the shadow of the instrument may be similar in appearance to itself, and local and global illumination conditions change with time. While one possibility would be to model the background and detect outliers to estimate the instrument pose as in [20], modeling the background in *in-vivo* settings remains challenging, particularly when the eye moves during the procedure.

In the context of surgical applications and with the goal of providing semi-automated assistance for clinicians during procedures, tool detection has received increased attention in recent years. Towards this end, a number of techniques have been proposed such as in [20], [21] where kinematic information and instrument templates were used to detect and track tools in image sequences. In [22]–[25], instrument models based on pixel, or local color are learned and used for detection and segmentation. Other methods, as in [26], [27] extracted edges and lines to ultimately infer tool tip position. Yet, to date a majority of methods have relied on the ability to alter the tool appearance directly by adding visible markers to facilitate tool detection [28]–[30].

Unlike other approaches for this task that demand prohibitively high computational costs [31] or extremely accurate initialization methods [19], [21], [32], our solution provides a feasible and automatic solution to tracking retinal instruments without the need for accurate instrument motion models. More importantly, this remains the case when the instrument enters and leaves the field of view often. To demonstrate this, our approach is validated on a microscope platform that uses a phantom eye and also on images from human retinal microsurgery. While a preliminary version of our method was presented in [33], this paper provides a full description of our approach, integrates gradient-based tracking methods in our framework, and presents extensive additional empirical results on both phantom eye data and human *in-vivo* data. While this work focuses on this particular application, the approach is relevant for a number of other applications as well.

The remainder of the paper is organized as follows: in Sec. 2 we first introduce some notation and describe the problem formulation. We then introduce tracking as a Bayesian sequential estimation problem and describe how our approach embodies this structure in Sec. 3. In Sec. 4 we describe the AT model for detecting retinal instruments. In Sec. 5, we perform extensive experimentation to validate our approach on both phantom and *in-vivo* data. Finally, we conclude with some closing remarks in Sec. 6.

2 Problem Formulation

The aim of this work is to locate a surgical instrument in a sequence of monocular images, gathered from the operating microscope. Similar to most detection and tracking approaches, we assume that the instrument's position and orientation[1], or *pose*, can be described by a relatively small number of parameters. As depicted in Fig. 1(*left*), we let the parameters representing the instrument be defined as $Y = (Y_1, Y_2, Y_3)$, where Y_1 corresponds to the instruments point of entry in the image (*i.e.*, a pixel location on the image boundary), Y_2 describes the angle the instrument makes with Y_1 and Y_3 is the instruments length measured in pixels. This particular parametrization is chosen as it is simple and intuitive to the retinal microsurgery application.

Ultimately, we are interested in determining the values of $Y^t = (Y_1^t, Y_2^t, Y_3^t)$, for all images in a sequence, $\mathcal{I}^T = (I^1, \ldots, I^T)$. For this reason, we treat Y^t as a random variable that must be inferred and where we want to compute $P(Y^t|\mathcal{I}^t), t = 1, \ldots, T$.

To do this, we first describe the pose space of the instrument. This is achieved in two steps. First, let the instrument's pose space when in the field of view, $\mathcal{S}_1$, be

$$\mathcal{S}_1 = \{[0, L] \times [-\pi/2, \pi/2] \times [\delta, D]\}$$

where L is the perimeter length of the image, δ and D are the minimum and maximum instrument lengths measured in pixels. In practice, δ is 10% of the width of the image.

1. The scale or size parameter is assumed to be known given that the tool must be in focus and microscopes used during procedures have very large focal lengths. Hence, we assume the tool scale is known.

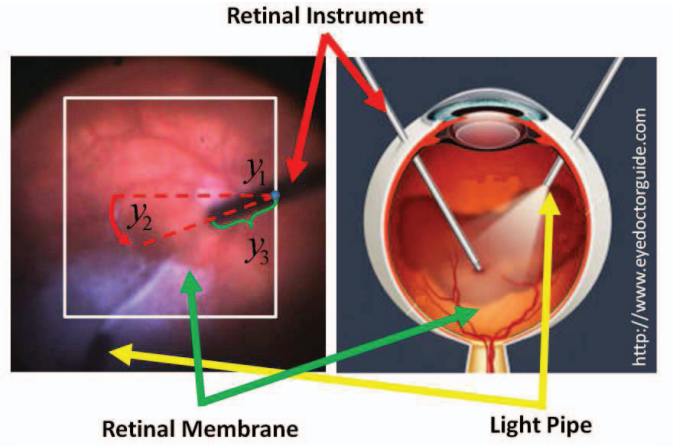

Fig. 1. (left) Tool parametrization: the retinal instrument has three parameters consisting of the point of entry of the instrument in the image, Y_1, the angle the instrument makes with the image boundary at the point of entry, Y_2 and the instruments length, Y_3. (right) Diagram of surgical environment, displaying the positioning of the light pipe and instrument during surgery.

Second, since the instrument may not be visible in the field of view of the camera, the separate space $\mathcal{S}_0 = \{\square\}$, is defined for this event (*i.e.*, the $\square$ is a token representing this case).

Finally, the complete pose space of the instrument is defined as $Y \in \mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_0$.

3 Active Testing for Tool Tracking

To detect and track an object, we cast the tracking problem in a Bayesian sequential estimation fashion [10], [11], [34]. That is, at time t, we must infer the random variable, Y^t, given the image sequence observed up to that time instant, $\mathcal{I}^t$. This can be expressed by,

$$P(Y^t|\mathcal{I}^t) = \int P(Y^t|Y^{t-1},\mathcal{I}^t)P(Y^{t-1}|\mathcal{I}^t)dY^{t-1} \quad (1)$$

$$\propto P(I^t|Y^t)\int P(Y^t|Y^{t-1})P(Y^{t-1}|\mathcal{I}^{t-1})dY^{t-1} \quad (2)$$

$$\propto P(I^t|Y^t)P(\hat{Y}^t) \quad (3)$$

where the conditional distribution given the observations can be rewritten as (1) by including the marginalization of Y^{t-1} and an application of Bayes theorem. (2) follows from (1) by another application of Bayes theorem, the assumption of Markov dynamics of the object, and the assumption that $P(Y^t)$ is a sufficient statistic for $\mathcal{I}^t$. In (3) we have defined $P(\hat{Y}^t) = \int P(Y^t|Y^{t-1})P(Y^{t-1}|\mathcal{I}^{t-1})dY^{t-1}$.

In most cases, various elements of the observation model, the dynamics and the distribution on Y have been approximated in order to allow both fast and feasible computation. For example, in methods based on Kalman filtering [10], the dynamics are assumed to be linear or are linearized, and both the distribution of Y and the observation model are Gaussian. In sampling-based methods [11], non-Gaussian distributions of Y are maintained by using particles. In our approach, we rely on a partitioning of $\mathcal{S}_1$ (using a conditional binary tree) to allow exact computation of posterior distributions and will be achieved by using a slightly adapted histogram filter [35], [36].

In the context of tracking, image observations are typically evaluated at a single location or a set of locations predicted by the distribution of the target (*i.e.*, at particle locations). But in the case of detection, following such a strategy is computationally hopeless as the number of hypotheses to evaluate is enormous. For example, when using particle filters, this would imply maintaining order the size of the pose space number of particles. For this reason, we need a mechanism to efficiently select which observations to make and the AT optimization scheme serves this purpose.

3.1 The Active Testing Model

The Active Testing (AT) [3], [17] can be viewed as an iterative stochastic optimization scheme aimed at reducing the uncertainty of a discrete random variable by sequentially asking "questions" or "queries" in an efficient fashion. In particular, this optimization scheme provides an approximation to the maximum likelihood estimate of the random variable when all possible questions are answered.

In general, the optimization process is as follows: one begins with a prior on the random variable to infer, p_0, and selects a subset of the pose space to query using a question regarding this subset. The question typically consists of computing a simple measurement on a region of the image, such as the proportion of pixels belonging to the object in some region of the image. Hence, a question is a coupling between a computation type *and* a region of the search space. The answer to the question is then used in a Bayesian way to recompute a new probability distribution or *posterior* distribution (*i.e.*, $p_1, p_2, p_3, \ldots$).

Selecting a new question for the following iteration, which we will denote as $\hat{X}$, is then achieved by choosing the question that reduces the expected entropy of the object as much as possible. This is equivalent to selecting the question that has the highest information gain,

$$\hat{X} = \arg\max_{X \in \mathcal{X}} MI(Y;X) \quad (4)$$

where MI is the mutual information [37] and $\mathcal{X}$ is the set of possible questions that are available. This procedure repeats until the entropy of the random variable drops below a pre-determined threshold, or a set number of questions have been asked.

Hence, in order to make use of the AT framework three pieces must be specified: (i) a prior distribution on the parameter, $P(\hat{Y}^t)$, (ii) a representation for the distribution of Y and (iii) a set of "questions" (and their

associated noise models), $\mathcal{X}$, pertaining to the parameter Y. We will specify these in Sec. 4.

3.2 Active Testing Filtering

At this point, we describe the general form of an Active Testing Filtering (ATF) algorithm (see Alg. 1). Here, the user initially provides an instrument dynamics model, $P(Y^t|Y^{t-1})$ and a prior on $P(Y^0)$. Then, for each image in the sequence, we first compute $P(\hat{Y}^t)$ (line 3) by using the provided dynamics model and the previous density. Depending on the model used, this can be computed in a number ways. We can then treat $P(\hat{Y}^t)$ as an initial prior for the AT optimization (line 4). That is, instead of using uninformative priors on the pose of the instrument for each image, we begin the AT optimization with $P(\hat{Y}^t)$, which carries information about where the instrument was previously located and how it may have moved.

Algorithm 1 Active Testing Filtering ($\mathcal{I}$)

1: Initialize: $P(Y^0), P(Y^t|Y^{t-1})$
2: **for all** I^t **do**
3: $\quad P(\hat{Y}^t) = \int P(Y^t|Y^{t-1})P(Y^{t-1})dY^{t-1}$
4: $\quad P(Y^t|\mathcal{I}^t) = ActiveTesting(I^t, P(\hat{Y}^t))$
5: **end for**

In this work, we select a simple linear instrument dynamics model of the form,

$$Y^{t+1} = AY^t + \mathcal{N}(0, \alpha) \tag{5}$$

where A is the dynamics transition matrix. In the experiments that follow, we use two different models: (i) A is the identity matrix which corresponds to assuming the tool has not moved from one frame to another. (ii) A is augmented to allow velocity estimates to be compounded in the new prior. Given that we know that the tool will enter and leave the field of view often, we expect both dynamic models to be consistently violated. While this may induce inappropriate priors $P(\hat{Y}^t)$, the active testing framework will still recover the pose of the instrument.

4 Active Testing Implementation

In this section, we describe the aspects of the AT optimization that must be specified: the representation of the probability distribution of Y and what "questions" will be available to localize an instrument. In particular, we will provide a set of questions, which can be viewed as a set of features that can be evaluated, combined and integrated by the framework and which are informative with respect to different coordinates of Y.

4.1 Probability Density Representation

To represent p_0 and the sequence of posterior distributions that will be computed, we make use of an abstract decomposition of the space $\mathcal{S}_1$. Let S denote a binary decomposition of the space $\mathcal{S}_1$. That is, S is a tree of subsets

$$S = \{S_{i,j}, i = 0, \ldots, H, j = 0, \ldots, 2^i - 1\} \tag{6}$$

The root of tree is $S_{0,0} = \mathcal{S}_1$ and $S_{i,j}$ is a subset of $S_{0,0}$. The decomposition of the tree is performed by splitting one coordinate of Y at a time, until a desired resolution is reached, at which point we repeat the procedure for another coordinate (*e.g.*, split Y_1, then Y_2 and so on). Simply put, the tree consists of a series of binary trees one after the other.

Fig. 2 depicts this decomposition visually. Here, we show a tree structure specified by (6), where blue nodes show where only Y_1 is being decomposed, red nodes show where Y_1 has been fully decomposed and Y_2 is in the process of being decomposed, and green nodes show those with both Y_1 and Y_2 fully decomposed and where only Y_3 is being refined.

Using this decomposition, describing the probability distribution of Y as a function of S is straightforward. If we denote the probability $P(Y \in S_{i,j})$ as $u_{i,j}$. Then since the node s$S_{i,j}$ are disjoint subsets, $u_{i,j} = u_{i+1,2j} + u_{i+1,2j+1}$ for every non-terminal (or leaf) node in S. Hence, observing the probability at a single level of S provides a piecewise constant representation of the distribution of Y.

Naturally, the space required to store this tree may be overwhelmingly large. For this reason, the tree will be generated in a lazy fashion and will allow us to represent the distribution of Y in a compact fashion. That is, the AT optimization will only begin with the root, $S_{0,0}$ and the tree will grow as questions are asked. This is closely related to Evolving Trees [38] which allow efficient organization of large amounts of data. Indeed, a key aspect of this method compared to classification trees, is that the construction of the tree is done online, dictated by the data at test time.

4.2 Set of Questions

To determine the pose of the instrument, the AT framework relies on the ability to ask "questions" about the content in the image. For a particular node $S_{i,j} \in S$, a "question" is a deterministic function of the image, $X_{i,j} : I_{S_{i,j}} \mapsto \mathbb{R}$, which computes a specific quantity from the image region specified by the pose subset $S_{i,j}$. The answers to the question $X_{i,j}$, denoted $Z_{i,j}$, is considered to be random and is interpreted in a probabilistic manner. In particular, when asking a question $X_{i,j}$ the answer, $Z_{i,j} = z$ is assumed to follow,

$$P(Z_{i,j} = z|Y) = \begin{cases} f_o(z; i, j) & \text{if } Y \in S_{i,j} \\ f_b(z; i, j) & \text{if } Y \notin S_{i,j} \end{cases} \tag{7}$$

where f_o and f_b are two distributions of responses, corresponding to the case where the instrument pose is in the space queried, and when it is not. These distributions are learned from representative labelled training

data and since the response, $Z \in \mathbb{R}$, both f_o and f_b are modelled as Gaussian (this will be detailed in the following subsection).

As in [17], we use two categories of questions in this framework: (i) noisy questions (Sec. 4.2.1) and (ii) noiseless or *oracle* questions (Sec. 4.2.2). These two categories of questions are motivated by the fact that oracle questions correspond to evaluating excellent, if not state-of-the-art, methods for detecting the target when looking at extremely small regions of the search space, while noisy questions help reduce the search space in order to ultimately use an oracle question. As shown in [17] this has the benefit of being computationally efficient when locating a target and allows for a simple mechanism to reject regions of the search space that have been observed fully. We now specify what questions are available in this application.

4.2.1 Noisy Questions

To detect our instrument we use five noisy questions such that each node of S only evaluates a single type of question. In particular, for $S_{i,j} \in S$ we denote the intervals for each coordinate of Y, as $Y_1 \in [a, b]$, $Y_2 \in [c, d]$ and $Y_3 \in [e, f]$. Also, we recall that δ is the minimum length the instrument must be protruding from the image boundary to be considered in the image, and let W be the width of the instrument. Fig. 2 visually depicts examples of where each question type (A through E) is evaluated in our decomposition and what computation is performed:

(A): In this question, $X_{i,j}$ computes the proportion of tool-like pixels in the region defined by $[a, b]$. This consists of a rectangular image patch along the boundary of the image, where the width of the patch is δ and is of length $[a, b]$.

Evaluating if a pixel belongs to the tool is achieved by evaluating if the RGB color of the pixel is likely to have come from a 3 dimensional Gaussian representing the instrument color. The parameters of the Gaussian are estimated using labeled training data, and pixels are classified as tool-like if their RGB color is within a fixed Mahalanobis distance to the Gaussian mean. The computed score is the proportion of tool-like pixels in the evaluated δ by $[a, b]$ patch.

(B): This question evaluates a series of template matches in order to estimate the precise location of the instrument entry point, Y_1. Centered on the boundary at the point $(a + b)/2$ and in increments of five degrees, we rotate a template of size $\delta \times 3W$, consisting of three $\delta \times W$ strips stacked together (*i.e.*, similar to [21]). At each rotation, we evaluate a template match, and then return maximum score observed over all evaluations.

The template match consists in evaluating a Haar-like feature [6] such that on each strip we sum the number of tool-like pixels using the color model described in question type (A), and subtract the sums of the two outer strips to that of the center region.

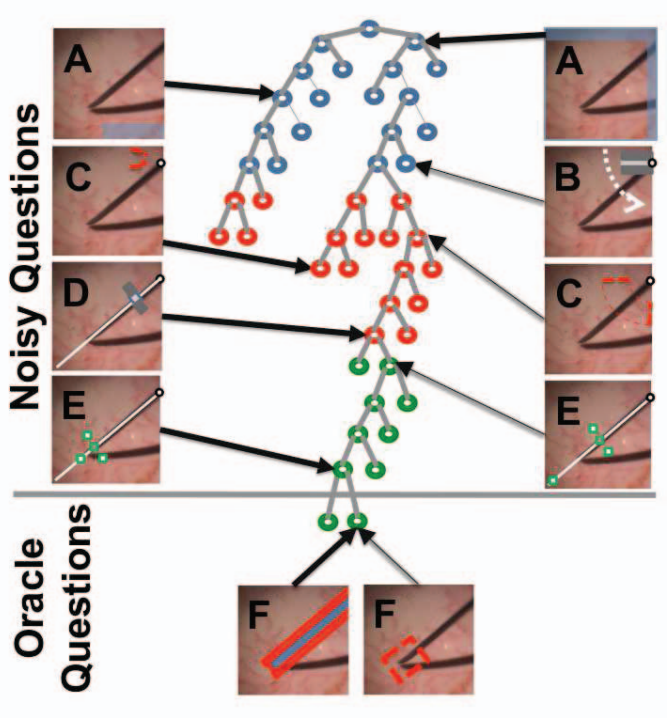

Fig. 2. Search space decomposition, density representation and image questions. The above tree represents a binary decomposition of the search space, where each node splits half the search space in two, by only spiting one coordinate of Y at a time. Blue, red and green nodes show when Y_1, Y_2 and Y_3 are being decomposed, respectively. By assigning the likelihood of the instrument parameters being within the search space of a given node, the tree S provides a representation of p_n. We also show examples of where different noisy question (types A through E) types can be evaluated, and what region of the image space they query. Oracle questions (type F) can only be evaluated at the leaf of the tree.

(C): This question also computes the proportion of tool-like pixels. As in (A), tool-like pixels are estimated by means of the same RGB color model. The region evaluated by this query type is defined by both $[a, b]$ and $[c, d]$. Defining the origin as the location on the boundary of the image $(a+b)/2$, we evaluate a restricted sector, by sweeping from c to d degrees and with length $\delta/2$ to δ to the origin. The proportion of tool-like pixels in this region is the computed score.

(D): Evaluates a template region, similar to that in (B), and returns the template matching score. A template of size $3W \times W$, consists of three $W \times W$ square regions stacked together. The template is positioned at a distance δ from boundary point $(a + b)/2$, and with an angle of $90 + (c + d)/2$ degrees (*i.e.*, perpendicular to the angle). The sum of tool-like pixels in the outer square regions is computed and subtracted to the sum in the inner square. Again, tool-like like pixels are computed as in (A).

(E): In this question, we evaluate a modified Haar-like feature. Along a line with intercept on the boundary at

$(a+b)/2$ and slope $(c+d)/2$, we position $W/2 \times W/2$ square regions at a distance of e and f pixels. The average intensity of each square is subtracted from each other. In addition, perpendicular to the slope and at a distance e to the boundary point, the average pixel intensity of two supplementary square regions of the same size are also computed and then subtracted to the previous two regions. This final score is then returned.

4.2.2 Oracle Questions

The second category of questions are assumed to be noiseless (*i.e.*, f_1 and f_0 do not overlap), and can only be evaluated at the leaves of our final tree, *i.e.*, when the pose of the instrument is explicitly hypothesized (see Fig. 2 question type (F)). Given that both the detection and tracking literature present a number of methods that appear to perform well in certain situations, we demonstrate the use of using two possible oracle questions: (i) template match which detects the pose of the instrument and (ii) gradient-based trackers typically used for traditional local tracking tasks. In our experiments, we show the effects and benefits of using either oracle:

Template Matching: We follow a similar approach to that of [21]. Given a specific hypothesis for the instrument location, we expect to find the instrument on the boundary at location $(a+b)/2$, with angle $(c+d)/2$ and length $(e+f)/2$. With the instrument width known, W, we perform a sum-of-squared difference (SSD) template match between the hypothesized pose and the projection of the tool-like color model on the image. That is, using the hypothesized pose, we construct an instrument template mask of width $3W$ and length $(e+f)/2+W$, with value 1 at instrument locations and 0 elsewhere. Placing the mask on the image, we apply the RGB color model to the overlapping regions of the image. The SSD is then computed from the projected tool-like pixel image and the constructed mask, and normalized by the total number of evaluated pixels. The final score is 1 if the normalized SSD is above a threshold and 0 otherwise.

Gradient-Based tracker: We use the recently developed Sum of Conditional Variance (SCV) objective function along with the ESM optimization strategy to refine the tool pose as proposed in [39]. The reference template used is an image patch describing the instrument tool tip, of size 40 by 40 pixels, extracted from the previous frame. Once the optimization scheme finishes, we apply a normalized cross-correlation template match (returning "yes or "no") to verify good convergence (which occurs when the score is above a threshold).

Naturally, many different classifiers or trackers could be substituted for those chosen here. Our aim is to show how to incorporate different oracles within our framework.

Note that if an oracle responds "yes" to any question regarding a hypothesized pose, then the posterior distribution becomes a Dirac (*i.e.*, all probability mass is concentrated on a single pose) because the noise models have non-overlapping support. Consequently, the entropy of the ensuing posterior distribution is zero and the algorithm terminates. Hence, having a good oracle questions is crucial to avoid the algorithm from finishing prematurely or erroneously.

4.3 Learning noise models

As described in the previous section, each node $S_{i,j} \in S$ has an associated question, $X_{i,j}$ with a corresponding noise model (densities (f_o, f_b) from (7)). Given that these densities are indexed by (i,j), it would appear as though a separate noise model for each node in S is required. Considering the size of the pose space, the quantity of training data to achieve this would be overwhelming.

In [17], the problem is somewhat avoided by using folded models, that take advantage of translational invariances within levels of the hierarchy. However, this trick is not possible in this setting given that the pose space is much larger and S does not maintain the same invariance properties.

To avoid the problem here, we propose to parametrize the noise models and interpolate the parameters based on the position of a node in the tree. For example, let us consider the answer to the noisy question of type (A) in Fig. 2. Given that the size of the object is known, we can expect to see a certain number of object-like pixels in the queried region. Similarly, if no object were present, then we would expect a much smaller number of object-like pixels to be found in the queried region. In addition, if the queried region were twice as large, the same intuition would still apply. For this reason our noise models are of the form,

$$\begin{aligned} f_b(x;i,j) &= \mathcal{G}(x;\mu_0|X_{i,j}|,\sigma_0|X_{i,j}|^2) \\ f_o(x;i,j) &= \mathcal{G}(x;\mu_1|Y|+\mu_0(|X_{i,j}|-|Y|), \\ &\quad \sigma_1|Y|^2+\sigma_0(|X_{i,j}|-|Y|)^2) \end{aligned} \tag{8}$$

where $\mathcal{G}(\cdot)$ is a Gaussian distribution, $|X_{i,j}|$ and $|Y|$ are the number of pixels contained in $X_{i,j}$ and the estimated instrument size in the image, respectively.

The parameters (μ_1,σ_1) and (μ_0,σ_0) are the means and variances for the likelihood of observing tool-like pixels for a given question type (assumed to be Bernoulli random variables). As such, any blue node in Fig. 2 has the same noise model as any other node of the same color, with its parameters interpolated based on its placement in the tree. Modeling the noise this way has the added benefit of being invariant to the fineness of the decomposition the pose space. For example, if the depth of the tree changed (*e.g.*, image is twice as large), we would not need to learn new noise models.

In practice, this type of noise model is learned for each of the noisy questions. While this clearly does not benefit questions of type (B), (D) and (E) (since they are always computed over the same sized area), the number of parameters needed to learn is greatly reduced for questions of type (A) and (C). As such, only four

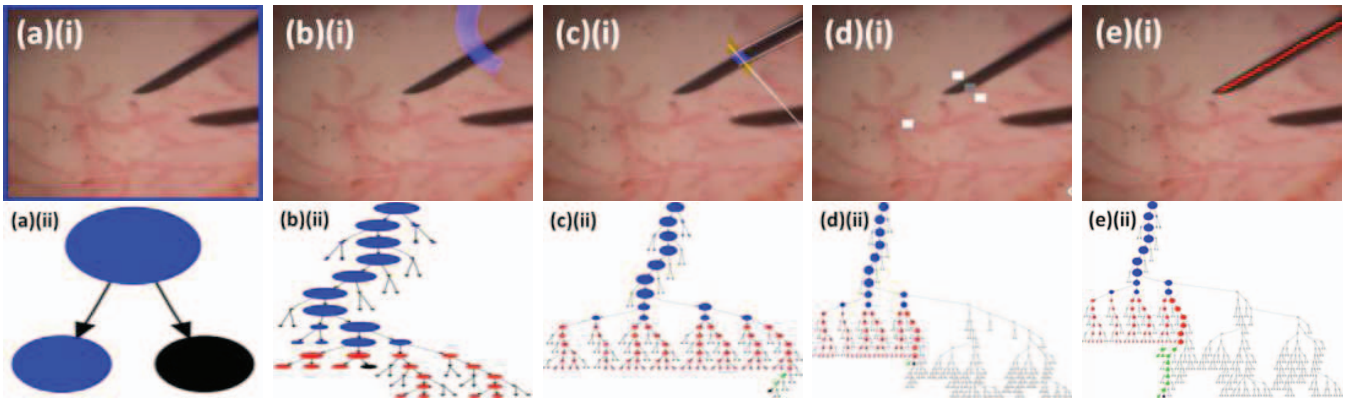

Fig. 3. Active Testing Iterations. Each image pair (top and bottom row) shows a question being evaluated and the corresponding state of the tree S at that point in time. See Video 1 for the entire image sequence.

parameters need to be learned for each question type and can be achieved by using an extremely small number of training images (*i.e.*, 20 labeled training images).

5 Experiments

In the following section we show how our approach performs on a live phantom eye platform, as well as on human *in-vivo* images. In both cases, we show qualitative and quantitative results of our method, and specify typical situations where our approach has difficulties maintaining accurate tracking.

Our framework was implemented on a Dell Precision PC with a Xeon 2.13GHz processor. The algorithm is coded in C++ and uses Opencv and the CISST library [40] for image acquisition and handling. Our PC is connected to receive video from a Grasshopper camera that is coupled to a microscope. The images acquired are 1600×1200 pixels large, and are captured at 30fps. The region of interest for the AT optimization is of size 256×256 and hence $Y_1 \in [1, L = 256 + (3 \times 255) = 1021]^2$.

The initial distribution of Y, $P(Y^0)$ is set to be an unbiased prior on the pose of the instrument. That is, $P(Y \in \mathcal{S}_1) = P(Y \in \mathcal{S}_0) = 1/2$, indicating that *a priori*, the instrument has equal likelihood of being in or not in the image. Note that we assign this probability at the root of S and assume uniform decomposition of the probability mass. While a small number of nodes may therefore be attributed with non-sensible probability, the practicality of this approximation is beneficial given that computing the exact probability is non trivial, and would be time consuming.

Finally, two versions of the algorithm are implemented. The first, **ATF-match**, uses the template match oracle question and the second, **ATF-track**, uses the gradient-based tracker oracle (as described in Sec. 4.2.2). With the exception of Sec. 5.1.2, where we observe the effect of different instrument motion models, we fix A to be the identity matrix (see Eq. (5)).

2. This is similar to the size of regions of interest during clinical procedures.

5.1 Phantom Eye Platform

We begin by providing some qualitative results as to how the proposed approach detects and tracks a surgical instrument in a phantom eye. To provide some intuition to the sequential nature of the AT algorithm, we have provided Video 1 (see additional videos) to visually depict both the questions asked and the evolution of S at each iteration of the AT optimization. Some snapshots of this video are shown in Fig. 3. The top row shows what question is being evaluated and the associated queried region (highlighted in each image) at given iterations of the optimization. The bottom row shows the corresponding evolution of the state of S. Here, the area of each node shown is proportional to the mass contained for that pose subset, and the color of each node represents which coordinate is being refined (as in Fig. 2). Additionally, the black node indicates which node is to be evaluated next.

Initially, only the root $S_{0,0}$ exists and is questioned. Having created children (Fig. 3(a)i-ii), the size of S is of three nodes. After a few questions, the tree has grown and refined itself past the first coordinate Y_1 and onto Y_2 and Y_3 (Fig. 3(b-c)i-ii). Eventually the correct Y_2 parameter (Fig. 3(d)i-ii) is refined, leading to a valid tool detection (Fig. 3(e)i-ii). Note that nodes which are extremely small are pruned as in [17]. This allows our search space to remain tractable and computationally manageable.

We also provided Videos 2-7, which show how our algorithm detects and tracks retinal instruments in our phantom environment. Fig. 4 shows a few snapshots from these sequences. The recorded sequences cover a wide range situations typically observed during retinal microsurgery: different types of instruments to track, severely blurred instruments, challenging non homogeneous illumination, no instrument in the field of view of the camera and the instrument shadow being present. In each image we have overlayed the AT search domain with a green box (except for (b) and (g) where the AT search domain is the entire image shown).

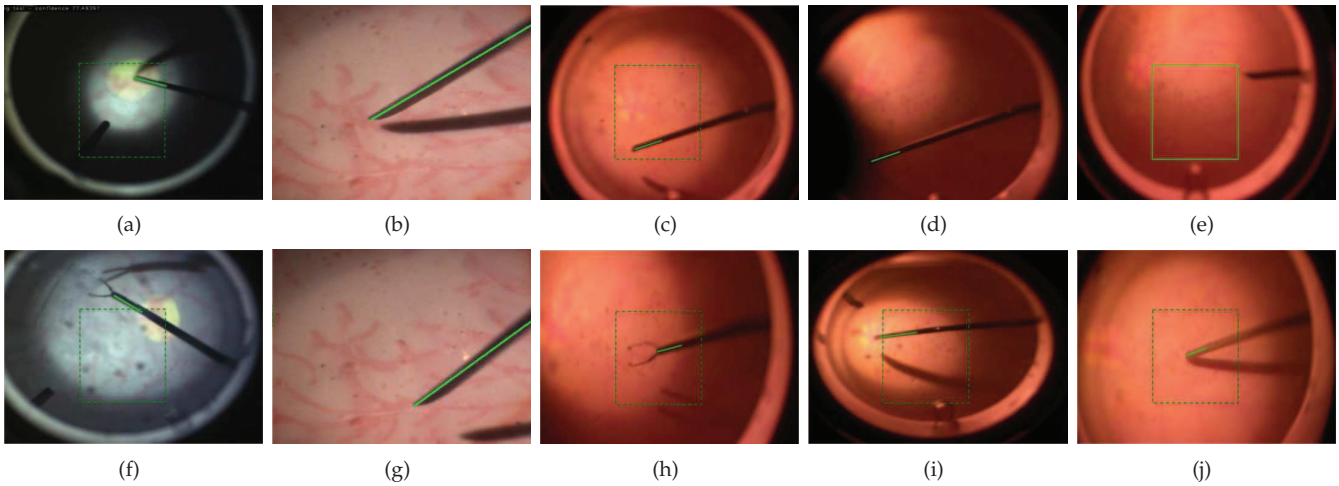

Fig. 4. Visual tracking examples in phantom environment. A variety of visual conditions typically encountered during clinical procedures are shown. The green box depicts the AT search region on our phantom platform.

5.1.1 Empirical Comparison

To evaluate the performance of the ATF approach, we compared it with several other methods: the Active Testing (AT-match) approach described above with a template match oracle but without any filtering (following [17], a Color-based Detector (CBD) as that in [41]), a Line-based tracker (LBT) (similar to [21]), and a particle filter [11] using our template matching oracle. We now further detail these methods.

The CBD [41] is a color based detector that requires a color model to evaluate the presence of tool-like pixels. Here, we set this color model to be the same as that used by our framework and at evaluation time, the instrument is segmented using this model. In addition to this, we also estimate the tool tip position by marching from the segmented centroid, along the direction of largest variance until a segmentation boundary is encountered. This position is considered to be the instrument tip. The LBT (following [21]) is a strip tracker that performs gradient based tracking on the color segmented image and a binary mask of the instrument. Tool motion is modelled with an image-plane rotation and translation vector. The particle filter [11] was set to use 1000 particles to maintain the distribution of the instrument, which was parametrized as in this application. The observation model consisted of using the same template match as in our framework, and used the same motion model as well. For both the LBT and the particle filter, initialization was performed by using the CBD.

To compare performances, we annotated by hand the location of the tool (*i.e.*, $Y = (Y_1, Y_2, Y_3)$) in an image sequence of over 400 images. These annotations provided ground truth for quantitative algorithm comparison. We then evaluate each approach by observing the error in the estimates of each parameter and the tool tip position, as well as the *true positive rate* (TPR), the *false positive rate* (FPR) and the precision for each approach (where a correct detection is where the estimated the tool tip location is within 10 pixels of the ground truth). The average time required by each method to find the location of the target for a single frame was also computed. Table 1 summarizes these results for each evaluated method. For the accuracy errors, we report the means and standard errors (in bracket) for each instrument parameter and tip.

In terms of coordinate accuracy, we can observe that the ATF methods generally performs better than the alternative methods. In particular, ATF-track performs better than other approaches when estimating the instrument tip position. This overall improvement can be attributed to the sequential parameter estimation approach that the active testing framework conducts. By estimating the first parameter, then the second and so on, each parameter is individually estimated accurately. This is in sharp contrast to the more direct LBT approach which locates the tool tip, and then estimates the necessary parameters, or the particle filter which simply samples the space directly.

In terms of detection accuracy, we notice that all methods tested provide more or less the same detection accuracy, with the exception of ATF-track which is significantly better than the others. This increase in precision is most likely due to the gradient-based tracker oracle question used. Also, we see that detection is significantly slower than tracking, as both AT-match and CBD run at much slower rates than the tracking algorithms. This confirms the advantage of tracking strategies over tracking by pure detection.

When comparing AT-match and ATF-match, we note that both methods perform similarly from an accuracy and detection point of view. However, we note that their speeds differ. Indeed, ATF-match is significantly faster than AT-match. This is most likely due to the use of informative priors. In fact, counting the number of nodes in final trees across all images, ATF-match trees have on average 75 nodes, while AT-match trees have around 210

TABLE 1
Comparison of algorithms. We show pixel accuracy of each method when estimating different parameters of the instrument, as well as the detection accuracy and time necessary for each method to process one frame.

	Accuracy Error				Detection Accuracy			
Method	Y_1	Y_2	Y_3	Tip	TPR	FPR (10^{-6})	Precision	Time (ms) per frame
ATF-track	4.13 (24)	2.42 (0.07)	**4.94 (0.27)**	**6.78 (0.6)**	**0.975**	**2.8**	**0.811**	8.0
ATF-match	2.97 (0.4)	2.37 (0.1)	14.11 (0.9)	11.03 (0.9)	0.839	6.6	0.631	8.54
AT-match	3.74 (0.4)	**2.01 (0.1)**	12.89 (0.8)	13.45 (0.8)	0.812	6.1	0.618	25.21
CBD	83.73 (2.4)	29.44 (0.8)	20.25 (0.7)	15.15 (0.79)	0.783	7.1	0.548	26.67
LBT	50.94 (7.3)	11.03 (0.9)	21.27 (2.1)	11.42 (0.2)	0.839	7.4	0.597	**4.8**
Particle Filter	**1.08 (0.05)**	11.53 (0.35)	6.64 (0.34)	6.91 (0.31)	0.841	2.9	0.783	6.28

nodes. This is a significant difference in the number of operations required to update the posterior distribution at each iteration of the AT optimization and accounts for the difference in speed between AT-match and ATF-match.

Given that our goal is to provide a tracking system, we would also like to have an understanding of how our system performs consecutively. To summarize this ability, we consider the event of correctly detecting a number of consecutive frames to follow a Geometric probability distribution. That is, with some probability ϵ, we correctly find the pose of the instrument in the next frame. Hence good tracking should be characterized by large values of ϵ. Computing this for each method, we find that ATF-tracker has the largest value with 0.98, followed by ATF-match (0.94), Particle Filter (0.93), LBD (0.92), AT(0.82) and CBD (0.82).

5.1.2 Alternative Tool Dynamics Model

We now briefly explore the effect of different instrument dynamics models (see (5)). As described in Sec. 3.2, we propose using two A matrices: (i) the identity (used until now) or (ii) augmented with velocity information.

Table 2 also shows a resume of the performance differences between the two proposed models. One can notice that in either case, the performances of the algorithms are extremely similar to each other. Most noticeably, we see that in terms of time, both methods run at approximately the same speed. This suggests that the dynamic models used in either case do not inhibit instrument localization and nor do they improve performance. This leads us to believe that the AT optimization ultimately is what provides timely solutions, rather than precise instrument motion models. Note that it could still be the case that alternative dynamics models could provide improvements in some cases.

5.2 Human In-Vivo Images

To validate the suitability of this approach for clinical settings, we evaluated our system on a human *in-vivo* image sequence. Our system was setup with the same parameters as previously described and then evaluated on 850 images. The initial 40 frames of the sequence were used for training purposes and were not included in the testing of our method.

Video. 8 show how our framework performs on this data and snapshots of this video are shown in Fig. 5. Here, we can see that even in situations where smoke is present, or when shadows overlap instrument regions considerably tracking is maintained and the instrument tip is accurately found. While this sequence is significantly more challenging than those acquired in our phantom experiments, reliable tracking is achieved for significant portions of this sequence.

However, as shown in Fig. 5, there are situations where our system fails to provide correct instrument pose. In particular, we can identify two such causes:

- Tool appearance changes due partial illumination variation. In some cases, the illumination on the instrument is not regular. Coupling this with the instrument tip appearing blurry (out of focus), our algorithm has difficulties precisely localizing the instrument tip, as depicted in Fig. 5(e).
- Poor oracle question. When using the gradient-based tracker oracle question, a threshold is used to validate valid convergence. Incorrect thresholds can lead to saying that the instrument is at a particular location when it is in fact not. As shown in Fig. 5(f), this may lead to being "stuck" on irrelevant image regions.

To relate the effectiveness of our method in this scenario to that reported on phantom data, we computed similar performance measures as done previously[3]. In all categories computed AT-track performed better than ATF-match (TPR; 0.6 vs 0.1. FPR; 5.1 vs 7.8 $\times 10^{-6}$. Precision; 0.49 vs 0.12. Accuracy Tip; 37.3 vs 82.3. ϵ; 0.65 vs 0.49.). These results indicate two distinct points. First, the task of detecting and tracking instruments is substantially more difficult in *in-vivo* sequences than in phantom sequences and is apparent from the drop in performances across all measures when compared to Table. 1. Second, the template match oracle, ATF-match, is in effect not capable of accurately detecting and tracking the instrument in this sequence. For this reason, in challenging tracking tasks such as the one at hand, the possibility of relying on successful gradient-based trackers is of great benefit.

3. Note that the CBD could not locate the instrument in an overwhelming number of images in the *in-vivo* sequence. Given that it initializes both the LBT and the particle filter, quantitative evaluation of these methods has been omitted here.

TABLE 2
Comparison of instrument dynamics models. Two instrument transition models, A, are tested.

A	Identity	Augmented	A	Identity	Augmented
Y_1	2.97 (0.4)	**2.63 (0.1)**	TPR	0.839	**0.842**
Y_2	2.37 (0.1)	**1.72 (0.08)**	FPR (10^{-6})	6.6	**6.4**
Y_3	14.11 (0.9)	**11.65 (0.7)**	Precision	0.631	**0.662**
Tip	11.03 (0.9)	**10.51 (0.7)**	Time (ms)	8.54	**7.21**

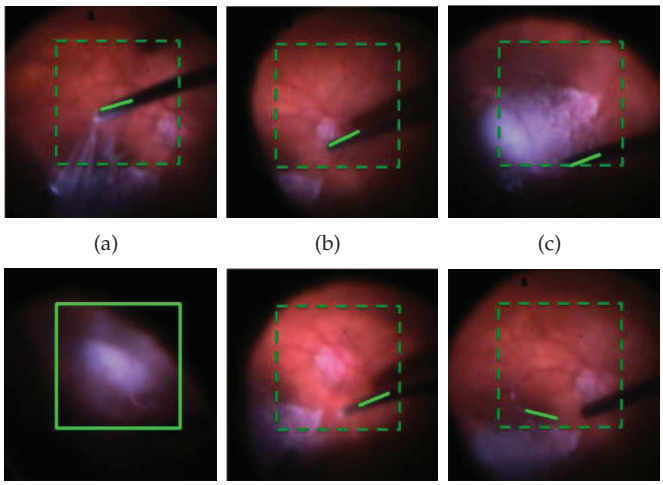

Fig. 5. Visual tracking example in a human *in-vivo* image sequence. The green region depicts the region considered by the AT optimization. (e-f) show two different cases where our system fails (see text for details).

6 Conclusion and Future Work

In this paper, we have a proposed a novel approach for the task of instrument detection and tracking in retinal microsurgery. By using the Active Testing paradigm, both these tasks can be treated as the same sequential parameter estimation problem, as opposed to two separate algorithmic tasks. Using filtering techniques, we have also shown how to effectively incorporate previous instrument information for the task of tracking. We have experimentally shown that the presented algorithm is capable of detecting and tracking retinal tools efficiently and robustly in cases where the object enters and leaves the field of view frequently. This has been demonstrated on both a live platform and on human *in-vivo* images. While presented in the context of retinal microsurgery, we are confident that this approach may apply to other surgical procedures, as well as for other object categories. Future work in this area will be directed to extending this method to stereo image sequences, as well as modeling illumination changes more consistently.

Acknowledgment

Funding for this research was provided in part by NIH Grant R01 EB 007969-01 and internal JHU funds. We would also like to thank Dr. Jim Handa MD and Dr. Peter Gehlbach MD for their insightful help. Marcin Balicki and Kevin Olds are credited for the development of the phantom eye used.

References

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Survey*, vol. 38, no. 4, 2006.

[2] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2000, pp. 142 –149.

[3] D. Geman and B. Jedynak, "An active testing model for tracking roads from satellite images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 1, pp. 1–14, 1996.

[4] O. Bernier, P. Cheung Mon Chan, and A. Bouguet, "Fast non-parametric belief propagation for real-time stereo articulated body tracking," *Journal of Computer Vision and Image Understanding*, vol. 113, no. 1, pp. 29–47, 2009.

[5] W. Geng, P. Cosman, C. C. Berry, Z. Feng, and W. R. Schafer, "Automatic tracking, feature extraction and classification of c. elegans phenotypes," *IEEE Transaction on Biomedical Engineering*, vol. 51, pp. 1811–1820, 2004.

[6] P. Viola and M. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[7] A. Vedaldi, G. Gulshan, M. Varma, and A. Zisserman, "Multiple kernels for object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 606–613.

[8] C. Lampert, M. Blaschko, and T. Hofmann, "Beyond sliding windows: Object localization by efficient subwindow search," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[9] D. Aldavert, A. Ramisa, R. Toledo, and R. Mantaras, "Fast and robust object segmentation with the integral linear classifier," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2010, pp. 1046–1053.

[10] R. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

[11] M. Isard and A. Blake, "Condensation - conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 28, no. 1, pp. 5–28, 1998.

[12] M. Andriluka, S. Roth, and B. Schiele, "People-tracking-by-detection and people-detection-by-tracking," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[13] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, "Robust tracking-by-detection using a detector confidence particle filter," in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 1515–1522.

[14] S. Shahed Nejhum, J. Ho, and Y. Ming-Hsuan, "Visual tracking with histograms and articulating blocks," in *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[15] R. Verma, C. Schmid, and K. Mikolajczyk, "Face detection and tracking in a video by propagating detection probabilities," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1215–1228, 2003.

[16] K. Toyama and G. D. Hager, "Incremental focus of attention for robust vision-based tracking," *International Journal of Computer Vision*, vol. 35, no. 1, pp. 45–63, 1999.

[17] R. Sznitman and B. Jedynak, "Active testing for face detection and localization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 10, pp. 1914–1920, 2010.

[18] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework part 1: The quantity approximated, the warp update rule, and the gradient descent approximation," *International Journal of Computer Vision*, vol. 56, no. 3, pp. 221–255, 2004.

[19] R. Richa, M. Balicki, E. Meisner, R. Sznitman, R. H. Taylor, and G. D. Hager, "Visual tracking of surgical tools for proximity detection in retinal surgery," in *Information Processing in Computer Assisted Interventions*, vol. 6689, 2011, pp. 55–66.

[20] R. Sznitman, H. Lin, M. Gupta, and G. Hager, "Active background modeling: Actors on a stage," in *International Conference on Computer Vision Workshops*, 2009, pp. 1222–1228.

[21] D. Burschka, J. Corso, M. Dewan, W. Lau, H. Li, H. Lin, P. Marayong, N. Ramay, G. Hager, B. Hoffman, D. Larkin, and C. Hasser, "Navigating inner space: 3-d assistance for minimally invasive surgery," *Robotics and Autonomous Systems*, vol. 52, pp. 5–26, 2005.

[22] D. Uecker, C. Leem, Y. Wang, and Y. Wang, "Automated instrument tracking in robotically assisted laparoscopic surgery," *Journal of Image Guided Surgery*, vol. 22, no. 6, pp. 429–437, 1995.
[23] Y. Wang, D. Uecker, and Y. Wang, "A new framework for vision-enabled and robotically assisted minimally invasive surgery," *Computerized Medical Imaging and Graphics*, vol. 1, no. 6, pp. 308–325, 1998.
[24] C. Doignon, F. Nageotte, and M. de Mathelin, "Detection of grey regions in color images: application to the segmentation of a surgical instrument in robotized laparoscopy," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 4, 2004, pp. 3394–3399.
[25] S. J. McKenna, H. Nait-Charif, and T. Frank, "Towards video understanding for laparoscopic surgery: instrument tracking," in *Image and Vision Computing New Zealand Conference*, 2005.
[26] J. Climent and P. Mares, "Automatic instrument localization in laparoscopic surgery," *Electronic Letters on Computer Vision and Image Analysis*, vol. 4, no. 1, pp. 21–31, 2004.
[27] S. Voros, J. A. Long, and P. Cinquin, "Automatic detection of instruments in laparoscopic images: A first step towards high-level command of robotic endoscopic holders," *International Journal of Robotic Research*, vol. 26, no. 11-12, pp. 1173–1190, 2007.
[28] A. Casals, J. Amat, and E. Laporte, "Automatic guidance of an assistant robot in laparoscopic surgery," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1996, pp. 895–900.
[29] G. Wei, K. Arbter, and G. Hirzinger, "Real-time visual servoing for laparoscopic surgery. controlling robot motion with color image segmentation," *IEEE Engineering in Medicine and Biology Magazine*, vol. 16, no. 1, pp. 40–45, 1997.
[30] O. Tonet, T. U. Ramesh, G. Megali, and P. Dario, "Image analysis-based approach for localization of endoscopic tools," in *Proceedings of Surgetica*, 2005, pp. 221–228.
[31] Z. Pezzementi, S. Voros, and G. Hager, "Articulated object tracking by rendering consistent appearance parts," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2009, pp. 3940–3947.
[32] M. Dewan, P. Marayong, A. M. Okamura, and G. D. Hager, "Vision-based assistance for ophthalmic microsurgery," in *Medical Image Computing and Computer Assisted Intervention*, 2002, pp. 49–57.
[33] R. Sznitman, A. Basu, R. Richa, J. Handa, P. Gehlbach, R. T. H., B. Jedynak, and G. D. Hager, "Unified detection and tracking in retinal microsurgery," in *Medical Image Computing and Computer Assisted Intervention*, 2011, pp. 1–8.
[34] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.
[35] G. Hager, *Task-Directed Sensor Fusion and Planning: A Computational Approach*. Springer, 1990.
[36] N. S. Peng, J. Yang, and Z. Liu, "Mean shift blob tracking with kernel histogram filtering and hypothesis testing," *Pattern Recognition Letters*, vol. 26, no. 5, pp. 605–614, 2005.
[37] T. M. Cover and J. A. Thomas, *Elements of information theory*. Wiley-Interscience, 1991.
[38] J. Pakkanen, J. Iivarinen, and E. Oja, "The Evolving Tree — a novel self-organizing network for data analysis," *Neural Processing Letters*, vol. 20, no. 3, pp. 199–211, 2004.
[39] R. Richa, R. Sznitman, R. H. Taylor, and G. D. Hager, "Visual tracking using the sum of conditional variance," in *IEEE Conference on Intelligent Robots and Systems*, 2011, pp. 2953–2958.
[40] P. Kazanzides, S. DiMaio, A. Deguet, B. Vagvolgyi, M. Balicki, C. Schneider, R. Kumar, A. Jog, B. Itkowitz, C. Hasser, and R. H. Taylor, "The surgical assistant workstation (saw) in minimally invasive surgery and microsurgery," in *MICCAI Workshop - Systems and Architectures for Computer Assisted Interventions*, 2010.
[41] R. Sznitman, D. Rother, J. Handa, P. Gehlbach, G. D. Hager, and R. H. Taylor, "Adaptive multispectral illumination for retinal microsurgery," in *Medical Image Computing and Computer Assisted Intervention*, 2010, pp. 465–472.

Raphael Sznitman received his B.Sc. in cognitive systems from the University of British Columbia in 2007. Following this, he received his M.Sc and PhD in computer science from the Johns Hopkins University in 2011. Currently, he is a postdoctoral fellow at the Ecole Polytechnique Federale de Lausanne (Switzerland) where he works in the computer vision laboratory. His research interests are primarily in computational vision, probabilistic methods and statistical learning, applied to visual tracking and object detection.

Rogerio Richa Rogerio Richa received his Ph.D. from the LIRMM (Laboratoire d'Informatique, Robotique et Microelectronique de Montpellier) in 2010 for his work in robust visual tracking for beating heart surgery. He then joined the LCSR (Laboratory of Computational Sensing and Robotics) at Johns Hopkins where he is currently working on the development of a robotic platform for retinal surgery. His main interests are computer vision, medical imaging and surgical robotics.

Russell H. Taylor Russell H. Taylor (Fellow, 1994) received his Ph.D. in Computer Science from Stanford in 1976. He joined IBM Research in 1976, where he developed the AML robot language and managed the Automation Technology Department and (later) the Computer-Assisted Surgery Group before moving in 1995 to Johns Hopkins, where he is a the John C. Malone Professor of Computer Science with joint appointments in Mechanical Engineering, Radiology, and Surgery and is also Director of the Engineering Research Center for Computer-Integrated Surgical Systems and Technology (CISST ERC). He is the author of over 275 peer-reviewed publications, a Fellow of the IEEE, of the AIMBE, of the MICCAI Society, and of the Engineering School of the University of Tokyo. He is also a recipient of numerous awards, including the IEEE Robotics Pioneer Award, the MICCAI Society Enduring Impact Award, and the Maurice Müller Award for Excellence in Computer-Assisted Orthopaedic Surgery.

Bruno Jedynak received his doctorate in Applied Mathematics from the University Paris Sud. His dissertation was performed at INRIA (Rocquencourt, France). After spending a year as post-doc in the Department of Statistics at the University of Chicago, he was appointed Maitre de conferences at the Universite des Sciences et Technologies de Lille. He is currently a faculty member of the Department of Applied Mathematics and Statistics at The Johns Hopkins University (JHU). He is also a faculty member of the Center for Imaging Science at JHU.

Gregory D. Hager Gregory D. Hager is a Professor and Chair of Computer Science at Johns Hopkins University and the Deputy Director of the NSF Engineering Research Center for Computer Integrated Surgical Systems and Technology. His research interests include time-series analysis of image data, image-guided robotics, medical applications of image analysis and robotics, and human-computer interaction. He is the author of more than 220 peer reviewed research articles and books in the area of robotics and computer vision. In 2006, he was elected a fellow of the IEEE for his contributions in Vision-Based Robotics.