Embedding High-Level Information into Low Level Vision: Efficient Object Search in Clutter

Ching L. Teo, Austin Myers, Cornelia Fermüller, Yiannis Aloimonos

***Abstract*— The ability to search visually for objects of interest in cluttered environments is crucial for robots performing tasks in a multitude of environments. In this work, we propose a novel visual search algorithm that integrates high-level information of the target object – specifically its size and shape, with a recently introduced visual operator that rapidly clusters potential edges based on their coherence in belonging to a possible object. The output is a set of fixation points that indicate the potential location of the target object in the image. The proposed approach outperforms purely bottom-up approaches – saliency maps of Itti et al. [15], and kernel descriptors of Bo et al. [2], over two large datasets of objects in clutter collected using an RGB-Depth camera.**

I. Introduction

Fig. 1. An example of a challenging cluttered scene.

Imagine you enter your kitchen as shown in Fig. 1 after a party, and you were asked to look for a *particular* pair of scissors. What would be your search strategy? Would you try to remember where you last saw the scissors? Or would you try to go for the obvious locations of where scissors would be placed – in the drawers, or besides the knives? Once you have prioritized *where* to start searching, you start to remember how your particular pair of scissors *looks* – its shape, size and maybe some unique identifying color or labels so you will recognize it from other pairs of scissors that have other uses.

This is an example of a typical search scenario that humans encounter everyday. The strategy is straightforward, consisting of two main parts – 1) going to the location of the object and 2) searching for the object near that location using its known appearance. Yet, it remains a formidable challenge for robots. There are several reasons:

The authors are from the Department of Computer Science, University of Maryland, College Park, MD 20742, USA {cteo,amyers,fer,yiannis}@umiacs.umd.edu

1) Navigating in cluttered environments. The robot must be capable of moving safely in cluttered environments without posing a danger to itself or to its surroundings. This requirement demands navigation strategies that include obstacle avoidance and advanced path planning in clutter [14], and developing good motion control strategies in confined environments [4].

2) Perceptual challenges. Another crucial aspect of a successful search strategy is to develop algorithms that can locate the target object in clutter once the robot is at a potential location. Regardless of the sensor used, the challenges are similar. They involve: a) rapidly determining the locations of the objects – via a series of *fixation points* to reduce the search space and b) performing recognition at these locations to identify the target object. This work focuses on determining the fixation points in this part of the strategy.

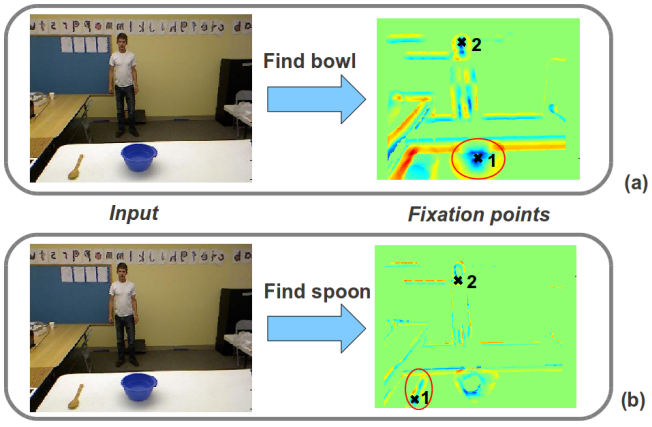

Fig. 2. Output of the algorithm: (Left) Input scene. (Right) Top two fixation points (black crosses with corresponding ranks) for the two target objects: (a) bowl and (b) spoon, computed using known properties of the target objects.

An important aspect of the human search strategy is that the *memory* of the target object is constantly invoked – i.e. knowledge about potential location and appearance is utilized. The apparent ease with which humans exploit this high-level information belies the computational complexities. In this paper, we introduce a novel approach that extends the use of a recently introduced image operator called the *image torque* [22]. This image operator, in its original form was designed as a generic mid-level operator that groups edge responses into potentially closed contours. Here we modify the operator using knowledge of the target object's properties to respond to specific edges and produce potential fixation

points that indicate where the object could be. An example of the output of the algorithm is shown in Fig. 2. In the sections that follow, we first review related work, and then describe the algorithm in detail. We then present evaluations on two datasets containing objects in varying degrees of clutter and discuss the effectiveness of our approach in real life scenarios.

II. Related Work

The problem of searching for objects in clutter has been studied by several prior works. Reviews of state of the art techniques in the field of robotics and computer vision can be found in [10] and its references herein. Our work is closely related to the problem of computing *salient* regions in images by modeling the attentional mechanism of the human visual system. Attention models can be separated into two main categories based on whether their deployment over a visual scene is guided by scene features or by intention: the first is called *bottom-up* and is driven by low-level processes; the second refers to *top-down* process [12]. For bottom-up attention, several models have been proposed [24], [19] including the *saliency map* of Itti et al. [15] which has become a standard baseline of bottom-up visual attention: saliences are computed independently from primitive features such as intensity, gradient orientations and color and combined later.

Since we are interested in detecting objects based on their known properties, our work is firmly placed in the realm of top-down approaches. Top-down attention is more complex because it represents objects in memory [13] and uses the memory to detect likely objects in an attended visual scene [23] – which is the approach that is most related to this work. There are also a few top-down visual attention models [21], [25], including the VOCUS system [9]. In this system, top-down visual attention is based on the maximum salient region of the target object image, where a model is developed of the target object image by computing weights of the appropriate features. During runtime, the weight model of the target object is used to construct a weighted sum of conspicuity maps, representing the level of saliency for a single visual feature. Others combine bottom-up and top-down attention [20]. The top-down component uses accumulated statistical knowledge of the visual features of the desired search target and background clutter, to optimally tune the bottom-up maps such that target detection speed is maximized. The performance of these top-down approaches is very much influenced by the scene where the object is and they fail when the scene changes in significant ways. By way of contrast, our approach differs in the sense that we integrate high-level knowledge of the object model – specifically shape and size information, *directly* via the torque operator with low-level edge features in the image, without the need to decide on the weights of each specific feature, or maps for combination, or a priori knowledge of the background.

III. Approach

In this section, we describe the approach for integrating high-level knowledge into the visual search problem as described in sec. I. We first introduce the *torque* operator and motivate its use for the object search problem. Next we describe how torque is computed in general and its extension when specific shape and size information about the target object is known. We then describe further details of the algorithm that supports the computation of the torque and conclude with a high-level system overview of the entire approach, together with the optimizations needed to make it a feasible top-down attentional mechanism for robots. We conclude with a discussion of how this approach compares with state of the art object recognition methods that are similar in spirit.

A. Image torque for fast object search

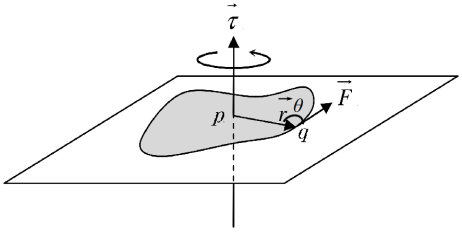

Fig. 3. From [22]. Image torque for discrete edges. $\vec{r}$ is the vector from the center pixel p to an edge pixel q. $\vec{F}$ is the tangent vector and θ is the angle between $\vec{r}$ and $\vec{F}$.

The recently introduced *image torque* [22] is a mid-level image operator tuned to closed contours in images. The underlying motivation is to find object-like regions by computing the "coherence" of the edges that support the object. Edge coherence is measured via computing a cross-product between the edge pixel tangent to a center point as shown in Fig. 3. Formally, the value of torque, τ_{pq} of an edge pixel q within a discrete image patch with center p is defined as:

$$\tau_{pq} = \|\vec{r}_{pq}\| \sin\theta_{pq} \tag{1}$$

where $\vec{r}_{pq}$ is the displacement vector from p to q and θ_{pq} is the angle between $\vec{r}_{pq}$ and the tangent vector at q[1]. The torque of an image *patch*, P, is defined as the sum of the torque values of all edge pixels, $E(P)$, within the patch as follows:

$$\tau_P = \frac{1}{2|P|} \sum_{q \in E(P)} \tau_{pq} \tag{2}$$

The torque has been used as a mid-level operator as follows: At every image point over multiple patch sizes the torque is computed. Then at every image point the largest torque value over the different patch sizes is selected to create a two-dimensional data-structure, called the torque *value map*. The extrema in this torque value map indicate locations in the image that likely are centers of closed contours.

Several interesting properties for the torque operator were explored in [22]. Some of the most relevant to our work are

[1]The sign of τ_{pq} depends on the direction of the tangent vector and for this work, we compute the direction based on the change in pixel intensities along the edge pixel.

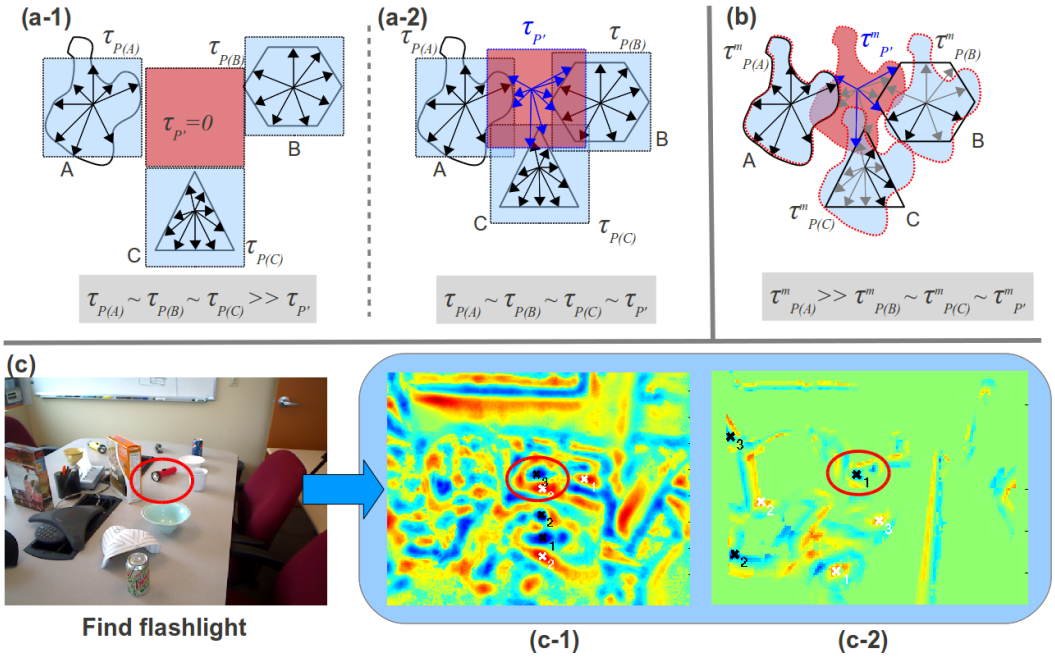

Fig. 4. How the torque operator performs in clutter: (a) Original torque for a non-cluttered (a-1) versus a cluttered (a-2) situation with three objects {A, B, C}. Arrows represent the edge support computed for each torque patch value: $\tau_{P(\cdot)}$. In the non-cluttered situation (a-1), torque values are high compared to the empty region $\tau_{P'}$. In a cluttered situation, edges from neighboring objects are accidentally added into the support for patch P' (a-2), resulting in similar torque values with true object patches. (b) The modified torque with high-level information, $\tau^m_{P(\cdot)}$ – shown here for the shape for patch 'A', enhances edges (dark arrows) that conform to part of A's shape while reducing the contribution of non-conforming edges (gray arrows). This results in torque patches that are tuned specifically for the target object's shape and size. (c) Results of τ_P (c-1) versus τ^m_P (c-2) in a real cluttered scene where the `flashlight` is to be located. Notice that in (c-1), there are numerous torque maxima/minima (white/black crosses with ranks) while in (c-2) the `flashlight` is the top fixation with less noisy torque values. Note that only the top 3 torque maxima/minima are shown for clarity.

the observation that the torque tends to respond strongest to closed regions, has large values at the center of regions, and ignores texture. Thus it is well suited as a tool for finding edges that belong to an object. This is because edges that are incoherent – e.g. texture edges have tangent vectors that are random, and summing them up via eq. (2) will result in a small τ_P (close to zero). In addition, because of the simple summation operation in eq. (2), τ_P of varying sizes can be computed rapidly using the method of integral images [5].

These properties make the torque an efficient operator for detecting object-like locations when the objects are themselves *not* within significant clutter. The reason is illustrated in Fig. 4(a), where we show a non-cluttered versus a cluttered situation of three simple objects. However, it is a purely bottom-up approach. In eq. (2) knowledge about which edges really belong to the object is not considered, and thus accidental inclusions from nearby edges of other objects will produce large torque values for patches that are *between* objects due to clutter, see Fig. 4(a-2). We show in the next section how this effect can be reduced so that torque can be used effectively in cluttered situations.

B. Extensions for known object properties

For the torque to handle cluttered situations effectively, the key is to modify the original formulation of the torque for an image patch eq. (2). The torque values of edge pixels, τ_{pq} are modified via an *object model* function, $m_{\mathcal{O}}(\cdot)$ such that edges that *conform* to the target object model, $\mathcal{O}$ are given higher weight while non-conforming edges are given less weight:

$$\tau^m_P = \frac{1}{2|P|} \sum_{q \in E(P)} m_{\mathcal{O}}(\tau_{pq}) \tag{3}$$

where τ^m_P is the modified normalized torque of an image patch. There are numerous ways one can design $m_{\mathcal{O}}(\cdot)$ – it can be simple: based on local image properties of the target object, or complex: the output of trained class specific edge classifiers such as [17]. In this work, we focus on using two specific *global* object properties that define $\mathcal{O}$: 1) *shape* – this is represented as a set $S_{\mathcal{O}}$ of known object masks (or poses) and 2) *size* – the approximate metric size, $[X, Y, Z]_{\mathcal{O}}$ of the object is known a priori. Using these sources of information we formulate $m_{\mathcal{O}}$ as:

$$m_{\mathcal{O}}(\tau_{pq}) = \frac{\tau_{pq}}{d_{qs}} \tag{4}$$

where d_{qs} is the minimum Euclidean distance of edge point q to the edges $s \in S'_{\mathcal{O}}$ on a given object, for a selected object pose. It should be noted that $S'_{\mathcal{O}}$ will be resized using data $[X, Y, Z]'_{\mathcal{O}}$, i.e. we use 3D data from an RGB-Depth camera so that only edges that conform closely to the desired target object shape are included in eq. (3).

We illustrate how τ^m_P helps reduce erroneous torque values from occurring within clutter in Fig. 4(b) using $m_{\mathcal{O}}(\cdot)$ as defined above. Referring to the figure, one can see that edges that belong to another object are likely to have a smaller

influence in the τ_P^m, while edges that approximate well the shape of the object model $\mathcal{O}$ are promoted. This enables target objects that are within a large amount of clutter to be enhanced, as shown in Fig. 4(c) on a real cluttered scene. The benefits of imposing $\mathcal{O}$ within τ_P^m is clearly shown here: the torque maps are less noisy (with less erroneous maxima/minima) and the targeted object is likely to have the highest torque values. A similar benefit extends to objects that are partially *occluded* in clutter or slightly deformed. The same principle applies. Occlusions and deformations only slightly perturb τ_P^m, and therefore the operator is robust to such effects. These results highlight a biologically plausible explanation of the modified torque operator: it is analogous to *receptive fields* in the visual cortex that are sensitive to particular sizes and shapes. Finally, it is important to note that since eq. (3) is similar in structure to the original torque formulation, integral images can be used to speed up the computations considerably.

Our approach to finding an object of a certain class consists of a series of processing steps. First, in a precomputation step, we derive using RGB-Depth data, the expected size of the object in the image, and we compute for every image patch the most likely orientation (or pose) of the object (sec. III-C). Then we use the torque operator to locate the regions in the image possibly containing the object. We modify the contribution of edges by giving larger weights to those edges that are nearer to the contour of the target object's model, and use the torque operator to group edges into closed contours. Section III-D describes the complete method.

C. Preprocessing: Pose and size estimation

An important requirement for the torque operator to function efficiently in clutter is to know which object poses in $S_{\mathcal{O}}$ is the most appropriate for use in eq. (3). Since the target object can appear in any possible pose and scale, it would seem that one would have to try *all* $|S_{\mathcal{O}}|$ poses and scales at each image patch – selecting the one that generates the largest absolute torque value in the end. This approach would have increased the computational time of each image patch considerably, and is therefore not feasible as a practical mechanism for top-down attention. A more efficient solution is to estimate at runtime the best pose within each image patch. The strategy is shown in Fig. 5. We do this by computing *shape context* features [1] from sampled edge points (we take 10% of the edge points in this work) within each patch, and compare it to precomputed shape context features of each model pose $S'_{\mathcal{O}}$ to determine the right pose to use at each edge point. As a final step, a large window (we used a 50×50 window) was used to compute the mode of the pose estimate to produce a *pose map* that estimates the best pose at each pixel location. The intuition for this step is that if the edges do indeed come from a known object with a particular pose, most of the edge pixels would have voted for the *same* pose. Using a small window to compute the mode removes noisy single pixel deviations from the majority votes.

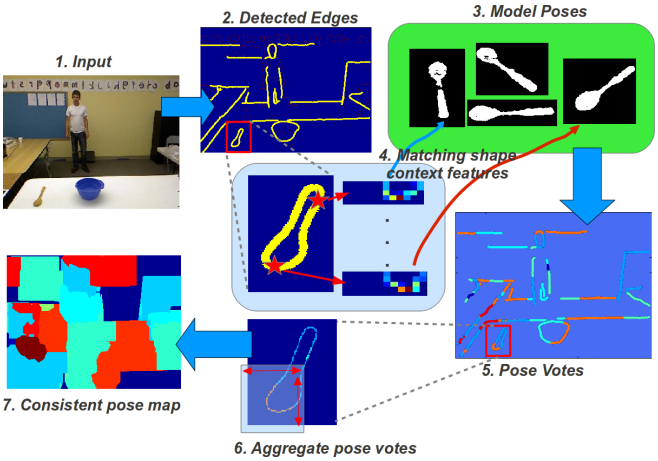

Fig. 5. Estimating consistent poses from shape context edge features. (1) Input image. (2) Detected edge points (in yellow). (3) A set of model poses (masks) for the target object, (4) Shape context features are extracted at each edge points and matched to shape context features in the model poses. (5) Each edge pixel then votes for the closest matching pose – different colors represent votes for a particular pose. (6) A window is run over the entire image to compute the votes. (7) The final consistent pose map.

Since the target object can occur at any location in the image, the apparent size of the object in the image will vary based on the distance of the object to the sensor. The right patch size to compute τ_{pq}^m is therefore dependent on the scene structure at runtime, which can be easily computed from depth information (either from stereo or directly from RGBD cameras). We use this information to compute an appropriate *scale map* (see Fig. 6) that indicates at x, y the correct image scale that the object must have at that particular location. By precomputing the pose and scale maps, we can rapidly compute the final torque values of the image at one pass, which greatly reduces the running time. See the next section for an analysis.

D. A knowledge driven top-down object detection mechanism

We are now ready to present the full algorithm summarized in Fig. 6. Object model $\mathcal{O}$ is computed from segmented images from RGBD data to obtain their respective pose masks and metric size information. The input is an image frame together with its computed depth map obtained from an RGBD camera. For determining the edge features, one can use either standard Canny edges or Pb edges [18] (we use Pb edges in all experiments here). We then apply a threshold on the length of these initially detected edges so that only long continuous edge segments are preserved. In this work we set the minimum length to 100 pixels; we found that this length helps in promoting object boundaries. The next step is to determine the pose and scale map for each pixel as described in sec. III-C. Finally, we apply eq. 3 to obtain a torque value map for each pixel. We apply to this torque map non-maxima suppression to get local maxima/minima, and use these extrema as the fixation points in the image, where we expect the target object.

A note on the computational complexity of the entire

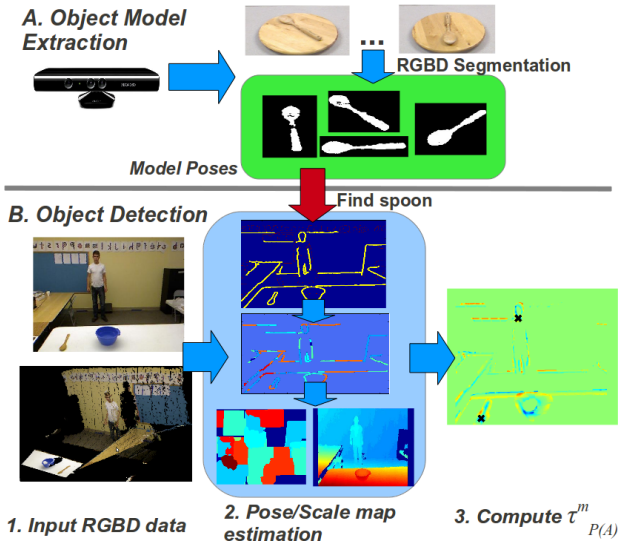

Fig. 6. Overview of the proposed top-down object detection algorithm. (A) Extracting object model properties from RGBD data: poses and size. (B) Runtime detection of target object: (1) Input RGBD data. (2) Compute pose and scale maps using object model information. (3) Compute τ^m_P and find extrema (the top 2 absolute extrema values are shown as black crosses).

algorithm. For an image of size $N \times M$, with $J = |S_{\mathcal{O}}|$ number of poses, G the number of edges with $G \ll (N*M)$, and assuming that the maximum number of sizes of image patches is limited to K (a reasonable assumption since most RGBD cameras can provide depth information up to $\approx$ 6m), the computation time for estimating the pose is $O(G*J)+O(N*M)$ (search + finding consistent pose votes). Computing the scale map takes at most $O(N*M)$ time, since we have 2.5D information directly from the sensor. Finally, for computing τ^m_P, since it takes constant time to compute a patch at one scale and pose, it takes at most $O(N*M)$ time for computing a torque value for each object. The total run time of the approach per object is therefore at most $O(G*J)+O(N*M)+O(N*M)+O(N*M) \approx O(N*M)$. Hence the computational time scales linearly with the image size and the number of object classes to be searched – $O(N*M*C)$ where C is the number of object classes in the worst case. Typical values of $\{N, M, J, K\}$ are $\{640, 480, 20, 10\}$ with C ranging from 6 to 8. Typical run times on an unoptimized Matlab implementation are around 30s per image per object (excluding the time to compute Pb edges). These run times can of course be reduced in a parallel implementation as most computation loops are independent.

E. Comparison to similar methods

Since we propose a "top-down" object detection approach, we need to mention other methods in computer vision that address the object detection problem in similar ways. These methods are usually referred to as "object classifiers," and their goal is to perform object recognition – to find *both* location and identity of a target object in an image. Many of these approaches use a *sliding window* and attempt to match the features from the window to the target object's features. The best known of these efforts were developed in the course of the PASCAL-VOC (Visual Object Challenge) [7] competition, which has 20 object classes in a variety of challenging image scenarios. Among the top performing algorithms is the deformable parts based model of Felzenswab et al. [8]. There are, however, limitations to such approaches: 1) the performance of these learning based methods is directly related to the number of training samples. For example, the deformable model algorithm performs well for certain classes – for example the class `person` while it fails considerably for the class `boat`. The main reason is that the set has many more annotated examples of humans than boats. 2) Although the deformable parts model tries to ensure a *global* spatial coherence of object parts, its internal representation for each part – typically, a histogram of orientated gradients (HoG) [6] or SIFT [16] – ignore the spatial structure of the image at the lowest feature levels (gradients, edges etc.).

Recently, Bo et al. [2] presented kernel descriptor features, which they showed to outperform competing feature extraction methods. They showed that orientation histogram features are equivalent to a certain type of match kernel over image patches. This view provides a framework to transform local pixel attributes into patch level features, and it avoids quantization errors associated with histogram binning. Using this approach Bo et al. present kernel descriptors based on gradient, color, and shape information. Once kernel descriptors are computed, pyramid efficient match kernels (EMK) aggregate these local descriptors into object-level features.

By contrast our proposed top-down object detection algorithm does not require a large amount of training data – in fact, only the general pose and metric size information is required. Such information can be obtained either from known measurements or even from generic drawings, which makes the approach general and not as data-dependent as other approaches. Our method also does not require a specific knowledge of the background, making it more adaptable to novel scenes compared to training-based methods which often require numerous background examples. This requirement limits their applicability in real robotic situations. Second, by using the torque operator and the formulation in eq. (3), unlike approaches that ignore completely the spatial configuration of image features, our approach encodes spatial information in a robust manner via the torque operator, resulting in a detector that is robust to clutter, partial occlusions and slight deformations of the target object.

IV. Experiments

A. Datasets

In order to evaluate the proposed algorithm in detecting objects in real clutter, we evaluated the system over two datasets captured using an RGBD camera. The first dataset, called `UMD-clutter` consists of three sequences taken with an RGBD camera mounted on a mobile robot that is moving in front of a cluttered table, and viewing the clutter from various angles and distances. There are seven objects in this dataset: {`Plastic spoon`, `Blue mug`, `Book`, `Bowl`, `Tissue box`, `Wooden spoon`, `Yellow mug`}. The three sequences: `clutter-01`, `clutter-02`,

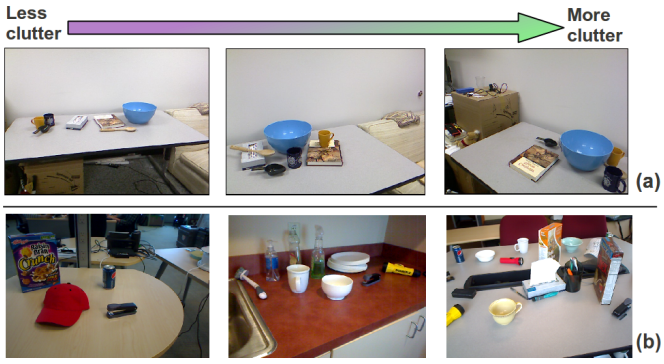

Fig. 7. Example frames from the evaluation datasets, shown from left to right with increasing clutter. (a) Sequences from `UMD-clutter`. (b) Sequences from `rgbd-scenes`.

`clutter-03` (around 500 frames @15fps) are organized in terms of increasing clutter – `clutter-01` has objects that are clearly visible, while `clutter-03` is the most challenging with numerous occlusions. Examples are shown in Fig. 7(a). The second dataset comes from the publicly available `rgbd-scenes` dataset[2]. This dataset consists of eight sequences of around 200 frames taken with an RGBD camera from a variety of environments with varying degrees of clutter as well (Fig. 7(b)). It consists of six different objects classes: {`Bowl, Cap, Cereal box, Coffee mug, Flashlight, Soda can`} Different sequences have different numbers of objects and to make it even more challenging, every object class consists of different object instances – e.g. object `Cap` can be a red cap or white cap. For the purpose of the evaluation and comparison, we only collected object models from one particular instance. In both datasets, the object models are derived from a sequence of RGBD data of the target object class (Fig. 6(A)) placed on a turnstile so that a large number of poses could be collected.

B. Procedure and evaluation protocol

For each sequence in both datasets, we selected a subset of the frames (every 10th frame) since the scene does not change dramatically between frames and our goal was to evaluate the performance of the algorithm from various viewpoints and distances. Since we are interested in evaluating the *quality* of the fixations, a suitable performance metric would be the Cumulative Match Characteristic Curve (CMC) [3], which plots the probability that a correct fixation occurs against the returned list of candidate fixations $[1 : R]$. The CMC is a well used metric in biometric identification systems and is often used for evaluating identification algorithms that return a ranked list. The closer the curve peaks near the top left corner – a high probability of correct identification with small $|R|$ – the better is the quality of the returned fixations.

As comparison, we evaluated the bottom-up saliency algorithm of Itti et al. [15] and the more recent graph-based visual saliency measure (GBVS) of Harel et al. [11]. The extrema in these these saliency maps were used as fixation points. As a state of the art visual object classifier method, we chose the publicly available kernel-based descriptor of Bo et al. [2] which we discussed in sec. III-E. For training and testing, we computed kernel descriptors from 16×16 image patches over a dense regular grid with a stride of 8 pixels. These descriptors were then transformed using EMK, where we considered 1×1 and 2×2 pyramid subregions, and 1000 basis vectors. A multi-class linear SVM was then trained on ground-truth patches belonging to each object class and a sliding window was used at test time to classify each image patch, producing a response map from which fixation points were extracted from the extrema. In addition, we report the results of the original torque implementation [22] where a search over fixed patch sizes: 3×3 to 100×100 was used. All comparisons were done using the default parameters noted in the original papers over all sequences from both datasets. We then compare the locations of the returned fixation points with hand-annotated ground truth labels of the object locations in the test images to compute the associated CMC metric.

C. Results

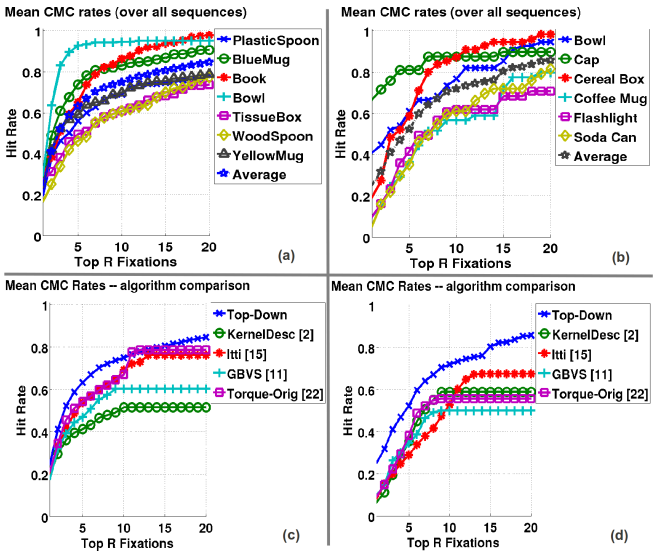

Fig. 8. [Top row] (a) & (b): CMC curves for all objects of the proposed algorithm '`Top-Down`' averaged over all sequences. [Bottom row] (c) & (d): Comparing averaged CMC curves over all objects from both datasets from all algorithms. Left column: `UMD-clutter`, Right column: `rgbd-scenes`.

We summarize the performance of the proposed `Top-Down` object detection mechanism by averaging the CMC over all sequences considered for both datasets and report them in Fig. 8(a) and (b). Comparison of the overall averaged performance of the algorithms – {`Itti, GBVS, KernelDesc, Torque-Original`} are presented in Fig. 8(c) and (d).

From the results, we can see that the proposed top-down approach has better performance compared to all the other algorithms, in terms of returning correct fixations. This is seen from Fig. 8(c) and (d), where the proposed approach

[2]Available from `http://www.cs.washington.edu/rgbd-dataset/index.html`

returns consistently the best performance over all top R returned fixations, even at small R. Compared to the other approaches, the top-down approach also does not saturate in its performance when R increases. This is due to the fact that other approaches do have underlying assumption on the target object and tend to bias their detections towards their underlying assumptions – e.g. color contrast, edge contrast, edge coherence etc. Next, we can see from Fig. 8(a) and (d) that the proposed approach is able to consistently detect *all* objects reliably at increasing R compared to other methods which have strong biases towards a particular object class or certain scene properties. This is in spite of the fact that some objects – e.g. `Plastic spoon`, `Soda can`, `Flashlight`, `Blue mug` are much smaller compared to other objects and often are partially occluded in some of the sequences. This highlights the strength of tuning the torque operator using τ_P^m towards detecting difficult object classes. We also should stress, that only simple edge features and primitive object knowledge was used, compared to the state of the art kernel descriptors `KernelDesc` that utilizes more discriminative features – i.e, color, texture etc. For both `UMD-clutter` and `rgbd-scenes`, the proposed top-down algorithm reports the best performance compared to other approaches with $> 70\%$ hit rate at $R = 10$ for both datasets.

V. Summary and Future Work

In this work, we have proposed a viable and robust top down visual object detection algorithm. Key to the algorithm is the use of a novel image operator called the *torque*. Using the torque as a computational mechanism, we adapted it as a tool for utilizing semantic information in low-level vision tasks. Specifically, we modified the torque computation with high-level information so that it becomes suitable for detecting specific object classes in cluttered environments. We also analyzed the performance of the proposed approach on two large datasets containing clutter with several different objects, and compared our approach to bottom-up saliency approaches and state of the art trained classifiers. We showed that despite the apparent simplicity of the features used – we only used edges, the novel operator clearly outperforms more complex methods which require significant training data or have strong underlying assumptions.

In future work, we intend to investigate how to design edge related functions $m_{\mathcal{O}}(\cdot)$ that carry important universal shape information. We also plan to integrate our approach with a segmentation framework to develop a full object recognition module. Comparisons with other top-down approaches over larger datasets will be conducted. The `UMD-clutter` dataset used with updated results and code will be made available online[3].

VI. Acknowledgements

The support of the European Union under the Cognitive Systems program (project POETICON++) and the National Science Foundation under the Cyberphysical Systems Program is gratefully acknowledged. Ching Teo is supported in part by the Qualcomm Innovation Fellowship. We thank Aleksandrs Ecins for his help in creating the support video for this work.

[3] http://www.umiacs.umd.edu/∼cteo/index_umdclutter

References

[1] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(4):509–522, Apr. 2002.

[2] L. Bo, X. Ren, and D. Fox. Kernel descriptors for visual recognition. In *NIPS*, pages 244–252, 2010.

[3] R. M. Bolle, J. H. Connell, S. Pankanti, N. K. Ratha, and A. W. Senior. The relation between the roc curve and the cmc. In *Proceedings of the Fourth IEEE Workshop on Automatic Identification Advanced Technologies*, AUTOID '05, pages 15–20, 2005.

[4] C. Chen, H.-X. Li, and D. Dong. Hybrid control for robot navigation - a hierarchical q-learning algorithm. *Robotics Automation Magazine, IEEE*, 15(2):37 –47, june 2008.

[5] F. C. Crow. Summed-area tables for texture mapping. *SIGGRAPH Comput. Graph.*, 18(3):207–212, Jan. 1984.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *In CVPR*, pages 886–893, 2005.

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

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

[9] S. Frintrop. *VOCUS: A Visual Attention System for Object Detection and Goal-Directed Search*, volume 3899 of *Lecture Notes in Computer Science*. Springer, 2006.

[10] K. Grauman and B. Leibe. Visual object recognition. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 5(2):1–181, April 2011.

[11] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. In *NIPS'06*, pages 545–552, 2006.

[12] J. M. Henderson. Human gaze control during real-world scene perception. *Trends in Cognitive Sciences*, 7:498–504, 2003.

[13] A. Hollingworth, C. Williams, and J. Henderson. To see and remember: Visually specific information is retained in memory from previously attended objects in natural scenes. *Psychonomic Bulletin and Review*, 8:761–768, 2001.

[14] A. Hornung, M. Phillips, E. Gil Jones, M. Bennewitz, M. Likhachev, and S. Chitta. Navigation in three-dimensional cluttered environments for mobile manipulation. In *ICRA*, pages 423 –429, May 2012.

[15] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(11):1254–1259, Nov. 1998.

[16] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, volume 2, pages 1150–1157, 1999.

[17] J. Mairal, M. Leordeanu, F. Bach, M. Hebert, and J. Ponce. Discriminative sparse image models for class-specific edge detection and image interpretation. In *ECCV*, pages 43–56, 2008.

[18] D. R. Martin, C. C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):530–549, May 2004.

[19] O. L. Meur, P. L. Callet, D. Barba, S. Member, and D. Thoreau. A coherent computational approach to model the bottomup visual attention. *IEEE Trans. on Pattern Anal. Mach. Intell.*, pages 802–817, 2006.

[20] V. Navalpakkam. An integrated model of top-down and bottom-up attention for optimal object detection. In *CVPR*, pages 2049–2056, 2006.

[21] V. Navalpakkam and L. Itti. A goal oriented attention guidance model. In *Proc. 2nd Workshop on Biologically Motivated Computer Vision (BMCV'02)*, pages 453–461. Springer, 2002.

[22] M. Nishigaki, C. Fermüller, and D. DeMenthon. The image torque operator: A new tool for mid-level vision. In *CVPR*, pages 502–509. IEEE, 2012.

[23] R. Rensink. Change detection. *Annual Review of Psychology*, 53:245–277, 2002.

[24] J. Tsotsos, S. Culhane, Y. K. W. Winky, Y. Lai, N. Davis, and F. Nuflo. Modeling visual attention via selective tuning. *Artificial Intelligence*, 78:507–545, 1995.

[25] Y. Yu, G. K. I. Mann, and R. G. Gosine. Modeling of top-down object-based attention using probabilistic neural network. In *CCECE'09*, pages 533–536, 2009.