Active Recognition and Pose Estimation of Household Objects in Clutter

Kanzhi Wu∗ and Ravindra Ranasinghe and Gamini Dissanayake

Abstract—This paper presents an active object recognition and pose estimation system for household objects in a highly cluttered environment. A sparse feature model, augmented with the characteristics of features when observed from different viewpoints is used for recognition and pose estimation while a dense point cloud model is used for storing geometry. This strategy makes it possible to accurately predict the expected information available during the Next-Best-View planning process as both the visibility as well as the likelihood of feature matching can be considered simultaneously. Experimental evaluations of the active object recognition and pose estimation with an RGB-D sensor mounted on a Turtlebot are presented.

I. INTRODUCTION

The use of autonomous mobile service robots in close cooperation with humans to help individuals with special needs is widely acknowledged. 3D object recognition and pose estimation (commonly known as object localisation) are indispensable tasks for these service robots that enable them to dynamically perform grasping and manipulations tasks. These robots are expected to operate in everyday environments such as offices and kitchens typically full of challenging conditions such as presence of clutter, varying lighting conditions and occlusion. These conditions pose extra challenges to localise the objects.

It has been recognised that the object localisation cannot be accomplished using measurements from a single viewpoint. Several previous research works have proposed to use multiple observations and combine the knowledge from these multiple views to recognise the objects unambiguously[1][2][3]. These techniques use autonomous (or active) approaches for scene exploration, hence to increase information gain and the quality of the information gathered to improve the recognition probability. While there have been some successes, a principled framework for the combined problem of active object recognition and localisation that incorporates the impact of the change in viewpoint on the quality of information gathered is not yet available.

Motivated by the need and the challenges, the research work presented in this paper proposes an active recognition strategy suitable for dynamically exploring a cluttered scene while localising the pre-trained objects. In this work an RGB-D sensor is used to build a feature rich model for each object which consists two sub-models: sparse feature cloud model and dense point cloud model. The sparse feature cloud model is used for object recognition and pose estimation. Features are augmented with information acquired through a sensor model that characterises the behaviour of the features when observed from different relative poses. The dense point cloud model is used in view reconstruction. Visibility is considered in conjunction with feature characteristics making it possible to accurately predict the potential information gain from a given viewpoint. The advantages of this strategy is illustrated using a path planning strategy that selects the next best viewpoint based on the information gain.

The key contributions of the work presented in paper are 1) an information rich model for robust and accurate prediction in Next-Best-View decision making, 2) a novel graph based algorithm for correspondences search, that exploits the 3D geometry constraints and 3) a practical active object recognition system which is successfully implemented on Turtlebot. The paper is structured as follows: Recent progress and practical experiments in active object recognition are reviewed in Section II. Section III details the object modeling and the key attributes attached to each feature in the proposed models. Section IV explains the recognition and pose estimation framework for a single observation and the active recognition strategy. The proposed system, implemented using an RGB-D sensor mounted on a Turtlebot and the active object recognition experiments are discussed in Section V. Section VI concludes this paper and provides a brief summary of the future work.

II. RELATED WORK

Recognising and estimating the pose of rigid objects are critical problems in robotic research. Gordon and Lowe[4] proposed a solution that matches SIFT [5] features between 2D images and 3D object models and compute the relative pose between the observed object and the camera through bundle adjustment. Based on [4], Collet et. al [6] presented a framework for Multiple Object Pose Estimation and Detection (MOPED). The object recognition and pose estimation is combined into one optimisation loop named Iterative Clustering Estimation (ICE). MOPED is able to provide real-time performance on practical robotic platforms. Recently, exploiting the availability of RGB-D sensors, Tang et. al [7] presented an instance recognition system by incorporating depth information. Their approach yields significant results on public Challenge and Willow datasets. Based on Tang’s work, Xie et. al [8] further improved the performance on these two datasets using dense feature representation and multimodal blending. Bo [9] introduced hierarchical matching pursuit (HMP) which learns features hierarchically as a multi-layer sparse coding network using RGB-D images. It is shown that a deep learning technique, based on the learned

∗Kanzhi Wu is supported by University of Technology, Sydney-China Scholarship Council (UTS-CSC) Joint Scholarship Centre for Autonomous Systems, University of Technology Sydney, Sydney, Australia kanzhi.wu, ravindra.ranasinghe, gamini.dissanayake@uts.edu.au

RGB-D features, enables superior performances on various datasets.

Under real world scenarios, objects maybe occluded or out of the Field-of-View (FoV) of the sensor. In order to capture the full environment, actions are needed to acquire more information. In order to solve the active recognition problem, the location that provides the Next-Best-View (NBV) has to be determined. Dutta-Roy [2] lists the state-of-the-art approaches in NBV for active object recognition. Recently, the work of Eidenberger et. al [10] on active recognition in cluttered environments presented a probabilistic planner which is realised by Partially Observed Markov Decision Process (POMDP). This approach is able to deal with complex, occluded environments with arbitrary poses of the objects. Focusing on similar objects, Browatzki et. al [1] proposed another strategy on a humanoid robot specifically to resolve the ambiguities between similar objects. Kriegel’s work [11] further combined active recognition and modeling for unknown object into a unified framework. In this case, the model of previously unknown object will be constructed online and dynamically added into the dataset. Also by using an RGB-D sensor, Lutz et. al [12] fused multiple recognition systems into one probabilistic framework and achieves better performance compared to the individual method. Recently, Atanasov [13] addressed active recognition problem as a nonmyopic planning problem, minimising the spend energy for sensor movements and decreasing the uncertainty of incorrect recognition likelihood at the same time.

In contrast to existing approaches which focus on either novel active planning strategies or space representation methods, in this work, we argue that a sensor-object model that characterises the influence of the range and viewing direction on the feature matching process is critical in active recognition. Using experimentally generated thresholds, our approach is able to provide much more accurate predictions as to the potential information available making it feasible to generate better robot trajectories for active recognition.

### III. INFORMATION RICH OBJECT MODELING

#### A. Models for Active Object Recognition

![Object modeling using RGB-D sensor](image)

(a) Dense point cloud  
(b) Sparse feature cloud

Fig. 1. Model of object “Fruity Bites”

In the proposed approach, an object is represented using two models: dense point cloud model $M_d$ and sparse feature cloud model $M_s$. $M_d$ is a dense RGB point cloud which characterises the shape and texture information of the object, shown in Fig. 1(a). In $M_s$, each feature $f_s$ consists of the 3D position in an object coordinate system and RGB information of the point $p_s$. A local image descriptor such as SIFT of the point $d_s$ is included in $f_s$. Both $M_d$ and $M_s$ can be easily built. During the experiments reported in this paper, an off-the-shelf Simultaneous Localisation and Mapping technique was used to construct these models by positioning an RGB-D sensor around the object as shown in Fig. 2. The feature correspondences were established using multiple steps of RANSAC (RANdom SAmple Consensus[14]). These were then used in the ParallaxBA (Parallax Bundle Adjustment[15]) to obtain accurate pose estimates. Using the optimised poses, $M_d$ is obtained by transforming point cloud of different observations and then aligning the point clouds into one. The model $M_d$ can be further refined using filtering and surface fitting. Feature locations in $M_s$ is generated from the optimised coordinates of each pair of correspondences.

#### B. Information Rich Attributes for Better Prediction of Viewpoint Quality

One of the most important tasks that need to be accomplished for active perception is the prediction of the quality of information that is likely to be available from a given viewpoint. Information quality depends on the amount of features visible as well as the ability to associate the observed features to those present in the object model. Widely used feature descriptors such as SIFT and SURF, although designed to be robust to changes in scale and multiple deformations, fail to establish associations if such changes are too large. Fig. 3(a) illustrates the impact of the distance from the camera to the object for 3 objects (“Sanitarium”, “Fruity Bites” and “Belvita”). While more than 1000 features are captured for Sanitarium object placed at a distance of 0.55 m from the camera, only 50 of these features can be detected and correctly matched at a distance of 1.2 m. It is seen from 3(a) that the number of correct matches decreases...
approximately exponentially with the increase of the distance between features and the camera. It was also observed that the impact of the change in scale is a property of the region from which the feature is extracted and as such cannot be captured in one general formula. 3(b) shows how the number of matches change due to the variation of the angle between the surface normal and the camera axis at a range of 65 cm. It can be easily seen from this figure that change in the view angle also has a significant impact on the matching ability.

![Figure 3](image)

(a) Scale variation analysis  (b) Viewpoint variation analysis

Fig. 3. The number of correct matches changes with the change of scale and viewpoint.

We propose to address this issue by attaching two attributes for each feature $f$: maximum observable distance $d_{\text{max}}$ and maximum observable angle $\alpha_{\text{max}} = g(d)$ which is a function of the distance $d$ between the sensor and the feature. These extra attributes are used in prediction and depending on the type of the descriptor, $d_{\text{max}}$ describes the maximum distance under which the feature can be extracted and potentially correctly matched to the descriptor $d_k$ in the model. $d_{\text{max}}$ is able to give a quantitative indication for the scale invariance of each feature. $\alpha_{\text{max}}$ denotes the upper bound of angle between the normal vector and the viewpoint, under which the associated feature can be repeated and correctly matched again. $\alpha_{\text{max}}$ is a function of the distance between the camera and the feature. The function $g(d)$ allows larger $\alpha_{\text{max}}$ in further distance and smaller $\alpha_{\text{max}}$ in closer distance.

The two parameters $d_{\text{max}}$ and $\alpha_{\text{max}}$ are set empirically. During object modeling, the local image patch of a feature is rescaled into multiple levels to obtain $d_{\text{max}}$. Through extracting features and descriptors on scaled images, $d_{\text{max}}$ is evaluated for each feature $f_k$ in the model. $\alpha_{\text{max}}$ is computed in a similar way by warping the local image patch using multiple levels of affine transformations. Section. V-B shows the impact of adding these two attributes in predicting the number of correct matches for new viewpoint.

IV. ACTIVE OBJECT RECOGNITION AND POSE ESTIMATION SYSTEM

A. Object Recognition and Pose Estimation Using a Single View Point

This section presents the framework for obtaining initial hypotheses as to the objects present in a cluttered scene and their relative poses. The details of the framework based on the algorithm depicted Fig. 4 is outlined below. A detailed analysis of the computational cost for this step is presented in Section. V-B.

1) RGB-D Segmentation: The input RGB-D point cloud of the whole scene is first segmented into multiple groups using the pre-processing method described in Richtsfeld [16]. A computationally light-weight version of [16] is implemented on the Turtlebot robotic platform. Two different models which describe the geometry shape of the object, planes and NURBS (Non-Uniform Rational B-Splines, Chapter. 10 in [17]), are used for fitting and segmenting the surface patches in the point cloud. Given that most man-made household objects have planar surfaces or curved shape that can be easily described using NURBS, these two models can capture informative patches in the input point cloud which are most likely to contain objects. Examples of RGB-D segmentation results are shown in Fig. 5.

![Figure 5](image)

Fig. 5. RGB-D segmentation results from cluttered environment. The 1st row is the image of input RGB point cloud and the 2nd row shows the segmentation results. A number of small patches, that are later eliminated can also be seen.

2) Feature Extraction and Clustering: Variety of local image feature descriptors such as SIFT[18] and SURF[5] are available for use in this step. Current algorithm uses SIFT features, extracted using either CPU or GPU implementation(SIFT-GPU from Wu [19]). Since both the RGB and the depth information are available in RGB-D sensor, the feature coordinate $p_k^C = [u_k^C, v_k^C]$ in the image space and corresponding 3D coordinate $p_k^S = [x_k^S, y_k^S, z_k^S]$ in sensor coordinate system are also stored together with the feature descriptor for 3D-3D pose estimation.

Using the segmented results from Section. IV-A.1, each point $p_k^C$ is labeled with $l_k$ which denotes the group it belongs to. The extracted features are clustered using $l_k$. The segmented patches which do not have enough number of extracted features are regarded as outliers and are removed. After this step, feature groups which potentially consist of one object can be generated as shown in Fig. 6.

3) Feature Matching: Feature correspondences are obtained using the fast ANN (Approximate Nearest Neighbour) searching method [20] to reduce the computational effort required. As shown on the right block of Fig. 4, a descriptor searching tree is built by joining all the features of every object in the dataset. Therefore, observed features in the same
group will be matched with features from different models. The ratio test is used as a way of confirming the nearest neighbour. If the ratio between the nearest distance and the second nearest distance is less than a specified threshold, the match is regarded as a success.

4) Searching Consistent Matches: The aim of this step is to find the consistent correspondences between the observation and the dataset. Even with the use of ratio test, there is still a possibility that incorrect matches between group $g$ and model $m$ in the dataset exist. A novel graph based algorithm for correspondences search, that exploits the available 3D information is proposed for this purpose. We note here that unlike MOPED [6] that relies on an RGB camera, the presence of depth information makes it possible to explore the geometrical constraints between each pair of matches to find the consistent correspondences in a simpler way.

A set of correspondences is notated as $c = \{c_1, c_2, \ldots, c_n\}$ where $c_i = (p_i^M, p_i^O)$. $p_i^M$ is the 3D point from the model and $p_i^O$ is the 3D point from observations. Two correspondences $c_i$ and $c_j$ are regarded as consistent according to Eq. 1. After constructing the graph, largest consistent clique in the graph which has at least $n_c$ number of vertices is located.

The proposed scheme is simple yet effective and was found to yield better performances compared with traditional methods such as RANSAC. The algorithm first traverses all the vertices in graph $g$, and count the number of linked edges $n_i$ with each $v_i$. Then the most connected vertex $v_c$ of the graph is located and compute the number of connected edge of $v_c$ as $n_c$. If $n_c > n_v$, all the vertices connected with $v_c$ and $v_c$ itself are considered as a consistent set and

$$||p_i^M - p_j^M||_2 - ||p_i^O - p_j^O||_2 < \sigma_d$$  (1)

where $\sigma_d$ is the noise threshold and $||p||_2$ is the L2-norm of $p$. Using Eq. 1, the relational graph $g = (v, e)$ of all the correspondences $c$ can be built. Each correspondence $c_i$ is represented as vertex $v_i$ of the graph and an edge $e_{i,j}$ is linked if $c_i$ and $c_j$ are consistent according to Eq. 1. After constructing the graph, largest consistent clique in the graph which has at least $n_c$ number of vertices is located.
removed from the graph. The graph is iteratively updated until \( n_c < n_v \). Once the most connected vertex of the current graph is less than \( n_v \), edges, all the remaining vertices are treated as outliers. The algorithm is summarised below.

**Algorithm 1: Consistent correspondences searching**

\[
\begin{align*}
\text{Input:} & \quad n \text{ pair of 3D matches } c = (p^M_i, p^O_i); \\
& \quad g = (v, e) \\
\text{Output:} & \quad \text{Consistent sets } \{s_1, s_2, \ldots\}; \\
& \quad s_i = \{c_{i_1}, c_{i_2}, \ldots, c_{i_k}\}
\end{align*}
\]

1. \( g = \text{Graph}(c); \)
2. \( v_{\max} = \text{MaxVertex}(g); \)
3. \( \text{while } |\text{ConnectedVertices}(v_{\max})| > n_v \text{ do} \)
4. \( \hat{v} = \{v_{\max}, \text{ConnectedVertices}(v_{\max})\}; \)
5. \( v' = v - \hat{v}; \)
6. \( g' = \text{Graph}(v'); \)
7. \( v_{\max} = \text{MaxVertex}(g'); \)
8. \( /\ast \hat{v} = \text{ConnectedVertices}(v_{\max}); \)
9. \( /\ast v = \text{MaxVertex}(g); \)
10. \( /\ast \text{returns the vertex which has the most linked edges}; \)

5) **SVD based Pose Estimation**: Given a set of 3D-3D correspondences, the pose estimation problem can be solved in closed form. Assuming same notations as IV-A.4, a point from the model \( p^M_i \) and the corresponding point from an observation \( p^O_i \) satisfy

\[
p^O_i = R p^M_i + t + \tau_i
\]

where \( \tau_i \) is the noise. Given at least 4 pairs of 3D-3D matches \( (p^M_i, p^O_i) \), the optimal \( R \) and \( t \) which minimise the sum of errors \( \sigma^2 \) below can be solved using Singular Value Decomposition (SVD) from [21]

\[
\sigma^2 = \sum_{i=1}^{n} ||p^O_i - R p^M_i - t||^2
\]

6) **Post-processing**: Post-processing is focusing on two problems:

- Remove correct redundant objects and incorrect overlapping objects:
  In the experiments, occasionally one object is grouped into separate clusters. This leads to a recognition result containing one object with multiple similar poses. We can easily detect this situation and combine the clusters together to obtain a single pose estimation using Section IV-A.5. When the recognition result shows different objects that are overlapped, the objects that has most matches and smallest reprojection errors is selected as correct and the remaining objects are deleted.
- Generate virtual environment using recognised objects:
  The poses of recognised objects are initially available with respect to the camera coordinate system \( F_c \). These are then translated into a fixed world coordinate system \( F_w \).

A virtual representation of the scene in the coordinate frame \( F_w \) is then created. This virtual environment is used in Section IV-B to predict the potentially visible features given a specific viewpoint.

**B. Next-Best-View Selection using Information Rich Model**

The virtual representation of the scene that encapsulates the current belief about the objects present and their poses can now be used to evaluate the quality of the information that can be expected if the robot moves to a new viewpoint. It is now, therefore, possible to evaluate the utility of nearby locations so that the robot can then be moved to the Next-Best-View. This process can be repeated until some termination criteria such as, stable object recognition and pose estimation, over a set number of moves is reached.

The overall flowchart of the active object recognition system is shown in Fig. 7. It begins with the virtual representation of the scene generated based on the information captured at the first robot pose. The three-step strategy is then used to predict the observable features which can be matched correctly from a new viewpoint.

1) **Raycasting**: In this step, all the features that are not visible from the new viewpoint due to occlusion by itself or other objects are rejected using raycasting by an octree structure [22]. The world space is voxelised and given a starting point (observation position) and destination point (feature), we obtain all the voxels intersected by this straight-line. Therefore, it is possible to find out whether any features exist in the intersected voxels. This step can be completed in few milliseconds. The detailed time performance is presented in Section V-B.

2) **Scale analysis**: Even though a feature point is not occluded, as discussed in section III-B there is no guarantee that this feature is detected and matched correctly. In the scale filtering step, a feature is removed from the candidate set if the distance between the sensor and the feature point, \( d_f \), is larger than \( d_{\text{max}} \).

3) **Viewpoint analysis**: Another key factor that influences the repeatability of feature matching is the viewpoint variation. Under the same distance, a feature can be re-detected and matched correctly under a limited range, as shown in 3(b). In experiments on robotic platform shown in Section V-A, a consistent threshold \( \alpha_{\text{max}} \) is set for all the features in the models. The vector from the feature to the sensor is denoted as \( \mathbf{v}_f \) and the normal vector of the local patch is denoted as \( \mathbf{v}_n \). If the angle between \( \mathbf{v}_f \) and \( \mathbf{v}_n \) is larger than \( \alpha_{\text{max}} \), we assume that the feature cannot be correctly matched.

Using the above three filtering steps, number of potentially correct matches \( n_m \) under a given viewpoint can be predicted. The focus of this paper is on the evaluation of the view point quality. As such, in order to illustrate the effectiveness of the algorithms proposed, a sequence of steps for the robot motion is generated using a simple greedy strategy that explores the nearest neighbours. For each neighbouring voxels, the current observation can be
compared with the predicted observation to identify, \( n_{\text{new}}^{\text{new}} \), the number of new matches which have not been detected before. The Next-Best-View is selected as the voxel which has the highest \( n_{\text{new}}^{\text{new}} \). This criterion always enables the camera to move to a new position to acquire new information. The path planning in each step follows the logic of information gain which is widely used in active object recognition and autonomous object modeling[23]. Given that the quality of information gathered from a given viewpoint is available, work proposed in this paper can be adapted for use with more sophisticated trajectory planners.

**V. Experiments and Discussion**

**A. Robotic Platform**

During the experiments, a Turtlebot with a Microsoft Kinect mounted on the top of it (shown in Fig. 8) was used. The motion of the RGB-D sensor is restricted to 2D space. Eight different objects were placed on the table with different orientations. Due to the presences of occlusions, all objects cannot be observed at the same time from one viewpoint, thus the robot needs to move in order to recognise all the objects and to estimate their poses.

In the experiments reported in this paper, the environment was divided into multiple cells and the robot was manually placed in each cell as dictated by the planning algorithms. For simplicity, we also assume that the robot orientation will be such that the camera will face towards the objects from each of the grid cells. The locations in which the robot can be positioned are shown in Fig. 9. 180 RGB-D images were collected by placing the robot in these locations so that the algorithm can be evaluated off-line.

The detailed active recognition pipeline used in evaluating the proposed strategy is as follows.

1) Start from red position in Fig. 9, given the observation data captured in that position, we start object recognition and pose estimation according to Fig. 4.

2) Based on the reconstructed environment using the estimated objects and their poses, we predict the number of new correct matches that can potentially be achieved for the \( n \) neighborhood positions.

3) The position with the largest number of correct matches which have never seen before is regarded as the Next-Best-View. We select the image and the...
point cloud captured from that position to simulate the movement.
The active recognition pipeline algorithm iterates through the above steps until the camera is moved back closer to the starting position.

B. Results and Discussion

1) Prediction of Possible Matches: Fig. 10 shows the differences between prediction results with and without using $d_{\text{max}}$ and $\alpha_{\text{max}}$ as additional information stored in the object model. Fig. 10(a) is the actual acquired image in frame 2 and Fig. 10(b) shows the matched features when robot moves to the 2nd frame. If $d_{\text{max}}$ and $\alpha_{\text{max}}$ are not used, the prediction is as Fig. 10(c). It can be seen that the predicted matches using the proposed strategy that uses this additional information shown in Fig. 10(d) are much closer to what actually happens.

![Figure 10. Object recognition and pose estimation results in different steps of the path](image1)

2) Planned Path and Reconstructed Scenario: Using the collected data at every pose shown in Fig. 9, we generate a path which can cover the whole space on the table. The planned path is shown in Fig. 11. When the robot moves along this path, we can recognise the objects one-by-one and finally cover all objects. Parts of the reconstructed scene during the movements are shown in Fig. 12. In Fig. 12(d), all the objects are recognised with accurate pose estimation results.

![Figure 11. Planned trajectory for active object recognition and pose estimation](image2)

(a) Virtual scenario in step 3, recognised 3 of 8 objects
(b) Virtual scenario in step 13, recognised 5 of 8 objects
(c) Virtual scenario in step 47, recognised 7 of 8 objects
(d) Virtual scenario in step 52, recognised 8 of 8 objects

![Figure 12. Object recognition and pose estimation results in different steps of the path](image3)

3) Computational Cost: Approximate time consumption for each of the steps in the algorithm during the robot trajectory in Fig. 11 is shown in Table I. The overall time consumption of recognition and estimation for each step is less than 1.5s even without using GPU computation. The voxelisation and the octree construction time is only depends on the size of the work space, therefore it is approximately the same during each iteration. Predicting matches for one voxel using raycasting, scale analysis and angle analysis takes less than 10 ms for about 3000 feature points. The overall prediction time is depending on the number of searching neighbourhood voxels.

In practice a robot is likely to be able to take measurements from every cell in the environment. Although, in the present implementation we only search for the nearest neighbours, it is possible to construct a graph and conduct a search over a longer planning horizon. In our unoptimised code, once the space is voxelised and octree is constructed, As the evaluation of each cell takes only around 10 ms depending on the number of features. Since the fact that candidate evaluations can be done in parallel, we expect that it will be feasible to employ more sophisticated planning algorithms based on the concepts presented in this paper.

VI. Conclusion

In this paper, an active object recognition and pose estimation system is presented which is able to localise cluttered household objects in the environment. By adding two more
attributes: maximum observable distance and maximum observable angle, to the model, our methods is able to provide much more realistic prediction in Next-Best-View decision making. The active recognition trajectory is generated by joining nearest neighbour NBV problem into a consecutive process. We propose a complete pipeline for object recognition and pose estimation based on this concept. The proposed system is implemented on a Turtlebot and the generated trajectory is able to cover all objects in the environment with accurate relative pose.

In the future work, we will focus on two aspects: 1) a more practical robotic platform which is able to demonstrate online object recognition and pose estimation; 2) an advanced framework which is able to take systematic uncertainties into consideration including recognition, pose estimation and most importantly sensor motion.

REFERENCES


