Damen, D., Gee, A., Mayol-Cuevas, W. W., & Calway, A. D. (2011). Detecting and Localising Multiple 3D Objects: A Fast and Scalable Approach. In IEEE IROS Workshop on Active Semantic Perception and Object Search in the Real World (ASP-AVS-11). Instiute of Electrical and Electronics Engineers (IEEE).

Peer reviewed version

Detecting and Localising Multiple 3D Objects: A Fast and Scalable Approach

Dima Damen[1], Andrew Gee[1], Andrew Calway[1,2], Walterio Mayol-Cuevas[1,2]
[1] Department of Computer Science, University of Bristol, BS8 1UB, Bristol, UK
[2] Bristol Robotics Laboratory, BS16 1QD, Bristol, UK
{damen, gee, andrew, wmayol}@cs.bris.ac.uk

***Abstract*— Object detection in complex and cluttered environments is central to a number of robotic and cognitive computing tasks. This work presents a generic, scalable and fast framework for concurrently searching multiple rigid texture-minimal objects using 2D image edgelet constellations. The method is also extended to exploit depth information for better clutter removal. Scalability is achieved by using indexing of a database of edgelet configurations shared among objects, and speed efficiency is obtained through the use of fixed paths which make the search tractable. The technique can handle levels of clutter of up to 70% of the edge pixels when operating within a few tens of milliseconds, and can give good detection rates. By aligning our detection within 3D point clouds, segmentation and object pose estimation within a cluttered scene is possible.**

Results of experiments on the challenging case of multiple texture-minimal objects demonstrate good performance and scalability in the presence of partial occlusions and viewpoint changes.

I. INTRODUCTION

Detecting real objects under clutter has been one key problem underpinning a series of tasks in cognitive computing and robotics. One of the key complexities arises after clutter and object viewpoints are taken into account, and dealing with these has resulted in a series of techniques using both 2D and 3D information. A more serious competence is being able to learn extra objects as the system explores more about the world, calling for methods that allow for scalability without a severe penalty on processing time.

Consider the case shown in Figure 1 where multiple texture-minimal but known objects are to be detected, and their pose estimated within scenes of a complex cluttered nature. In some cases, as it is here, it is conceivable that the areas over which the process has to operate can benefit from some level of prior knowledge, and therefore background subtraction could be used. However, this still results in non perfectly segmented regions, multiple object occlusions and novel views to be dealt with.

This paper proposes a method that uses constellations of 2D edgelets as the representation. The method is generic, scalable and fast for concurrently searching for multiple rigid texture-minimal objects. Scalability is achieved by using indexing of a database of edgelet configurations shared among objects, and speed efficiency is obtained through the use of fixed paths which make the search tractable. By aligning our 2D edgelet configurations with a corresponding trained point cloud, object pose estimation within a cluttered scene is also possible.

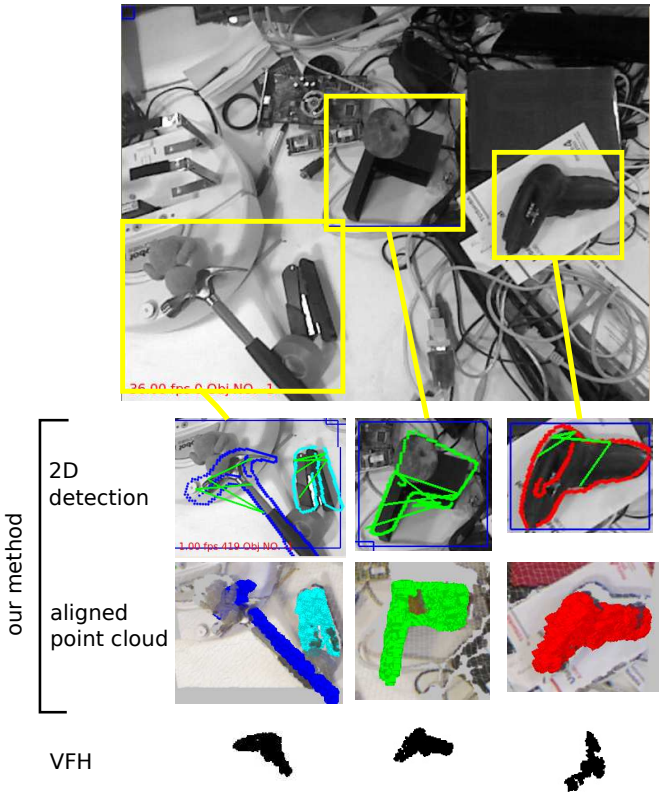

Fig. 1. Using depth background subtraction, three search regions are considered. Our 2D detector along with an aligned point cloud are compared to the best-matched training cluster using VFH object descriptor. When the cluster contains occluded or touching objects, our 2D detector is still able to detect one or more objects.

We have carried out a number of experiments with our approach for several key aspects that include detection performance, framerate expectation, scalability as the object library increases and clutter handling. We then contrast our approach to a state-of-the-art 3D object detector (VFH) [14] where we exemplify how our method can deal with larger levels of occlusion and clutter handling.

II. DETECTING TEXTURE-MINIMAL OBJECTS

There have been many previous approaches to shape matching for textureless/texture-minimal rigid objects and so we focus here only on some directly relevant works. In [9] chamfer distance matching is improved and made faster by

using 3D distance transforms and directional integral images. Detection time of 710ms with 300 reference views are reported. This work uses a search strategy where time increases linearly with more views or with more objects. Such methods are being used in robotics platforms (e.g. [10]).

In [7], image patches represented by histograms of dominant orientations are matched by efficient bitwise operations. This enables detection within 80ms of one object using a rich, 1600 reference views. However, the representation is not rotation or scale invariant (which can be indeed addressed by more reference views). Importantly, the complexity increases with multiple objects, with detection time increasing to 333ms for 3 objects. Wiedemann *et al.* [17] organise object views into hierarchies. Exhaustive search over discrete scales and rotations gives detection times within 300-900ms for a single object. Separate hierarchies are required for each object, giving linear increase in complexity for the case of multiple objects.

There are many techniques which make use of relationships between edge features either within local neighbourhoods to create features for classification [2], [11] or amongst constellations of edgelets located around the contour [3], [4], [5], [8], [12]. For example, Carmichael *et al.* [2] train weak classifiers using neighbourhood features at various radii for detecting wiry objects like chairs and ladders. In [4], consistent constellations of edge features over categories of objects are learnt from training views and used for recognition via exhaustive search over test images. Although these methods demonstrate impressive detection performance, they are not designed for the fast operation we aim to achieve, and are geared towards single object search, with complexity scaling linearly when multiple objects need to be detected.

Scalability to multiple objects was addressed in earlier works by the use of indexing and geometric hashing, similar in form to the library look up that we use in our method. Examples include early work by Lamdan and Wolfson [18] and Grimson [6], and later by others [13], [1]. Of particular note is the work of Beis and Lowe [1]. They use descriptors to represent the relative position and orientation of groups of adjacent line segments which are then searched using a kd-tree for recognising 3-D objects. Our method can be seen as building on these techniques, incorporating the discrimination benefits of constellations of edgelets and the novel use of fixed path configurations to give tractable library look up. Details of the method are given in the next section.

III. DETECTION USING EDGELET CONSTELLATIONS

Our detection of a 3D object is based on the object's shape when seen from different vantage points around the viewing sphere. We refer to the edge map as seen from one point on the viewing sphere as a **view** of that object. The views can be retrieved from 2D images around the viewing sphere or a CAD model. For simplicity, we do not distinguish between different views and different objects, as the aim is to jointly detect and localise the object. The next section explains the chosen descriptor for a constellation of edgelets extracted from the edge map, and Section III-B describes how two constellations are matched. Section III-C then discusses using fixed-paths for extracting similar constellations, and Section III-D explains the detection in cluttered environments.

A. Edgelet Constellations

From the view's 2D image, a set of edgelets $\{e_i\}$ are extracted where an **edgelet** is a short straight segment, represented by its centre point (pos) and orientation (ori). The edgelets are sampled from straight lines with a maximum length. A **constellation of edgelets** is an *n-tuple* of edgelets $c = (e_1, .., e_n)$. These can be adjacent, near-by or far apart (Figure 2). Constellations characterise both local parts and global shape by encoding the relative orientations and positions between segments of nearby and distant edges. The vector v_i connects the edgelet e_i with the consecutive edgelet e_{i+1} in the constellation's tuple. For an n-tuple, there are $n-1$ vectors and accordingly $n-2$ angles θ_i where $cos(\theta_i) = (v_i \cdot v_{i+1})/(|v_i||v_{i+1}|)$. The **base angle** θ_0 is the angle between the first edgelet and the first vector. The set of vectors are normalised to a unit vector and θ_0 is used to align with the vertical direction (Figure 2). These normalised and aligned vectors define the **path** connecting the edgelets in the constellation, and is represented by the angles $\Theta = (\theta_0, .., \theta_{n-2})$. In addition to the path, the constellation is described by $\Phi = (\phi_1, .., \phi_{n-1})$ and $\Delta = (\delta_1, .., \delta_{n-2})$ where $\phi_i = \widehat{e_i, e_{i+1}}$ is the relative orientation of consecutive edgelets, and $\delta_i = |v_{i+1}|/|v_i|$ is the relative length of consecutive vectors. The descriptor $f(c) = (\Theta, \Phi, \Delta)$ is translation-, rotation- and scale-invariant. It encodes the joint presence of n-edgelets with certain relative orientations and positions. Notice that the same set of edgelets could result in a different descriptor when the order of edgelets changes in the constellation's tuple. Though this might be perceived as an additional complexity, this complexity is managed by using fixed paths as will be shown in the Section III-C.

B. Matching Constellations

Given a test image, a constellation of test edgelets $c_t = (t_1, .., t_n)$ is represented by the same descriptor f. The test

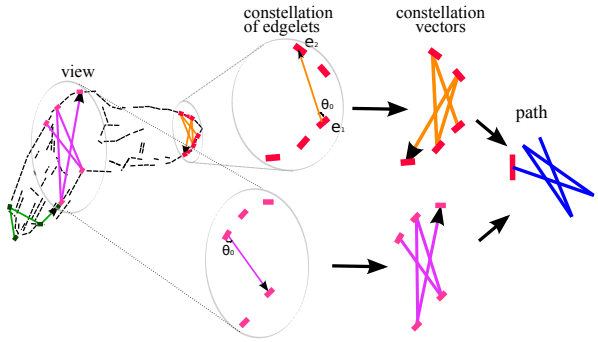

Fig. 2. Constellations of edgelets from one view of an electric screwdriver. The constellations can be local (part-based) or global. The path of the constellation is the normalised angular vectors connecting the constellation's edgelets. The figure shows two constellations of the same path.

constellation c_t matches a view constellation c_v if $\forall i; |\theta_i^v - \theta_i^t| < T_\theta|$, $|\phi_i^v - \phi_i^t| < T_\phi$ and $|\delta_i^v - \delta_j^t| < T_\delta$. When a match is found, an affine transformation H is estimated from the corresponding edgelets, and the remaining edgelets from the view ω are mapped to their corresponding positions and orientations in the test image. An affine homography can be estimated for any constellation for which $|c| \geq 4$ (considering positions only and assuming the edgelets are not co-linear). Iterative closest edgelet (ICE) can then be used to refine the alignment where the closest edgelet to the transformed e_i using the homography H is $\tau(e_i, H)$, i.e.

$$\tau(e_i, H) = \begin{cases} \arg\min_{t_j} d(H(e_i), t_j) & d(H(e_i), t_j) < \beta \\ null & otherwise \end{cases} \quad (1)$$

where a distance measure between two edgelets $d(e_i, e_j)$ assesses the similarity in orientation (using L-1 norm) and spatial closeness (using L-2 norm) [15]

$$d(e_i, e_j) = |e_i.pos - e_j.pos|_2 + \lambda|e_i.ori - e_j.ori| \quad (2)$$

In Equation 2, λ weights the orientation term. The cost of the detection E, using the refined homography H' after ICE convergence, is the scaled sum of the distance measures between corresponding edgelets.

$$E(\omega, H') = \frac{\sum_i \min(d(H'(e_i), \tau(e_i, H')), \beta)}{|\omega|} \frac{|\{\tau(e_i, H')\}|}{|\{e_i; \tau(e_i, H') \neq null\}|} \quad (3)$$

Here the distance measures are summed along with a penalty measure for missing correspondences β, and the scale is estimated by the number of edgelets in the test image to the number they correspond to from view edgelets. An object is then detected at edges $\{\tau(e_i, H')\}$ if $E(\omega, H') < \alpha$.

C. Edgelet Constellations over Fixed Paths

An exponential number of constellations is present in each training and test image. To manage the matching complexity, we adopt a fixed-path approach. All constellations over the same path Θ_k within a tolerance ϵ in the angles, and starting from each edgelet, are found in each view (see Figure 3). This is applied to all views and all objects. For a given Θ_k, the unique descriptor $f^{\Theta_k}(c_i) = (\Phi, \Delta)$ distinguishes different constellations with the same path. This forms an indexed library, where the hash key is the descriptor f^{Θ_k} of size $|(\Phi, \Delta)| = 2n - 3$, and the value is the corresponding view and the n-sized tuple of edgelets that generated the descriptor. Given the training objects, the search is for paths that can find *enough* constellations in the different views. We randomly select paths and rank them by the number of constellations found in all training views. In the experiments next, we use up to 6 paths from 100 randomly chosen path angles. We set the number of edgelets in each constellation to 5 based on preliminary tests. Shorter tuples have a higher chance of hallucinating detections while longer tuples decrease the recall. By keeping a comprehensive library of all constellations of path Θ_k, it is sufficient to extract one constellation of the same path from the object in the test image to produce a candidate detection that is verified using the rest of the view edgelets. A separate library is built for each chosen fixed path, and a quantised hash-table is created so a descriptor would directly lead to candidate matches.

D. Cluttered Environments

For each fixed-path Θ_k, all pairs of edgelets with a relative position that satisfies the base angle θ_0 in Θ_k (within the tolerance ϵ) are highlighted. A pair is chosen at random, and the search for edgelets that complete the fixed-path is carried out (Figure 4). This search is speeded up by pre-calculating relative measurements between all pairs of edgelets. As the tuple is appended, the descriptor f^{Θ_k} is incrementally calculated and compared to the corresponding library. When the partial descriptor cannot match any descriptor in the library, the search is prematurely stopped, and another pair is pursued. To speed up detection only one constellation is pursued from each starting pair of edgelets. Given that the library contains a comprehensive list of all possible view constellations of path Θ_k, the risk of skipping a pair before pursuing all the constellations starting with that pair is acceptable, and proves sufficient during the experiments. When a test constellation matches a view constellation with an error $E(\omega, H') < \alpha$ (Equation 3), the test edgelets $\{\tau(e_i, H')\}$ are explained edgelets based on this detection. These edgelets are removed from any further searches. If all pairs are tested, another fixed path Θ_2 is used. For k fixed paths, the worst case is $O(k \cdot p^2)$. A further modality of handling clutter exists when some prior scene knowledge is available where a depth sensor is used as explained in Section IV-D.

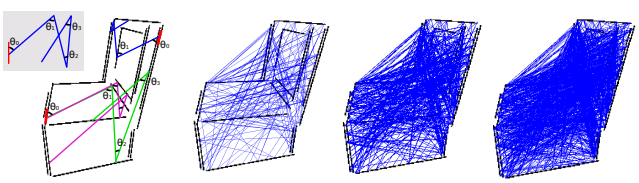

Fig. 3. For a given path (upper left), two constellations from the same starting edgelet, and one from a different starting edgelet are shown (left). All constellations found with $\epsilon = 0.01, 0.015, 0.02$ are also shown.

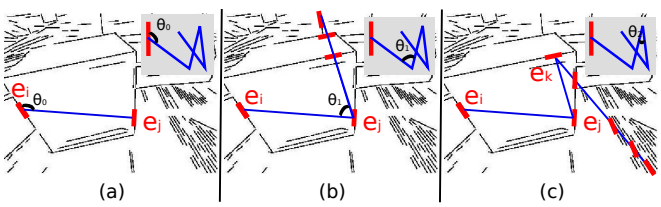

Fig. 4. For a pair of edgelets (e_i, e_j) that satisfy the base angle θ_0 (a), an edgelet e_k that completes the path is sought. When several edgelets are found (b), one is selected and the path is completed (c).

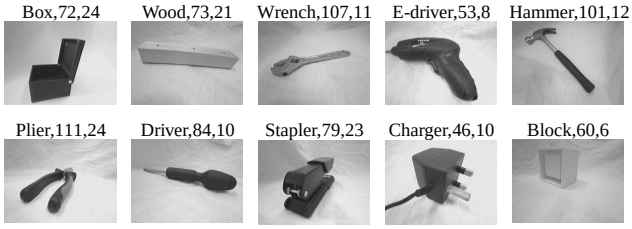

Fig. 5. One view from each of the 10 textureless objects in the dataset with (name, # of views, # of occurrences in test images)

IV. EXPERIMENTS AND RESULTS

We have tested the method on a dataset consisting of 10 real tools and components. Training images were coarsely sampled around the viewing hemisphere (Figure 5) To extract edgelets, we use the line-segment detector in [16]. The edgelet length is set to 10 pixels and ϵ is set to 0.02 (see Figure 3).

Note that due to the initial random order of pairs of test edgelets, and pursuing one path from each pair, detection performance can vary between runs. This is illustrated in Figure 6 which shows the results from 5 runs on a single image along with the edge map reflecting the complexity of the search. In the figure, the block was detected in 4 out of 5 runs while the stapler was detected in 3 out of 5.

A. Detection performance

We first compare the detection performance for the different tools (Figure 7), discarding the false-negative effect from the greedy removal of edgelets. This is evaluated using 100 ground-truthed test images with one to four objects per image. The PASCAL overlap criterion is used to evaluate the detections. The figure shows very low recall for the driver (yellow) and high false positive rate for the wood (red). This is because the wood's rectangular shape can often match other rectangular objects in the background. The e-driver, plier, wrench and stapler achieve the best detection results due to their distinguished shape. The box and the block are ambiguous which increased their FPPI.

B. Expected framerate

We ran the detector at multiple frames per second on a 300 frames video sequence containing 6 out of our 10 objects with surrounding clutter. In this case each frame is analysed from scratch, without considering the detections from previous frames. Figure 8 plots the recall and precision as the frame rate increases from 1 to 17fps. This is run using a non-optimised C code on a standard laptop machine (2.53Hz, 6GB DDR3 SDRAM). Detection can achieve above 50% recall at about 7fps while searching for the 10 textureless objects. The same performance can be achieved at 11fps when searching for a single object.

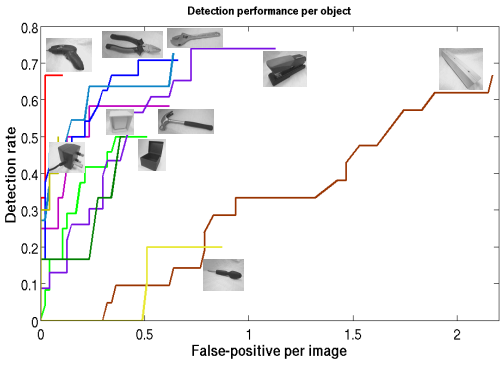

Fig. 7. FPPI versus Detection rate for the tools dataset

For a system that runs on a stream of live images, it is affordable if an object is missed in one frame as long as it can be detected in a few subsequent frames. Figure 9 reports a histogram of the number of frames between correct detections when running at 5fps. For each video, the object to be detected was present in all frames, either in full-view or partially occluded. This was tested for all the 10 objects, and an average of 65% of detections were found within 3 processed-frames from a previous correct detection. This goes up to 86% for the plier and drops to 33% for the driver.

C. Scalability

For the detector to be considered scalable, we expect that as the number of objects and views increases, the detection time scales gracefully. A single non-cluttered test image is selected for each object, and the test focuses on how the performance is affected as the library size increases. Figure 10 plots the average (and standard deviation) detection time from 100 runs compared to the library size as the number of objects in the library increases from 1 to 10. The

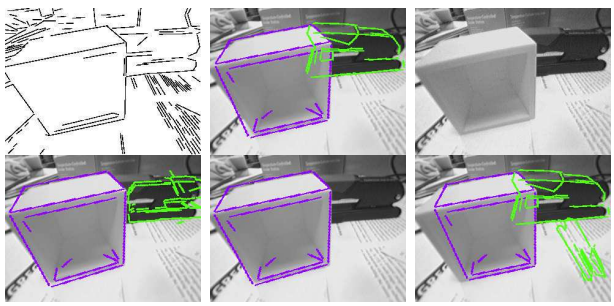

Fig. 6. Five runs on a single image (limited to 200ms). False negatives are caused by not selecting a constellation that belongs to the object.

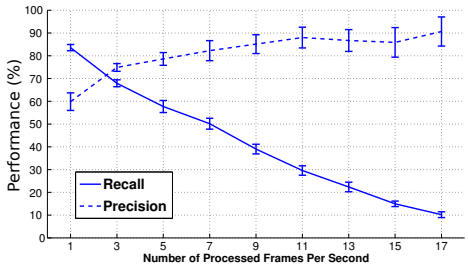

Fig. 8. As the maximum time limit decreases, the recall and precisions are plotted for a cluttered multi-object sequence.

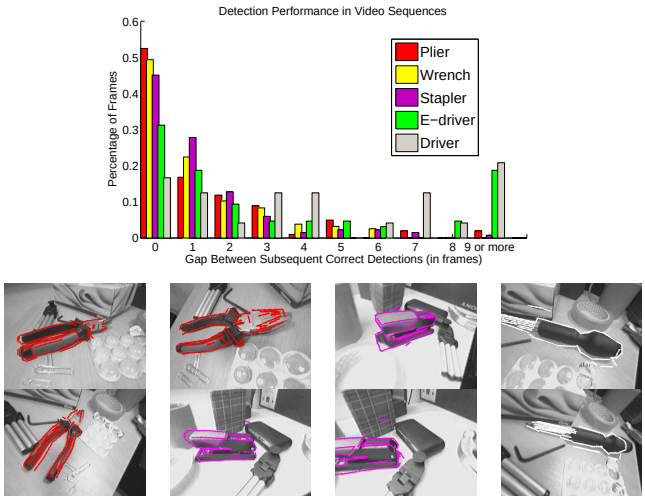

Fig. 9. Histogram of missed number of frames between subsequent detections for five objects, along with frames showing sample detections from the video sequences.

increase in detection time results from comparing to a larger number of descriptors within each indexed bin, as well as assessing ambiguous matches. From the figure, the driver has a nearly linear average time, while the increase in time for detecting the wrench is 2.7x as the library size increases by 10x.

D. Clutter handling and using a depth sensor

The other factor affecting the method's performance is the increase in clutter. Figure 11 is plotted from three video sequences, each starting with the object alone on the desktop then more objects are added. The detector is run at 5fps and the figure shows that the approach can tolerate clutter increased up to 70% of the edge pixels. For more complex environments with a higher percentage of clutter, other techniques can be used to retrieve search regions within which objects of interest can be found. One method to get such search regions is to use depth background subtraction. During training, the point cloud, as retrieved from an RGB-D sensor, is saved as background prior to the introduction of objects. At run-time, the segmented point cloud is used to highlight bounding boxes containing interesting objects. All edges within the bounding box are considered for possible edgelet constellations based on the fixed paths. In our tests the background segmentation is not perfect due to small

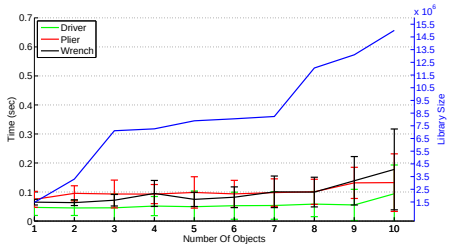

Fig. 10. As the library size increases by 10x (blue line), the average detection time increases by 1.7x-2.7x for 3 objects.

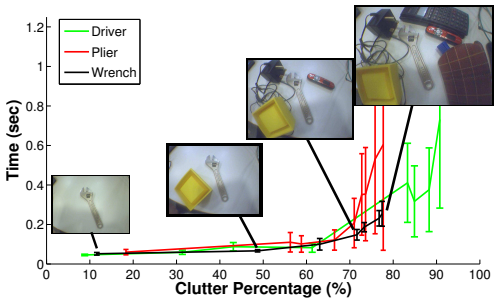

Fig. 11. The average detection time as the percentage of clutter increases for different objects. Sample images from the 'wrench' sequence are shown.

changes in the scene and also addition of non-class clutter objects. Figure 12 shows examples of detections in complex environments using depth background subtraction.

The 2D detections can also be used to align the object within the cloud point. During training, an RGB-D sensor is used to combine each view with a segmented point cloud (Figure 13). After the object is detected, the in-plane homography is used to rotate the corresponding model's 3D point cloud around its centre of mass. The model point cloud is then overlaid on the scene's point cloud with the depth estimated from that of matched scene's points.

We compare this with a 3D object descriptor; the viewpoint feature histogram (VFH) [14]. This descriptor is dependant on obtaining a separate cluster of points containing the object. For each cluster, a 308-D descriptor is calculated and compared to a previously calculated list of descriptors for all training models from various views. The histogram intersection distance is used to compute the mismatch between the descriptors, and the returned point cloud is that of the

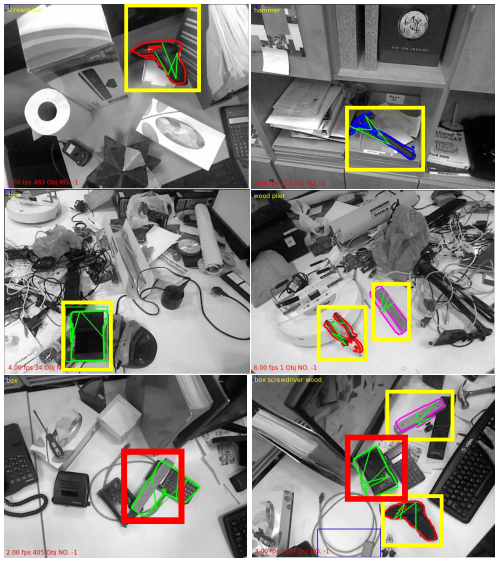

Fig. 12. Detection in complex scenes using depth background subtraction. Bounding boxes show a search region and all edges within the region are considered for the detection. Yellow indicates a correct detection while red indicates a false positive.

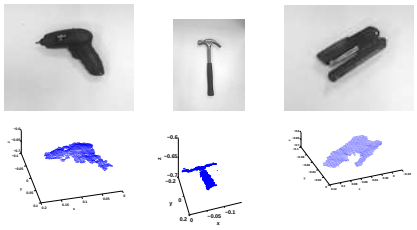

Fig. 13. Using the RGB-D sensor, views are sampled along the hemisphere. The 2D image (above) and the accompanying point cloud segmented from the background (bottom) are shown for one view of three objects.

lowest mismatch. The clustering - and thus the descriptor - is easily corrupted by the presence of occluding or touching objects. Figure 14 compares our detector to the retrieved best-matched model using VFH descriptors on 3 clusters. The VFH descriptors were calculated using the available implementation within Point Cloud Library (PCL). While VFH is able to correctly retrieve the object in the case of the occluded screwdriver, it is confused in the two other cases. Also in the first correctly retrieved point cloud, the mismatch score doubles as the point cloud is occluded by another object. Similar results are shown in Figure 1.

V. CONCLUSION AND FUTURE WORK

We have presented a framework for scalable detection of objects using constellations of edgelets. The constellations are tractable using fixed paths, and are described using translation-, scale- and rotation-invariant descriptors. By using edgelet constellations, the method is more robust to occlusions and camouflaged edges. The method was tested on a dataset of textureless real tools and objects. The technique allows concurrent detection of multiple objects each with views around the viewing sphere. A 50% chance of detection under clutter can be achieved within 140ms. The system can handle up to 70% visual clutter and scales gracefully as the number of objects increases.

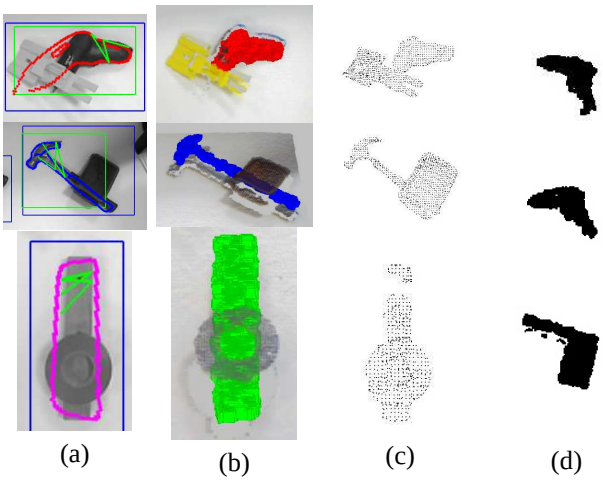

Fig. 14. Our detections (a) can correctly localise the model's point cloud (b) even when the clusters (c) contain occluding or touching objects. The best matched model using VFH descriptors (d) fails to correctly detect the object in two out of the three cases.

As the scene's complexity increases, bounding boxes representing search regions are used to search for objects of interest. In this work, depth background subtraction is used for retrieving such search regions. Other approaches can be alternatively used, like colour or point cloud priors, and these are left for future work.

As opposed to some current 3D object descriptors, this method does not require object clusters to be cleanly segmented. When the object is occluded or is touching neighbouring objects, the path-based detector can still detect and align the model's point cloud successfully.

Acknowledgement We would like to thank Pished Bunnun for his valuable input on the method and the C++ implementation. This work was supported by the EU Project COGNITO FP7-ICT-248290.

REFERENCES

[1] J. Beis and D. Lowe, "Indexing without invariants in 3D object recognition," *IEEE Transactions on Pattern And Machine Intelligence (PAMI)*, vol. 21, no. 10, 1999.

[2] O. Carmichael and M. Hebert, "Object recognition by a cascade of edge probes," in *British Machine Vision Conference (BMVC)*, 2002.

[3] A. Chia, S. Rahardja, D. Rajan, and M. Leung, "Object recognition by discriminative combinations of line segments and ellipses," in *Computer Vision and Pattern Recognition (CVPR)*, 2010.

[4] O. Danielsson, S. Carlsson, and J. Sullivan, "Automatic learning and extraction of multi-local features," in *International Conference on Computer Vision (ICCV)*, 2009.

[5] V. Ferrari, F. Jurie, and C. Schmid, "From images to shape models for object detection," *International Journal of Computer Vision (IJCV)*, vol. 87, no. 3, 2009.

[6] W. Grimson and D. Huttenlocher, "On the sensitivity of geometric hashing," in *International Conference on Computer Vision (ICCV)*, 1990.

[7] S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua, and N. Navab, "Dominant orientation templates for real-time detection of texture-less objects," in *Computer Vision and Pattern Recognition (CVPR)*, 2010.

[8] M. Leordeanu, M. Hebert, and R. Sukthankar, "Beyond local appearance: Category recognition from pairwise interactions of simple features," in *Computer Vision and Pattern Recognition (CVPR)*, 2007.

[9] M. Liu, O. Tuzel, A. Veeraraghavan, and R. Chellappa, "Fast directional chamfer matching," in *Computer Vision and Pattern Recognition (CVPR)*, 2010.

[10] D. Meger, M. Muja, S. Helmer, A. Gupta, C. Gamroth, T. Hoffman, M. Baumann, T. Southey, P. Fazli, W. Wohlkinger, P. Viswanathan, J. Little, D. Lowe, and J. Orwell, "Curious George: An integrated visual search platform," in *Canadian Conference on Computer and Robot Vision (CRV)*, 2010.

[11] K. Mikolajczyk, A. Zisserman, and C. Schmid, "Shape recognition with edge-based features," in *British Machine Vision Conference (BMVC)*, 2003.

[12] A. Opelt, A. Pinz, and A. Zisserman, "A boundary-fragment-model for object detection," in *European Confernece on Computer Vision (ECCV)*, 2006.

[13] C. Rothwell, A. Zisserman, D. Forsyth, and J. Mundy, "Canonical frames for planar object recognition," in *European Conference on Computer Vision (ECCV)*, 1992.

[14] R. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3D recognition and pose using the viewpoint feature histogram," in *Intelligent Robots and Systems (IROS)*, 2010.

[15] J. Shotton, A. Blake, and R. Cipolla, "Contour-based learning for object detection," in *International Conference on Computer Vision (ICCV)*, 2005.

[16] R. G. von Gioi, J. Jakubowicz, J. Morel, and G. Randall, "LSD: A line segement detector," *IEEE Trans. on Pattern and Machine Intelligence (PAMI)*, vol. 32, no. 4, 2010.

[17] C. Wiedemann, M. Ulrich, and C. Steger, "Recognition and tracking of 3D objects," in *DAGM Symposium on Pattern Recognition*, 2008.

[18] Y.Lamdan and H.J.Wolfson, "Geometric hashing: A general and efficient model-based recognition scheme," in *International Conference on Computer Vision (ICCV)*, 1988.