Sequence-Level Object Candidates Based on Saliency for Generic Object Recognition on Mobile Systems

Esther Horbert[1]*, Germán M. García[2]*, Simone Frintrop[2], Bastian Leibe[1]

***Abstract*— In this paper, we propose a novel approach for generating generic object candidates for object discovery and recognition in continuous monocular video. Such candidates have recently become a popular alternative to exhaustive window-based search as basis for classification. Contrary to previous approaches, we address the candidate generation problem at the level of entire video sequences instead of at the single image level. We propose a processing pipeline that starts from individual region candidates and tracks them over time. This enables us to group candidates for similar objects and to automatically filter out inconsistent regions. For generating the per-frame candidates, we introduce a novel multi-scale saliency approach that achieves a higher per-frame recall with fewer candidates than current state-of-the-art methods. Taken together, those two components result in a significant reduction of the number of object candidates compared to frame level methods, while keeping a consistently high recall.**

I. Introduction

The field of visual object recognition is currently undergoing a major paradigm shift. There has been tremendous progress both on an image classification [1] and on a category detection level [2], [3]. Approaches are now available that can reliably detect a small number of object categories in complex scenes [2] or that can recognize the most prominent objects in web images from a large number of classes [1], [3]. Still, the recognition problem is far from solved. Ironically enough, this is most visible when considering the problem of recognizing everyday objects in a continuous video stream that roughly emulates what a human or mobile robot sees when moving through a common household scene (see Fig. 1). In such a scenario, there are simply so many possible objects that it is hard to come up with an exhaustive set of categories for which specific detectors could be trained. In addition, those objects are typically not the central motive of a photograph (as in many recognition benchmarks), but they may be just another (small) part of a cluttered scene. As a result, the hitherto dominant paradigm of window-based classification with exhaustive search is reaching its limits.

A recent trend is therefore to invert the recognition pipeline and first generate a set of category independent object proposals [4], [5], [6], [7], [8], [9] to support and guide object search [9], [10]. However, most current approaches still need to generate hundreds of proposal regions per image in order to achieve a high recall. In contrast, saliency-based approaches have been proposed with the goal of finding and segmenting a single, prominent object in web images [11], [12], [13], [14]. Those approaches typically generate proposal regions that adhere better to object boundaries, but their saliency formulation limits them to finding one or at best very few objects in an image; they cannot be applied for finding *all* objects in a cluttered scene.

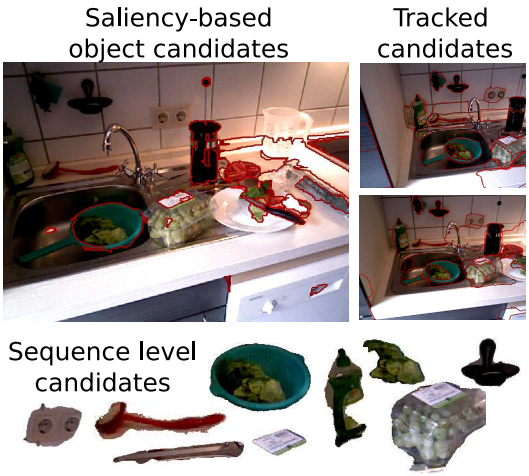


Fig. 1: Overview of our sequence-level object candidate detection. (Left) Candidates from our frame-based, multi-saliency object discovery. (Right) Tracked candidates. (Bottom) Visualization of some sequence-level candidate objects.

In this paper, we want to think this trend further. As object candidate generation methods become increasingly accurate, we envision that large-scale recognition from video will turn into a retrieval problem, where a low-level module processes the incoming video stream and generates object candidates that are sent as queries to a (possibly cloud-based) recognition service. Concrete application scenarios could be a wearable camera (*e.g.*, a Google Glass like device) that recognizes objects in the user's field of view or a mobile service robot that performs everyday tasks in people's homes. In such a setting, it is not necessary that every object is recognized in every video frame. Rather, the number of recognition queries will quickly become the main cost factor. It is thus desired that this number be as low as possible, while covering all relevant scene objects by at least one query.

We therefore propose to address the object candidate generation problem on the level of entire video sequences. Instead of generating a large set of proposal regions for each frame, we are interested in reporting a small and consolidated number of object candidates for an entire video. For this, we take advantage of the temporal coherence of video input in order to track and evolve region candidates over time.

[1]are with the Computer Vision Group, RWTH Aachen, Germany {horbert, leibe}@vision.rwth-aachen.de
[2]are with the Institute of Computer Science III, University of Bonn, Germany {martin, frintrop}@iai.uni-bonn.de
*indicates equal contribution

We start from a set of region candidates for each frame and build upon a fast segmentation based low-level tracker [15] to propagate each of those regions independently over the next frames. Tracking fulfills a dual purpose in this procedure. First, it allows us to link independently extracted candidates from different frames that correspond to the same object. Second, it acts as a natural filter to improve proposal quality. Single-frame candidates often extend beyond object boundaries due to bad color contrast or suboptimal object viewpoint. When a tracker is initialized to such a proposal region, it will quickly diverge when the camera moves due to parallax effects. We take advantage of this effect through a series of consistency checks, terminating bad tracks already at an early stage. We then rank the remaining region tracks using shape, contrast, and tracking quality criteria and condense them into a set of sequence-level object candidates. As our experiments will show, this greatly reduces the number of object candidates, while keeping a consistently high recall.

In detail, this paper makes the following contributions: 1) We propose a novel approach for sequence-level object proposal generation from video. Starting from a set of candidate regions extracted from each frame, our approach tracks each candidate region independently over time and selects the best representatives among the tracked set. 2) In order to generate the per-frame object candidates, we present a novel method based on multi-scale saliency that achieves a higher per-frame recall with fewer candidates than current state-of-the-art methods [4], [8]. 3) We demonstrate that the combination of those two approaches results in a concise scene summary consisting of a small number of high-quality sequence-level object candidates that could be used as queries to a recognition service. 4) We present a new benchmark dataset for object discovery from video consisting of very challenging video sequences of cluttered scenes with detailed object annotations and use this dataset to compare our approach to the state-of-the-art.

The paper is structured as follows. The next section discusses related work. Sec. III then gives an overview of our approach and highlights the main design goals. Sec. IV presents our saliency-based object proposal generation approach, after which Sec. V describes the proposed pipeline for tracking and consolidating object candidates over time. Experimental results are reported in Sec. VI.

II. Related Work

Object Discovery. The capability to discover and segment unknown objects is of considerable interest for many applications in mobile robotics [16], autonomous vehicles [17] and general visual scene analysis [18]. Many approaches in those areas either assume an active camera [19], [20], active interaction with potential objects [21], [22], or make use of 3D cues from an RGB-D sensor [5], [23], [18], [24], [25]. Our focus is on extracting object hypotheses from the video stream of a moving, monocular camera (*e.g.*, from a Google Glass-like setup), where we cannot control the camera motion and we do not have ready access to 3D information. We therefore concentrate here on approaches that are only based on (monocular) vision.

Many approaches for object discovery in images proceed by sampling a set of candidate regions and ranking them according to their "objectness", *i.e.*, to the likelihood that the region corresponds to a full object [4], [6], [7], [8], [9]. E.g., [4] randomly sample bounding boxes to define the candidate regions and rank them using a Naive Bayes framework combining global saliency, color contrast, edge density, and location cues. [6] generate multiple figure/ground segmentations by solving a constrained parametric min-cuts problem from a grid of seed points and learn a ranking classifier based on Gestalt cues. [7] create occlusion boundary based seed regions [26] and group them using a learned affinity measure between regions. They then use structured learning for ranking. [9] follow a similar strategy, but use a variety of complementary grouping criteria and color spaces starting from superpixels [27] to sample more diverse proposals. [8] also start from superpixels [27] and randomly group connected regions by sampling partial spanning trees that have high sums of edge weights. The main effort in those approaches is spent on learning a good prediction model to rate the "objectness" of a segment based on a set of pixel or region based cues. In our work, we use the much simpler idea that the content of a good segment should differ from the content of the surrounding region. This is a property that is captured by the center-surround contrast that is at the heart of saliency approaches.

Object Saliency Criteria. Saliency and visual attention have been intensely investigated for decades in human perception as well as in computer vision and robotics. While early computational models have been mainly designed to simulate human eye movements [28], interest has recently increased to use saliency for object candidate generation as a preprocessing step for classification. However, the main focus has so far been on web images [11], [12], [13], [29], which often exhibit photographer bias. Many saliency methods have taken advantage of the special properties of such images, *e.g.*, that objects are often large and rarely intersect with the image borders [12], [29]. For mobile cameras, such assumptions fail and we need methods that work well without them.

The key element of saliency methods is usually a measure of center-surround contrast. While this was traditionally addressed with biologically inspired Difference-of-Gaussian methods [28], several other methods were recently proposed to compute this contrast. For example, [30], [31], [13] use information theory to compute the difference between center and surround distributions. Other methods compute the center-surround contrast on superpixels [32], [33]. This is especially useful when detecting salient objects, which is a combination of saliency computation and segmentation. Other approaches address this task by applying a segmentation method to salient blobs, *e.g.*, Graph Cuts [12]. We follow a different approach here by computing segmentation and saliency independently and using saliency to select the segments that form an object candidate.

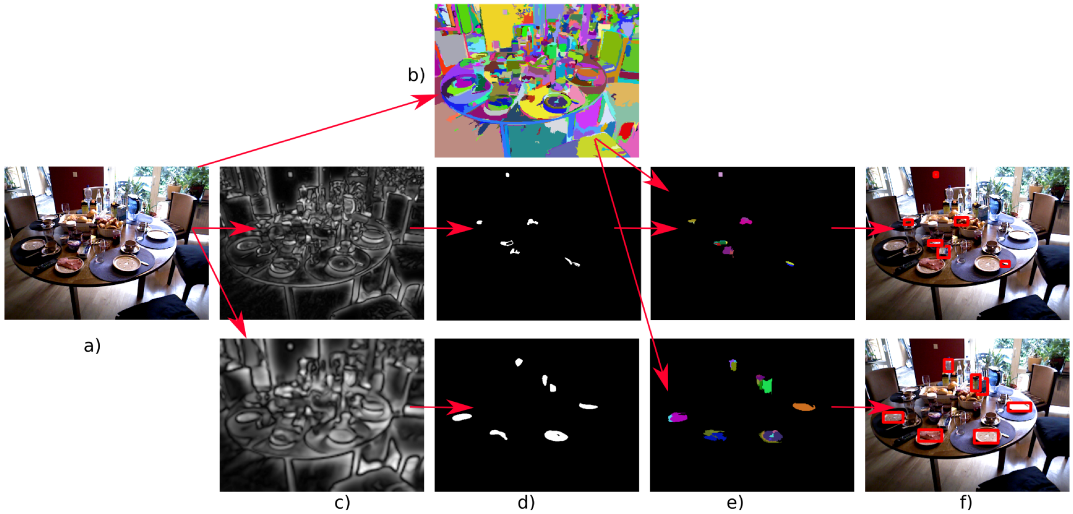


Fig. 2: Object candidate generation: a) original image, b) superpixel segmentation c) for octaves 1 and 2, their specific saliency maps, d) salient blobs obtained by region-growing, e) combining superpixels into object candidates with help of salient blobs, and e) bounding boxes of the object candidates.

A limitation of the above saliency approaches is that they are targeted at finding the most salient object in a scene in the sense of a global pop-out measure. In order to make saliency usable for our purpose, we therefore need to adapt the saliency formulation such that it yields candidates that cover as many objects in an image as possible. We achieve this using a novel multi-scale saliency described in Sec. IV.

III. Overview of our Approach

The goal of our approach is to generate a concise set of object candidates that cover the relevant objects in a cluttered scene (see Fig. 1).[1] Contrary to existing literature [4], [5], [8], however, we address the problem at the sequence instead of at the frame level. Thus, the main conceptual novelty is in how per-frame region candidates are used to obtain sequence-level results.

Fig. 1 outlines the main steps of our approach. For each frame of the input video sequence, we extract the top-k proposal regions using the multi-scale saliency method described in Sec. IV. We then track all candidates over time and use the tracker confidence as an additional proposal quality criterion. If a tracked region stays consistent over significant camera motion, this is a strong indicator that the tracked region contour indeed corresponds to a valid object boundary. After each frame, we check for duplicates and allow new region candidates to supersede existing ones if they exhibit a better quality score. Thus, our approach builds up a set of region trajectories that span different viewpoints, increasing the chance for valid objects to be nicely delineated from their neighbors. From the final set of trajectories, we then report the best-scoring candidates and select a representative view for each of them that could be used as a query for recognition. The following sections flesh out this pipeline in more detail.

[1] As *relevant objects* we consider "manipulable units with internal coherence and external boundaries", as in the psychological definition in [34].

IV. Generating Object Candidates

We first describe our method to generate object candidates based on a combination of segmentation and saliency. The method is an extension of our previous work [35]. Our new method contains an improved salient blob detection and a split-octave mode which is especially useful for challenging scenes with multiple objects. The object candidates are generated in three steps: first, we segment the image into perceptually coherent regions (superpixels); second, we compute a saliency map that highlights salient image regions; finally, saliency is used to select and combine superpixels that form an object proposal. An overview of the approach is shown in Fig. 2.

Superpixel Segmentation. We use the popular graph-based superpixel segmentation method by [27] as the basis for our object candidates (cf. Fig. 2(b)). We chose the parameter k, which determines the scale of observation, to slightly over-segment the image, since assembling superpixels is later accomplished by saliency selection.

Multi-Scale Saliency Computation. As the basis for our saliency computation, we chose the method proposed in [35], which has been shown to outperform several other state-of-the-art saliency methods.[2] The method has the advantage that it computes precise saliency maps, it works for large as well as for small objects, it is applicable to web images as well as to video frames, and it is real-time capable. Briefly stated, the saliency computation works as follows. Operating on a scale-space structure (Gaussian pyramid, 2 scales and 4 octaves), computations are performed for intensity and color features. For both, center and surround contrasts are computed by Difference-of-Gaussians for different sizes at each pixel location. Color features are computed in an opponent-color space with a red-green and a blue-yellow axis.

[2] Code for an updated, faster re-implementation: http://www.iai.uni-bonn.de/∼frintrop/vocus2.html

In this paper, we now extend the method from [35] to a multi-scale approach with split octaves. For this, we regard the octaves of the scale-space structure independently instead of fusing them into a single saliency map. Since objects of different sizes will achieve the strongest response at different octaves, this enables us to detect nested candidates. *E.g.*, if an apple lies on a plate, one octave will highlight the apple and another one the plate, resulting in candidates for both objects. Thus, octave-specific saliency maps result in more candidates, finding also difficult objects which are otherwise missed. In the following, we call the octave-specific saliency maps S_l, with layers (octaves) $l \in [1, .., 4]$, where each map is the sum of the two scale maps of that layer. Examples of two octave-specific saliency maps can be seen in Fig. 2(c). In Sec. VI, we will show that while the single saliency map approach performs better with few candidates per frame, we achieve higher recall for a larger number of candidates. Thus, when aiming at finding all objects in a cluttered scene, the split-octave method is preferable.

The next step for proposal generation is to extract salient blobs from the saliency maps. In contrast to current state-of-the-art saliency methods, we are interested in finding a large number of objects per frame. Thus, simple thresholding of the saliency maps, which works reasonably well for images with one or a few prominent objects per frame, is not sufficient. Instead, we extract several blobs of different extents at each local maximum in the maps S_l. Thus, we determine the local maxima $\{m_1, ..., m_N\}$ within each octave-specific saliency map S_l, where $m = (m_x, m_y)$. After ranking the maxima by their saliency $S_l(m_x, m_y)$, seeded region growing [36] is applied to each of the maxima, starting from the most salient one, to obtain salient blobs. We denote the salient blob of m as B_m. The region growing step takes each maximum m as a seed and recursively investigates its neighboring pixels: if the saliency of a neighbor pixel p is above a threshold, the pixel is assigned to B_m. The threshold is set relative to the saliency of m. This means, p is assigned to B_m if $S_l(m_x, m_y) \geq S_l(p_x, p_y) \geq t \cdot S_l(m_x, m_y)$. We use two different thresholds: $t = 0.6$ and $t = 0.7$. The result of this method is a set of two nested salient blobs of different extents for each local maximum. This method differs from [35], where adaptive thresholding is used to extract blobs from the saliency map. As our experiments will show, the region growing approach obtains more candidates, with higher precision as well as higher recall, when considering more than about 30 candidates per frame.

Combining the salient blobs of all maxima, we obtain for each octave a set of salient regions which are used in the next section to form object candidates. Some of the salient blobs are displayed in Fig. 2(d).

From Superpixels to Object Candidates. To generate object candidates, we combine several superpixels based on their overlap with the salient blobs. Every superpixel that is covered by a salient blob by at least 30% is chosen to belong to the candidate formed by this salient blob. Next, we perform a non-maximum suppression step where we discard candidates that overlap strongly with other candidates (more than 80% in both ways). Some examples of candidates that are obtained this way are shown in Fig. 2(e). Finally, the set of all candidates for one image is obtained by combining the candidates of all octaves. Since a saliency value can be assigned easily to each proposal by computing the average saliency of the corresponding salient blob, the candidates can be ranked. This is especially useful for applications in which real-time criteria require a prioritization of candidates.

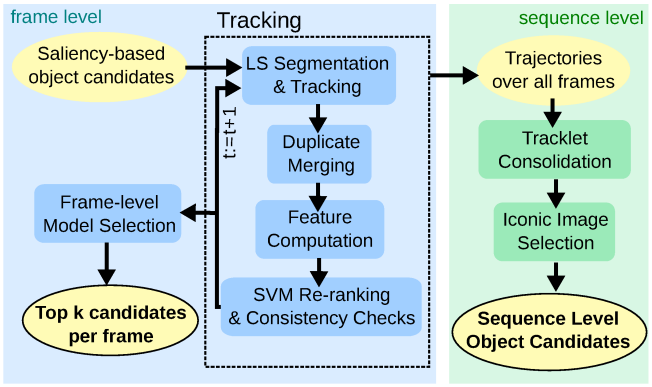


Fig. 3: Overview of the tracking pipeline for obtaining sequence-level object candidates. Moreover, frame-level candidates are computed as intermediate result for comparison.

V. Tracking Object Candidates

When working with a video instead of single images, the objects are visible in more than one frame. With applications in mind, it seems redundant to examine each individual object anew in each frame. We therefore propose to track object candidates in the video, with the goal of finding all individual objects in the complete scene regardless of which and how many frames they are visible in.

Fig. 3 shows an overview of the proposed tracking pipeline. We initialize a tracker with each saliency-based object candidate, track the candidates through the video and finally select sequence-level candidates with the help of an objectness classifier.

A. Tracking

For each frame, we initialize new tracks using the N highest scoring object candidates. Since two candidates might cover the same area in the first or in later frames, we check for duplicates and merge strongly overlapping tracklets. Next, we compute a set of quality features for each tracked region combining appearance cues and tracking confidence. We terminate tracklets if the tracking has clearly failed and we then score and re-rank all remaining tracklets using an RBF-kernel SVM. Tracklets which score negatively multiple times in a row are also discontinued (this counter is reset when there is a new candidate for the same region). Whenever intermediate results are required for evaluation, we perform frame-level model selection (non-maximum suppression based on bounding box overlap) to obtain the top-k candidates. The following paragraphs will describe each of these stages in more detail.

Region features
Object dimensions: width, height, aspect ratio.
Color contrast: χ^2 distance of color histograms [4]
Region symmetry: Maximum overlap of both contour halves (over 10 rotations of center axis).
Color symmetry: Minimal χ^2 distance of color histograms (over 10 rotations)
Contour convexity: #pixels in region / #pixels in its convex hull.
#Non-empty bins in color histogram.
Tracklet features
Tracker confidence: *c.f.* Section V-A.
Bwd tracker confidence: when tracking the current region backwards into the previous frame.
Bwd tracking overlap: Overlap between last contour and backwards-tracked contour.

TABLE I: Frame level and track level features used for ranking tracked regions.

Trajectory Initialization and Duplicate Merging. In each frame, we use N object candidates to initialize new tracklets. We filter out candidates that are thinner than 10 pixels and initialize the tracker with a region 4 pixels larger than the candidate. In case there already is a tracklet which strongly overlaps with the new candidate, no new tracklet is started. Moreover, since tracklets can evolve to the same region, we merge tracklets that overlap significantly. In our implementation we use an overlap threshold of intersection-over-union (IOU) $> 70\%$.

Level Set Segmentation and Tracking. For tracking region candidates, we use the segmentation-based level set approach from [15]. This probabilistic framework segments and tracks regions using their color distribution. It has been shown to be very fast and robust to motion blur, appearance changes, and rapid camera movement. It is particularly suitable for our task, since it does not only track the position, but also the region of the target object. The tracked segmentation is adapted in every frame to account for viewpoint changes and non-rigid deformations. Our re-implementation is able to track and re-segment one region at approximately 40 fps.

Starting from an initialization region, the object is first segmented. Foreground and background probabilities P_f and P_b are modeled with color histograms and the contour is described with a level set (LS) embedding function $\mathbf{\Phi}$, which is evolved to optimize the energy functional from [15].

The object's location is modeled as the position $\mathbf{p}$ of the *object frame*, a rectangular region around the contour, described by the parameters of a warp that transforms the image into the object frame. As in [15], we choose the warp to include *translation*, *scale*, and *rotation* to cope with camera motion. In each frame, the object is tracked by performing a rigid registration of the contour, such that the foreground and background model optimally match the image content. We define the tracker confidence as follows:

$$conf(T_j) = \sum_{i \in fg(T_j)} P_f(\mathbf{x}_i) + \sum_{i \in bg(T_j)} P_b(\mathbf{x}_i), \qquad (1)$$

where $P_f(\mathbf{x}_i)$ is proportional to the probability of pixel $\mathbf{x}_i$ belonging to the foreground (frequency of the pixel's color in the foreground color model), P_b analogous for background. For more details please refer to [15].

Features. Ideally, we would like to use the saliency scores from the object candidate generation stage also for judging the quality of tracked regions. However, the absolute saliency scores are incomparable between frames. We therefore compute a larger set of features which are used in both stages. These can be divided into region features and tracklet features, as shown in Tab. I.

Consistency Checks. We maintain candidate quality in two stages. We make the assumption that correct object regions can be tracked, but not each region which can be tracked is an object. In the first stage, we terminate degenerated tracklets that are clearly not tracking a consistent region anymore. This stage makes no decision about whether the tracked region is an object or not; it simply finds failed tracks.

There are several criteria with which failed tracklets can be identified. If the tracking algorithm cannot distinguish foreground from background or has lost the target, the typical behavior is extreme scaling or movement. Thus, we sort out regions which become very small or big, which moved very fast, or have an extremely low tracking score or convexity. Moreover, we perform backward tracking (inspired by [37]), *i.e.*, we track the region one frame into the past to see if returns the same region. If the overlap of the region in the last frame and the backwards tracked region or the backward tracking score is too low we also discontinue the track.

SVM Re-Ranking. In the second stage, we use an SVM classifier to score and re-rank each tracked region and filter out low-scoring tracklets. Since tracks might drift or fail at some point, each frame in a track is scored independently. We train a Gaussian RBF kernel SVM to classify regions into objects and non-objects. Since the data we want to classify can only be computed from tracklets, we create the training feature vectors by running our system without any duplicate removal and with less strict consistency checks on a separate training set. We initialize new tracklets from the saliency object candidates in every 5^{th} frame and track all candidates independently until the tracklets are terminated. This results in a large number of tracklet frame characteristics with both positive and negative examples. We then train the SVM using cross-validation and a grid search for the SVM parameters.

When applying our approach to a new test sequence, we use the SVM to score each tracked region. If the classification is negative more than M times in a row, the tracklet is discontinued. Whenever there is a new candidate that is a duplicate for a tracked region, the tracklet's counter is reset in order to avoid periodic re-initialization of tracklets for the same region.

B. Sequence-Level Candidate Selection

After performing tracking for the entire video sequence, we merge tracklets that show the same object across small tracking gaps using greedy clustering based on the tracklet similarity from [38]. Finally, we rank the resulting tracks by their SVM scores and report the top-ranking results. For visualization, we select a representative view of each tracked region as the frame with best SVM score. Altogether, this

results in a significant reduction in the number of object candidates compared to the frame-level input, since each track can now be represented by a single candidate. Fig. 4 shows some sequence-level candidates that can be obtained by our approach.

VI. Evaluation

We introduce a new benchmark dataset for the evaluation of object discovery methods from video, called *kitchen object discovery dataset*. It consists of five challenging video sequences recorded in real-world indoor environments containing a high degree of clutter. The sequences have on average about 600 frames and contain up to 80 objects. In contrast to many popular benchmarks, our dataset contains real-world images without photographer bias and with a large amount of objects and clutter. In each frame, there are on average 23 objects visible, but some views contain up to 43 objects. Object ground truth in terms of pixel-precise binary maps was annotated manually on every 30th frame, keeping the identity of objects over frames. This makes it possible to evaluate on a sequence level. The videos, annotations and the saliency based candidates are available on our website[3].

In the following, we first evaluate our new saliency-based object candidates, and second the sequence-level candidates obtained by tracking over frames.

Saliency-Based Object Candidates. We first evaluate the quality of our saliency-based candidates by computing the precision (percentage of candidates that corresponds to a ground truth object) and recall (percentage of discovered ground truth objects) values depending on the number of candidates per frame. Our saliency method is able to operate in two modes: the single saliency mode, and the split-octave approach that generates candidates on each level of the saliency pyramid. We compare our methods with four recent approaches that have shown good performance for object candidate detection: The objectness measure of Alexe et al. [4] (OM), the Randomized Prim Object Candidates of Manén et al. [8] (RP), the contour detector with hierarchical image segmentation of Arbelaez et al. [39] (gPb), and our previous method [35] that uses adaptive thresholding instead of region growing to extract candidates and computes a single saliency map (AT). For all methods, we rank the candidates according to their quality ([39] do not provide a score for their hierarchical regions. We extract regions with a watershed algorithm and use the difference between the maximum and the minimum contour score as region score). Since [4] and [8] deliver bounding boxes instead of pixel-precise regions, we use the smallest surrounding rectangle around the regions also for [39] and our own method to make the methods comparable. Candidates which have at least 50% overlap (intersection-over-union) with the ground truth are counted as correct.

Fig. 5 (left) shows the average precision and recall over the number of candidates per frame. From our two methods, the single-scale version achieves consistently the highest precision, while the split octave variant achieves the same precision as AT and better precision than the other three approaches. For the recall, the quality depends on the number of candidates that is considered: for few candidates per frame (up to 80), the single saliency approach (orange) achieves the highest values. For more candidates, the split-octave version (red) that generates more candidates achieves the highest recall. Especially difficult objects that are missed with the other methods are detected with this approach: for 200 candidates/frame, we achieve a recall of 52%, whereas all other methods remain below 40%. Here, we favor the split-octave version with more candidates and a higher recall since our sequences contain up to 43 objects/frame and the tracking of candidates sorts out bad candidates later on. So, we chose the split-octave method for the remaining evaluations. We also evaluate the frame-level candidates generated by the tracker in comparison to the saliency-based object candidates with which the tracker was initialized (Fig. 5 (right)). Our tracker is able to achieve a higher recall than the saliency-based candidates, which shows that the tracker is able to track good candidates into later frames, where they are not among the saliency-based candidates.

In our video scenario, it is not only of interest how many objects are detected on a per-frame level, but even more how many real-world objects are found over the whole sequence. Some objects might not be found in one frame, but in another one. Therefore, we additionally measure the global recall, *i.e.*, the percentage of discovered individual objects from all objects that are visible over the entire sequence. The results are summarized in Fig. 6 (bottom) in terms of the global recall obtained at the end of the sequence. The results show that our saliency-based object candidates (2nd row) is able to find on average 85% of the objects in the scene and thus outperforms all the baseline methods clearly. The tracking and feature evaluation (1st row) further improve the result to achieve 91% on average. Additionally, we plot how the global recall evolves over time (cf. Fig. 6, top).

Sequence Level Results. Finally, we evaluate the recall over the absolute number of candidates. With this, we show how many individual objects were found in the whole sequence, regardless of which frame they were found in. (These plots should not be confused with the global recall over time, where the accumulated recall over the course of the video is shown.) For the saliency-based object candidates, this means we consider all candidates for every annotated frame. For the tracking-based candidates, one candidate consists of one tracklet. A tracklet is considered correct if it has more than 50% intersection over union overlap with a ground truth object in at least one annotated frame. Fig. 7 shows the percentage of individual objects found per number of candidates for the five different sequences. These results show that with the tracking stage we can significantly reduce the number of candidates while still achieving higher precision and recall than with the saliency-based candidates. In all of the sequences, we only need between 500 and 750 track-level candidates per sequence to reach above 80% recall –

[3] http://www.vision.rwth-aachen.de/projects/kod/

Fig. 4: Correct sequence level candidates (tracking initialized with 200 saliency-based object candidates).

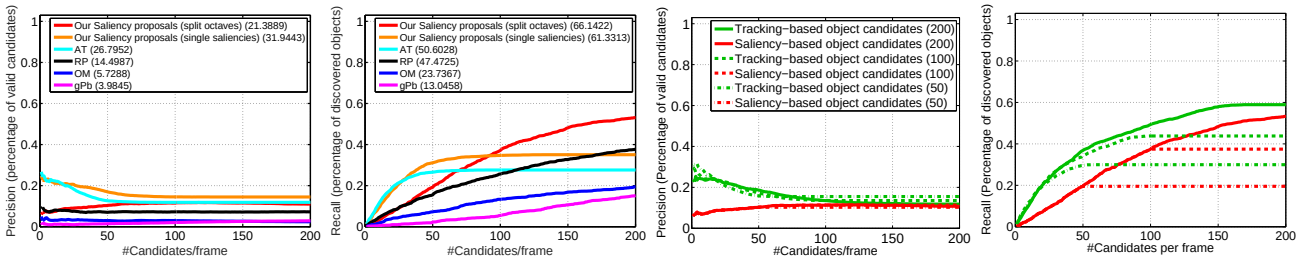
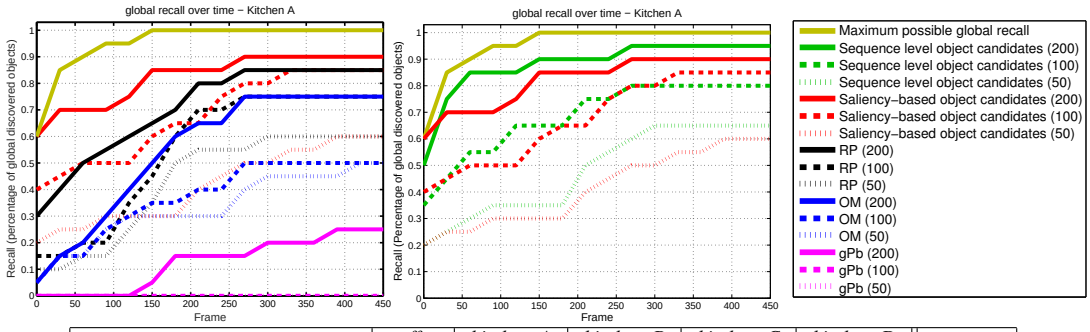


Fig. 5: Precision and recall over number of candidates per frame averaged over all sequences. (Left) Our two new methods (orange: single saliency map, red: split octaves) compared to the methods AT [35], OM [4], RP [8], and gPb [39]. Numbers in the legend denote AUC values. Since here we are mainly interested in high recall, we use the split octave version (red) in the following experiments. (Right) For 50/100/200 object candidates. Green: Frame-level tracking-based object candidates.



	coffee	*kitchen A*	*kitchen B*	*kitchen C*	*kitchen D*	*average*
Tracked frame-level candidates	**0.81**	**0.95**	**0.88**	**0.96**	**0.93**	**0.91**
Saliency-based candidates	*0.78*	*0.90*	**0.88**	**0.96**	*0.76*	*0.86*
RP [8]	0.73	0.85	0.74	**0.96**	0.70	0.80
OM [4]	0.47	0.75	0.67	0.69	0.65	0.65
gPb [39]	0.42	0.25	0.60	0.57	0.43	0.45

Fig. 6: (Top) Global recall over time for *kitchen A*, *i.e.* how many of the individual objects in the scene were found over the course of the video, for 50/100/200 object candidates per frame. (Left) Our saliency-based candidates in comparison to baseline methods. (Middle) Saliency-based candidates in comparison to frame-level tracking-based object candidates. (Bottom) Table showing the global recall for 200 object candidates per frame. Best result in bold and second best in italic.

compared to the 200 saliency-based object candidates per frame for each frame of the sequence that we start from, this is a significant reduction! Fig. 4 shows the correct sequence level candidates in some example frames.

VII. Conclusion

We have presented a new method for object candidate generation on a sequence-level. It is especially well suited for videos from mobile devices in which it is important to limit the cost factor of recognition queries. Our method consists of two steps: first, a new frame-based object candidate detector based on multi-scale saliency determines candidates with a higher per-frame recall than current state-of-the-art methods. We show that this method is able to detect most of the objects in a scene even in very complex scenarios with plenty of objects and a high degree of clutter. Second, the candidates are tracked over time in order to group candidates that belong to the same real-world object and filter out inconsistent regions. Thus, our approach delivers a set of region trajectories that combine different views of an object. These two components result in a significant reduction of candidates compared to frame-based methods, while keeping a consistently high recall. We show that we are able to detect on average 91% of the objects with this method. Finally, we select a representative view for each track that can be used as a query for recognition in future work.

Acknowledgements This research has been funded by the DFG project "Situiertes Sehen zur Erfassung von Form und Affordanzen von Objekten" (FR 2598/5-1 and LE 2708/1-1) under the D-A-CH lead agency programme.

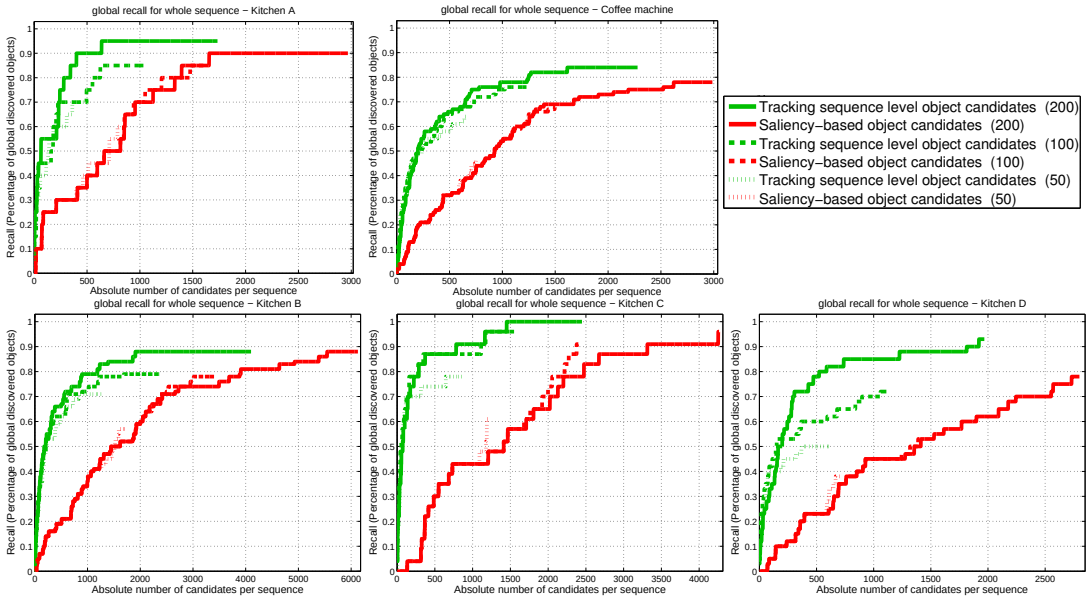


Fig. 7: Sequence level results: we show the global recall (percentage of individual objects found) over the absolute number of object candidates per sequence for 50/100/200 considered object candidates per frame. Green: Sequence level object candidates. Red: Saliency-based object candidates.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *NIPS*, 2012.

[2] P. Felzenszwalb, B. Girshick, D. McAllester, and D. Ramanan, "Object Detection with Discriminatively Trained Part-Based Models," *PAMI*, vol. 32, no. 9, 2010.

[3] T. Dean, M. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik, "Fast, Accurate Detection of 100,000 Object Classes on a Single Machine," in *CVPR*, 2013.

[4] B. Alexe, T. Deselaers, and V. Ferrari, "Measuring the Objectness of Image Windows," *PAMI*, vol. 34, no. 11, pp. 2189–2202, 2012.

[5] M. Bleyer, C. Rhemann, and C. Rother, "Extracting 3D Scene Consistent Object Proposals and Depth from Stereo Images," in *ECCV*, 2012.

[6] J. Carreira and C. Smichisescu, "CPMC: Automatic Object Segmentation Using Constrained Parametric Min-Cuts," *PAMI*, vol. 34, no. 7, 2012.

[7] I. Endres and D. Hoiem, "Category-Independent Object Proposals with Diverse Ranking," *PAMI*, 2014.

[8] S. Manén, M. Guillaumin, and L. Van Gool, "Prime Object Proposals with Randomized Prim's Algorithm," in *ICCV*, 2013.

[9] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders, "Selective Search for Object Recognition," *IJCV*, vol. 104, no. 2, 2013.

[10] S. Fidler, A. Sharma, and R. Urtasun, "Bottom-Up Segmentation for Top-Down Detection," in *CVPR*, 2013.

[11] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H.-Y. Shum, "Learning to Detect a Salient Object," *PAMI*, 2009.

[12] R. Achanta and S. Süsstrunk, "Saliency Detection using Maximum Symmetric Surround," in *ICIP*, 2010.

[13] D. Klein and S. Frintrop, "Salient Pattern Detection using W_2 on Multivariate Normal Distributions," in *DAGM*, 2012.

[14] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang, "Saliency Detection via Graph-based Manifold Ranking," in *CVPR*, 2013.

[15] C. Bibby and I. Reid, "Robust Real-Time Visual Tracking using Pixel-Wise Posteriors," in *ECCV*, 2008.

[16] J. Shin, R. Triebel, and R. Siegwart, "Unsupervised 3D Object Discovery and Categorization for Mobile Robots," in *ISRR*, 2011.

[17] D. Wang, I. Posner, and P. Newman, "What Could Move? Finding Cars, Pedestrians and Bicyclists in 3D Laser Data," in *ICRA*, 2012.

[18] A. Karpathy, S. Miller, and L. Fei-Fei, "Object Discovery in 3D Scenes via Shape Analysis," in *ICRA*, 2013.

[19] M. Bjoerkman and D. Kragic, "Active 3D Segmentation through Fixation of Previously Unseen Objects," in *BMVC*, 2010.

[20] G. Kootstra and D. Kragic, "Fast and Bottom-Up Detection, Segmentation, and Evaluation using Gestalt Principles," in *ICRA*, 2013.

[21] D. Katz, M. Kazemi, J. A. Bagnell, and A. Stentz, "Interactive segmentation, tracking, and kinematic modeling of unknown 3d articulated objects," in *ICRA*, 2013, pp. 5003–5010.

[22] D. Schiebener, A. Ude, and T. Asfour, "Physical Interaction for Segmentation of Unknown Textured and Non-textured Rigid Objects," in *ICRA*, 2014.

[23] M. Johnson-Roberson, J. Bohg, M. Björkman, and D. Kragic, "Attention-based Active 3D Point Cloud Segmentation," in *IROS*, 2010.

[24] M. Bjoerkman, N. Bergstroem, and D. Kragic, "Detecting, Segmenting and Tracking Unknown Objects using MRF Inference," *CVIU*, 2013.

[25] E. Potapova, K. M. Varadarajan, A. Richtsfeld, M. Zillich, and M. Vincze, "Attention-driven object detection and segmentation of cluttered table scenes using 2.5 d symmetry," in *ICRA*, 2014.

[26] D. Hoiem, A. Efros, and M. Hebert, "Recovering Occlusion Boundaries from an Image," *IJCV*, vol. 91, no. 3, pp. 328–346, 2011.

[27] P. Felzenszwalb and D. Huttenlocher, "Efficient Graph-based Image Segmentation," *IJCV*, vol. 59, no. 2, 2004.

[28] L. Itti, C. Koch, and E. Niebur, "A Model of Saliency-Based Visual Attention for Rapid Scene Analysis," *PAMI*, vol. 20, no. 11, 1998.

[29] K. Shi, K. Wang, J. Lu, and L. Lin, "PISA: Pixelwise Image Saliency by Aggregating Complementary Appearance Contrast Measures with Spatial Priors," in *CVPR*, 2013.

[30] N. Bruce and J. Tsotsos, "Saliency, Attention, and Visual Search: An Information Theoretic Approach," *J. of Vision*, vol. 9, no. 3, 2009.

[31] D. Klein and S. Frintrop, "Center-surround Divergence of Feature Statistics for Salient Object Detection," in *ICCV*, 2011.

[32] F. Perazzi, P. Krahenbuhl, Y. Pritch, and A. Hornung, "Saliency Filters: Contrast based Filtering for Salient Region Detection," in *CVPR*, 2012.

[33] L. Zhu, D. Klein, S. Frintrop, Z. Cao, and A. Cremers, "Multi-Scale Region-Based Saliency Detection Using W2 Distance on N-Dimensional Normal Distributions," in *ICIP*, 2013.

[34] C. von Hofsten and E. Spelke, "Object perception and object-directed reaching in infancy." *J. of Exp. Psych.*, vol. 144, no. 2, 1985.

[35] S. Frintrop, G. Martín García, and A. Cremers, "A Cognitive Approach for Object Discovery," in *ICPR*, 2014.

[36] R. Adams and L. Bischof, "Seeded Region Growing," *PAMI*, vol. 16, no. 6, pp. 641–647, 1994.

[37] Z. Kalal, K. Mikolajczyk, and J. Matas, "Forward-Backward Error: Automatic Detection of Tracking Failures," in *ICPR*, 2010.

[38] R. Kaucic, A. Perera, G. Brooksby, J. Kaufhold, and A. Hoogs, "A Unified Framework for Tracking through Occlusions and Across Sensor Gaps," in *CVPR*, 2005.

[39] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour Detection and Hierarchical Image Segmentation," *PAMI*, vol. 33, no. 5, 2011.