Active vision for dexterous grasping of novel objects

Arruda, Ermano; Kopicki, Marek; Wyatt, Jeremy

DOI:
10.1109/IROS.2016.7759446

Active vision for dexterous grasping of novel objects

Ermano Arruda Marek Kopicki Jeremy L. Wyatt

***Abstract*— How should a robot direct active vision so as to ensure reliable grasping? We answer this question for the case of dexterous grasping of unfamiliar objects. When an object is unfamiliar, much of its shape is by definition unknown. An initial view will recover only some surfaces, leaving most of the object's surface unmodelled, and also leaving shadow regions which may or may not contain obstacles. These two features make it difficult both to select reliable grasps, and to plan safe reach-to-grasp trajectories. Grasps typically fail in one of two ways, either unmodelled objects in the scene cause collisions, or object reconstruction is insufficient to ensure that the grasp points provide a stable force closure. These problems can be solved more easily if active sensing is guided by the anticipated actions. Our approach has three stages. First, we take a single view and generate candidate grasps from the resulting partial object reconstruction. Second, we drive active vision to maximise surface reconstruction quality around the planned contact points. During this phase the anticipated grasp is continually refined. Third, we direct gaze to unmodelled regions that will affect the planned reach to grasp trajectory, so as to confirm that this trajectory is safe. We show, on a dexterous manipulator with camera on wrist, that our approach (85.7% success rate) outperforms a randomised algorithm (64.2% success rate). Our approach also matches the grasp success of our original method, but with fewer views to pick the grasp.**

I. Introduction

Grasping of novel objects is a hard problem on which there has been steady progress [10], [11], [8], [14], [7], [16], [6], [3], [15], [4]. We now possess methods that are able to generate dexterous grasps for unfamiliar objects, using incomplete object reconstructions. Nonetheless, the reliability of grasping rises with the quality and completeness of the reconstruction available. Given an active vision system, we would like to minimise the number of views taken, while maximising grasping reliability.

At the root of the difficulties is a chicken and egg problem. On the one hand, given that the initial point cloud can be highly incomplete, it is hard to plan a reliable grasp to begin with. On the other hand, if we knew the likely planned grasp then we could direct gaze more efficiently. In this paper we solve this problem by employing a grasp planner that can generate grasps for novel objects in the face of fragmentary reconstructions. We use grasp candidates to guide active vision, and the results of active vision to refine grasp planning.

We gratefully acknowledge support of FP7 grant IST-600918, PacMan, and a studentship from Brazilian Science without Borders for Ermano Arruda.

Arruda, Kopicki and Wyatt at CN-CR, University of Birmingham, Edgbaston, Birmingham, United Kingdom, B15 2TT, Tel.: +44-121-4144788, Fax: +44-121-4144281, {exa371,msk, jlw}@cs.bham.ac.uk

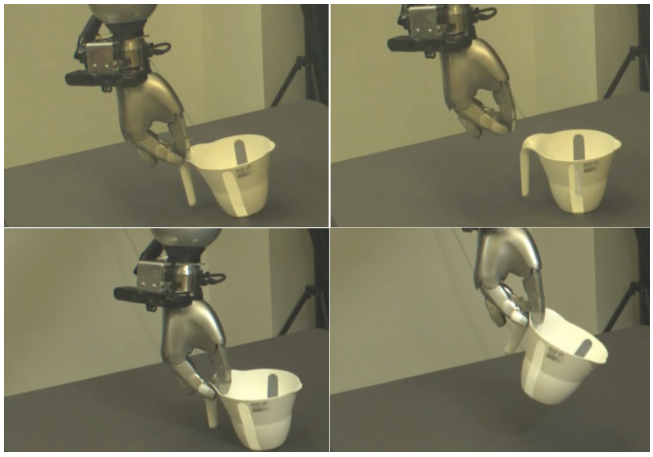

Fig. 1. Grasp failure and grasp success. The top row shows a failed grasp without active view selection. The bottom row demonstrates a successful grasp after active view selection. The difference was due to the quality of surface reconstruction close to the planned grasp points.

First we describe related work, and then proceed to describe our active vision method. This has two parts, a routine driven by the planned contact points, and a routine driven by the need to ensure a safe reach to grasp trajectory. We then present experimental results on 14 novel objects, comparing our method with a randomised view planner.

II. Related Work

Active vision, or more generally active perception, is defined as the study of modelling, planning and control strategies for perception when the sensor can be actively moved [2]. The field of active perception for robots started with work by [2], [1]. The greatest advantage of active perception is that many problems that are hard to solve in the passive observer paradigm become easier. In the context of manipulation, researchers have focused on devising strategies for view selection based on recovery of the full shape of the object to be grasped [12], [5].

Nonetheless, for most practical manipulation purposes, full object reconstruction is either too costly or simply infeasible. It is also typically redundant, since most of the time only a limited portion of the object surface is in contact during a grasp. These practical considerations were taken into account by a number of works. For instance, the approach proposed by [10], [11] is able to transfer previously demonstrated grasps to new objects without the need for grasp force analysis, and is able to cope with an incomplete point clouds of the target object. Additional efforts were made by [8], focusing on task and grasp transferability from limited

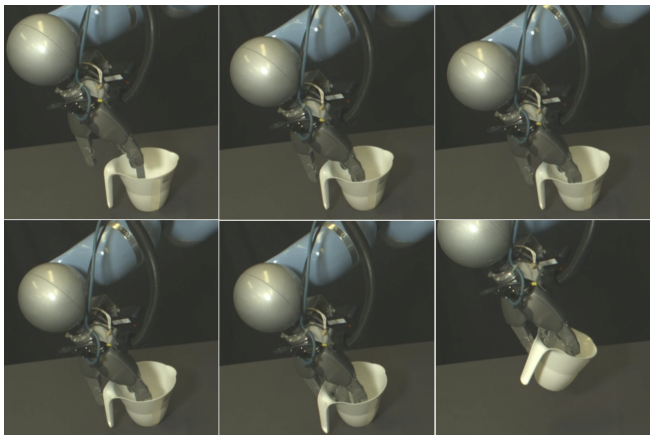

Fig. 2. An example grasp using random views. The grasp is successful. However, it can be seen that the grasp trajectory starts pushing the object aside with its fingers long before the final grasp closure takes place in the last picture on the right. This is a typical scenario that leads to failed grasps.

training data, i.e. demonstration and partial object point clouds. The work done by [7], focused on learning grasps by letting the robot autonomously explore and try grasps while at the same time being able to transfer those self-discovered grasps to novel objects. In [15], efforts were made towards finding stable grasps given limited visibility of object shape from cluttered scenes. The problem of shape incompleteness is dealt with by Bohg et al. [4] by trying to fill the gap between the missing parts of the objects using symmetry assumptions.

Although there has been progress in grasping in the face of partial reconstruction and novel objects, there exists a clear need for active perception so as to decide when and how much to fill in the missing information. In addition, we wish to ensure robot safety, avoiding hardware damage due to unexpected collisions. We now proceed to describe our proposed approach to tackle these issues.

III. View Selection

We first sketch our method, and then proceed to the details. The robot begins by taking a single view from a fixed location of the scene. A depth camera mounted on the robot's wrist is used. The robot is then able to choose views, which in turn provide incomplete point clouds of the object. A dexterous grasp planning algorithm is then run, which generates a large number of candidate grasps on the partial point cloud for the object. These grasps will typically assume the existence of graspable surfaces on both sides of the surface defined by the point cloud. The predicted contact locations are then used to drive the next view. The next view is chosen to maximise the quality of the point cloud at the planned contact locations. If a grasp cannot be found, we employ information gain view planning, using a 3D occupancy map. Once the quality of the relevant surface reconstruction is sufficiently high, or a limit on the number of views is reached, the grasp is fixed. Then a second phase of active vision aims to verify a safe path to the grasp location. To achieve this we again use the 3D occupancy map. This is used to calculate the probability of a collision free trajectory. Active views for safety are driven to reduce the average entropy in cells through which the candidate reach-to-grasp trajectory passes. This ensures a safe grasp. We now proceed to describe the representations, and the three criteria used to drive active vision at different stages (contact based, information gain, and safety based).

Algorithm 1 Next Best View Exploration

function NEXTBESTVIEW($\Xi, \Gamma, \Lambda, G, V, T$)
  $\Omega = \emptyset$ ▷ Most recent found contact points
  $\tau = None$ ▷ Most recent found grasp trajectory
  $stop = false$
  while not $stop$ **do**
    $\overset{*}{\xi} = selectNBV(\Xi, \Gamma, \Lambda, V, \Omega, \tau)$
    $V = append(V, \overset{*}{\xi})$ ▷ Appending $\overset{*}{\xi}$ to V.
    $\gamma = capture(\overset{*}{\xi})$ ▷ Point cloud from pose $\overset{*}{\xi}$.
    $\Gamma = \Gamma \uplus segmented(\gamma)$
    $\tau, \Omega = findGrasp(\Gamma, G)$ ▷ Grasp planning with current Γ_t based on Kopicki, Wyatt et al [10].
    $\Lambda = updateOcTree(\Lambda, \gamma, \overset{*}{\xi}, \Omega)$
    $T = append(T, \tau)$
    $stop =$ CHECKSTOP(V, T)
  end while
  $\overset{*}{\tau} = \arg\min_{\tau \in T} p(\tau|\Lambda)$
  Return $(V, \overset{*}{\tau}, \Gamma, \Lambda, T)$
end function
function CHECKSTOP(V, T)
  Return $(|V| \geq 2$ and $|T| \geq 1)$ or $|V| \geq 7)$
end function

A. Representations

We start by describing the underlying representations used to define our approach. Let $\Xi = [\xi_1, \xi_2, \ldots, \xi_N]$ be a list of possible camera poses, where $\xi_i \in SE(3)$, and $V \subset \Xi$ is the set of already visited camera poses. This list must be finite, and should provide good coverage of the workspace. In addition, let γ be a point cloud obtained from a certain camera pose ξ. We define Γ_t as the combined object point cloud, segmented from the table plane, after t views have been taken,

$$\Gamma_t = \Gamma_{t-1} \uplus segmented(\gamma), \tag{1}$$

i.e., Γ_t is the result of segmenting the object point cloud from the table plane in γ and integrating this result with our previous obtained object point Γ_{t-1}.

In addition to the object point cloud, we also maintain a representation of the full robot workspace as a 3D occupancy grid, implemented with an octree. We shall refer to this 3D occupancy grid as Λ, which is updated after each view and observation (ξ, γ). The implementation we use [9] allows us to easily represent known and unknown parts of the robot workspace Λ and thus to define the information gain and safety based view planning strategies.

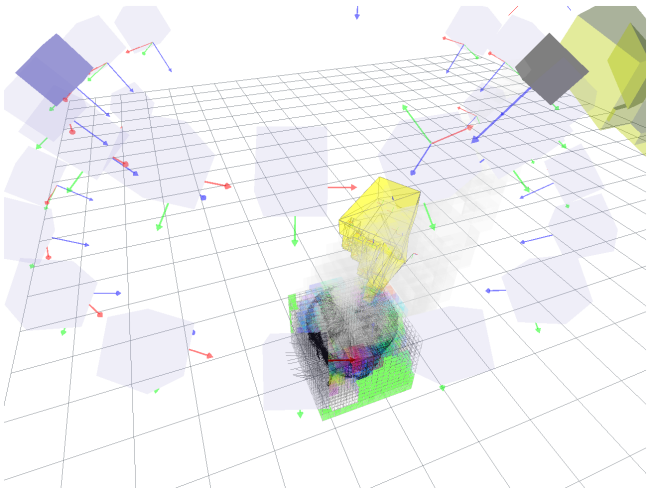

Fig. 3. View camera poses forming the set Ξ, camera pose highlighted in darker purple belongs to the set of visited poses V. The object in the centre is circumscribed by a voxelised cube. Information gain view exploration sculpts this cube when no contacts are found.

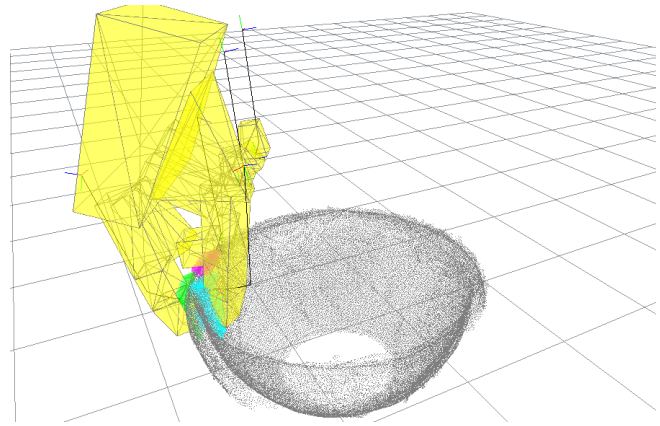

Fig. 4. (Left) An example grasp, showing the contact regions for different finger links in different colours. Contact driven vision attempts to view all these planned contact locations.

It is possible to find a grasp trajectory τ by transferring a learnt grasp G to the given object represented by Γ_t using the method of Kopicki, Wyatt et al. [10]. Using the same method, we are able to extract contact points from Γ_t, yielding a list of contacts $C = [\mathbf{c_1}, \ldots, \mathbf{c_M}]$, where $\mathbf{c_i} = (w_i, \mathbf{p_i}, \mathbf{n_i})$ is composed of a weight $w_i \in \mathcal{R}$, indicating its relevance to the grasp, the contact location $\mathbf{p_i} \in \mathcal{R}^3$, and the surface normal at that point $\mathbf{n_i} \in \mathcal{R}^3$. Points are relevant to a grasp if they are close to planned contact point for that grasp. The weight w_i falls off exponentially as the distance from the planned finger position increases. Let us also define $\Omega = [(\mathbf{c_1}, z_1), \ldots, (\mathbf{c_M}, z_M)]$ as a list of contact points expanded to include the current quality z_i of the observation of each point from the best view ξ to date. Contact driven vision prioritizes looking at the planned contact points for which there is currently low quality reconstruction, rather than elsewhere on the object. We now describe contact-based view selection in detail.

B. Contact Based View Selection

We let the viewing direction of a certain view pose $\xi_k \in \Xi$ be the vector $\mathbf{v_k}$, which we always constrain to point towards the object Γ_t. We define the quality of observation of a contact point $\mathbf{c_i}$ from a given ξ_k as

$$\theta_{ki} = \theta(\xi_k, c_i) = \arccos(min(0, \mathbf{v_k}^T \mathbf{n_i})). \tag{2}$$

This models the fact that the depth errors rise as the surface becomes perpendicular to the image plane. Thus when looking at contact point with surface normal $\mathbf{n_i}$, we assign higher values to views that are square on, where the image plane normal and the surface normal directly oppose one another, i.e. $\mathbf{n_i}$ and $\mathbf{v_k}$ form an angle of 180 degrees, according to our convention that $\mathbf{v_k}$ always looks towards the object.

Thus, for each element $(\mathbf{c_i}, z_i) \in \Omega$ we store the contact points $\mathbf{c_i}$, and define z_i the best quality of observation to date with respect to all visited poses as

$$z_i = \arg\max_{\xi_j \in V} \theta_{ji}. \tag{3}$$

Finally, let $F_\tau = [\mathbf{f_1}, \ldots, \mathbf{f_R}]$ be the list of finger link normals for the finger surfaces that involved in the grasp. These are calculated for the last time step of the trajectory τ. We then define the value of a particular (untried) view with respect to a particular contact $\mathbf{c_i}$ as

$$\sigma(\xi_k, F_\tau, \mathbf{c_i}) = w_i \sum_{r=1}^{R} max(\theta_{ki}, z_i) \frac{1 - sign(\mathbf{f_r}^T \mathbf{n_i})}{2}. \tag{4}$$

This defines high value views as being those views which gaze head on at contact points. Note that when looking at a certain contact point $\mathbf{c_i}$ we are able either to improve our previous best viewing quality if $\theta_{ki} > z_i$, or leave it as it is. Note also that the multiplying term $\frac{(1-sign(\mathbf{f_r}^T \mathbf{n_i})}{2}$ serves as a switch that yields 0 or 1. This simply ensures that the view must be of the side of the point cloud where the finger will contact. In other words it models the geometric constraint that a link must have with a contact point, i.e. the surface normal $\mathbf{f_r}$ of a given finger link must point in the opposite direction of the contact normal $\mathbf{n_i}$, otherwise this contact point is meaningless with respect to this given finger link, meaning that viewing it is not useful. Finally, the normalised weight w_i scales this value according to its overall relevance to the grasp as defined by the appraoch of Kopicki, Wyatt et al. [10]. It follows that the total utility of a given view ξ is given by

$$u_1(\xi, \Omega, \tau) = \sum_{i=1}^{N} \sigma(\xi, F_\tau, \mathbf{c_i}). \tag{5}$$

We are then able to rank the potential views by calculating the total value of a view with respect to all contact points, and picking the view that has the maximum value according to Eq 6.

Algorithm 2 Select Next Best View Contact Based

function NBVCONTACTBASED($\Xi, \Gamma, \Lambda, V, \Omega, \tau$)
  $\overset{*}{\xi} = None$
  if $|V| = 0$ **then**
    $\overset{*}{\xi} = head(\Xi)$
  else if $\Omega \neq \emptyset$ **then**
    Let $\overset{*}{\xi}$ be selected according to Eq 6
  else
    Let $\overset{*}{\xi}$ be selected according to Eq 14
  end if
  Return $\overset{*}{\xi}$
end function

$$\overset{*}{\xi} = \arg \max_{\xi_k \in \Xi - V} u_1(\xi, \Omega, \tau). \tag{6}$$

C. Information Gain View Selection

Of course, if no grasp can be found, then grasp driven view selection cannot run. In this case, the robot should look at the workspace around the recovered point cloud. To support this we define an information-gain based utility function for view selection. Intuitively, this strategy makes sense, since no contacts were found with the knowledge we have about the object shape so far, represented by Γ_t. Therefore, one should ideally adopt an exploratory behaviour to seek for new parts of the object.

For this purpose, let $bmin(\Gamma_t), bmax(\Gamma_t) \in \mathcal{R}^3$ be the respective minimum and maximum limits of the bounding box that circumscribes the object point cloud Γ_t. We are then able to extract the set of voxels $\Lambda_{object} \subset \Lambda$ inside this bounding box. If we assume the surface of this voxelised solid box Λ_{object} is visible from all cameras, as shown in Fig 3. Then we can define a simple strategy to minimise the entropy about the object's shape, by selecting views that are going to have maximum predicted information gain about the voxels in Λ_{object}. Intuitively, our goal is to select views such that we gradually sculpt the solid cube, such that we will eventually reach a constant entropy value for this cube, due to self-occluding parts of the object, from which point no views are going to bring any more information gain.

Our first step to fulfil this goal is to define a rule with which we can determine the set of visible voxels in Λ_{object} visible from a camera pose ξ. The visibility test is performed using a typical frustum culling graphics procedure, with a few slight modifications. First, we transform the set of voxels Λ_{object} into the camera coordinate system. During the projection phase of the pipeline, we allow many free voxels along the line of sight to be projected onto identical image coordinates, but we do not allow either unknown voxels, nor occupied voxels to be projected on top of one another. As a consequence, we find a border in our initial solid cube $\Lambda_{visible}(\xi) \subset \Lambda_{object}$, which contains all free voxels visible on the image plane, together with boundary voxels that might be either unknown or occupied, as shown in Fig 5. Thus,

Algorithm 3 Safety Exploration

function SAFETYEXPLORATION($\Xi, \Gamma, \Lambda, \tau, V$)
  $stop = false$
  while not $stop$ **do**
    $\overset{*}{\xi}, value = safetyNBV(\Xi, \Lambda, \tau)$
    $V = append(T, \overset{*}{\xi})$
    $\gamma = capture(\overset{*}{\xi})$ ▷ Point cloud from pose $\overset{*}{\xi}$.
    $\Gamma = \Gamma \uplus segmented(\gamma)$
    $\Lambda = updateOcTree(\Lambda, \gamma, \overset{*}{\xi}, \Omega)$
    $T = append(T, \tau)$
    $stop =$ CHECKSTOPSAFETY($value$)
  end while
  $p = p(\tau|\Lambda)$
  Return (V, p, Γ, Λ)
end function
function CHECKSTOPSAFETY($value$)
  if $value \leq \beta$ **then**
    Return $true$
  else
    Return $false$
  end if
end function

Algorithm 4 Safety Exploration View Selection

function SAFETYNBV(Ξ, Λ, τ)
  $\Lambda_c = findPassingVoxels(\Lambda, \tau)$ ▷ Finding voxels through which the hand is passing
  Using Λ_c, let $\overset{*}{\xi}$ be selected according to Eq 14
  $value = u_2(\overset{*}{\xi}, \Lambda_c)$
  Return $(\overset{*}{\xi}, value)$
end function

$\Lambda_{visible}(\xi) = \{s_1, \ldots, s_D\}$ is defined as our set of voxels of interest for information gain prediction. We then follow to describe the information gain prediction for the set of voxels $\Lambda_{visible}(\xi)$.

1) Information Gain Prediction: Let the occupancy probability of a voxel $s_d \in \Lambda_{visible}$ up to our most recent observations $o_{1:t}$ be $p_{s_d} = p(s_d|o_{1:t})$. We can write the entropy of this voxels by viewing it as a Bernoulli random variable with entropy

$$H(s_d) = -p_{s_d} \log(p_{s_d}) - (1 - p_{s_d}) \log(1 - p_{s_d}), \tag{7}$$

By using a log-odds representation of our occupancy probability such as in [9], [13], we can then define future predicted occupancy probability of s_d as

$$L(s_d|o_{1:t}, o'_{t+1}) = L(s_d|o_{1:t}) + L(s_d|o'_{t+1}), \tag{8}$$

where $o'_{t+1} \in O = \{occupied, free\}$ is an imaginary future measurement and $L(s_d|o)$ is also called *inverse sensor model* [17]. The inverse sensor model is defined likewise as in [9] as

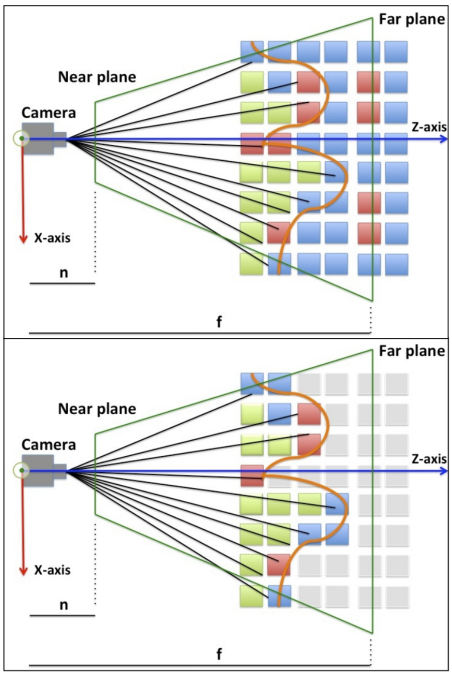

Fig. 5. Cross-section view of a typical visibility check. In the picture, occupied voxels are represented in red, free voxels are green and unknown voxels have dark-blue colour. Having defined a viewing frustum to match the real depth camera specifications, frustum culling procedure is performed in which free voxels are assumed to be transparent, whereas unknown or occupied voxels occlude each other. As a result, only the voxels coloured on the bottom image are defined as being visible after the execution of this procedure.

$$L(s_d|o) = \begin{cases} L_{occ}, & \text{if } o = occupied. \\ L_{miss}, & \text{otherwise.} \end{cases} \tag{9}$$

Note that our occupancy probability converted from log-odds is then

$$p_{sd|o'_t} = p(s_d|o_{1:t}, o'_{t+1}) = 1 - \frac{1}{1 + exp(L(s_d|o_{1:t}, o'_{t+1}))}. \tag{10}$$

We make a simplifying assumption that an imaginary measurement has uniform distribution, i.e. $p(occupied) = p(free) = 0.5$. Thus, we define our predicted entropy resulting from an imaginary measurement as the expected value

$$\begin{aligned} H'(s_d) = -\sum_{o' \in O} p(o')\{p_{sd|o'} \log(p_{sd|o'}) \\ +(1 - p_{sd|o'}) \log(1 - p_{sd|o'})\} \end{aligned} \tag{11}$$

Therefore, the information gain of looking at a particular voxel $s_d \in \Lambda_{visible}(\xi)$ from a given view ξ is given by

$$I(\xi, s_d) = H(s_d) - H'(s_d), \tag{12}$$

where the average information gain per voxel is given by

$$u_2(\xi, \Lambda_{visible}(\xi), \Gamma_t) = \sum_{s_d \in \Lambda_{visible}} \frac{I(\xi, s_d)}{D}, \tag{13}$$

Algorithm 5 Grasp Driven Active Sense

procedure ACTIVEGRASP(Ξ, G)
  $\Gamma = \emptyset, \Lambda = \emptyset$
  $V = \emptyset, T = \emptyset$ ▷ List of visited views and found trajectories, respectively. Initially empty
  $grasp = false$
  $\overset{*}{\tau} = None$
  while not $grasp$ **do**
    $V, \overset{*}{\tau}, \Gamma, \Lambda = nextBestView(\Xi, \Gamma, \Lambda, G, V, T)$
    $V, p, \Gamma, \Lambda = safetyExploration(\Xi, \Gamma, \Lambda, \overset{*}{\tau}, V)$
    if $p \leq \alpha$ **then**
      $grasp = true$
    else
      $T = T - \{\overset{*}{\tau}\}$ ▷ Removing $\overset{*}{\tau}$ from our candidate trajectories
    end if
  end while
  $executeGrasp(\overset{*}{\tau})$
end procedure

where $D = |\Lambda_{visible}|$ is the number of visible voxels. Note that we refer to the average information gain per voxel, since different views have different numbers of visible voxels in their field of view after frustum culling. This makes the predicted information different gain for different views comparable.

2) Information Gain View Selection: Using the definitions aforementioned, when no contacts are available, we are finally able to select next best views according to a maximum information gain strategy via

$$\overset{*}{\xi} = \arg \max_{\xi_k \in \Xi - V} u_2(\xi_k, \Lambda_{visible}(\xi), \Gamma_t). \tag{14}$$

D. Safety View Planning

In safety view planning we are interested in estimating the probability of collision during the execution of a given trajectory τ, disregarding the collision with the final contact points Ω. Effectively, we estimate the probability of an unexpected collision along the trajectory τ. This is a typical scenario in which the robot hand collides with an unknown part of the object due to the fact that the grasp was originally planned from an incomplete model of the object's shape Γ_t. In addition, we are also able to access how certain we are regarding this collision estimation by computing the current entropy for this particular trajectory τ. As such, we select views as to minimise the entropy of the voxels through which the robot hand is going to pass when following a given grasp trajectory τ. This enables us to have a final relatively certain estimation with respect to unexpected collisions that might damage the robot hand, or simply make the grasp fail.

Let the set of voxels through which the hand bounds pass when following a trajectory τ be Λ_c. These voxels are retrieved by simulating the hand moving along the trajectory τ and querying at each time step of this trajectory the voxels the hand is passing through in our voxelised workspace Λ.

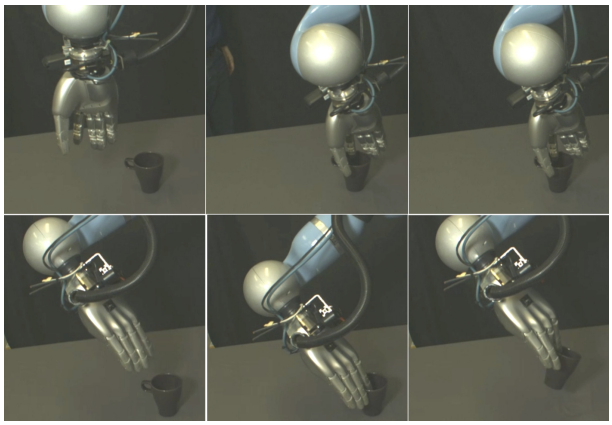

Fig. 6. Top three pictures show a failed grasp due to unexpected collision with parts of the object that are not involved in the grasp. Bottom three pictures show a successful and safe grasp selected by our approach.

Fig. 7. The 14 objects used for trials.

Having retrieved those voxels, let p_{s_c} be the probability of occupancy of a given voxel $s_c \in \Lambda_c$. The probability of collision can be calculated by

$$p_{collision}(\tau, \Lambda_c) = 1 - \prod_{s_c \in \Lambda_c} (1 - p_{s_c}). \tag{15}$$

For numerical reasons, we prefer to refer to Eq 15 using only the product term, representing the probability that all voxels along the trajectory τ are free, in its logarithmic form as

$$\kappa(\tau, \Lambda_c) = \ln \prod_{s_c \in \Lambda_c} (1 - p_{s_c}) = \sum_{s_c \in \Lambda_c} \ln(1 - p_{s_c}), \tag{16}$$

note that $p_{collision}(\tau, \Lambda_c) = 1 - exp(\kappa(\tau, \Lambda_c))$.

Finally, to select views in order to get better estimations for Eq 15, we use the same utility function defined in 13. Thus if we let $\Lambda_{c_{visible}}(\xi) \subset \Lambda_c$ be the set of visible voxels by a certain view pose ξ. Next best views are then selected according to

$$\overset{*}{\xi} = \arg \max_{\xi_k \in \Xi - V} u_2(\xi_k, \Lambda_{c_{visible}}(\xi), \Gamma_t). \tag{17}$$

In practice, we allow safety exploration to run while the information gain is above a predefined threshold, i.e. $u_2(\xi_k, \Lambda_{c_{visible}}(\xi), \Gamma_t) > \beta$. If this criteria is not met, the final probability of collision is reported according to Eq 15. The trajectory τ is therefore executed or not based on the probability of collision.

IV. Experiments

The following section outlines the experiments we conducted to test our view selection approach. General pseudo-code of our implementation is described in Alg 5, which is divided into two sub-phases. First a contact-based next best view exploration procedure is run as outlined by Alg 1. In this first phase, at least two views are selected, up to a maximum of 7 views if after the second view no grasp trajectory and contacts are found. The first view is fixed, only subsequent views after this fixed view are selected according to the criteria for contact-based view selection. The second phase of Alg 5 is run in order to estimate the probability of collision of the best promising candidate trajectory $\overset{*}{\tau}$, selected as the trajectory with the lowest probability of collision prior to the safety view exploration phase, given our current knowledge of the object Γ_t and workspace Λ. This second phase is outlined in Alg 3. Note that the safety exploration phase stops if the current selected view predicts information gain below a certain threshold β. If, at the end of the safety exploration phase, we discover that this trajectory $\overset{*}{\tau}$ has collision probability above a certain threshold α, we reject $\overset{*}{\tau}$ and cycle back to phase 1, i.e. Alg 1.

A. Methodology

Using Alg 5 we performed trials on a set of 14 novel objects shown in Fig 7. In our experiments, we compared our active view algorithm with a random view selection strategy. In other words, we substituted all calls of the selection procedures Alg 2 and Alg 4 by a uniform random view selection scheme. Furthermore we limited the two phases of this modified random-based approach to be constrained to the same number of views that our algorithm performed in both phases. It is also important to note that in our experiments we have set the size of the voxels in our 3D occupancy map Λ to be $0.0025m$, for relatively fine precision. Table I shows the final data for this experiment.

B. Results

The results shown in Table I outline the contrast between the two approaches. We first note that the success rate of our proposed view selection approach achieved a success rate of 85.7%, whereas the modified random-based approach showed a success rate of only 64.2%. A closer look at Table I reveals that random exploration tended to yield unsafe grasps, under the same view number constraints as our active view selection approach. This indicates that random view selection would probably need to cycle back to generate new grasp trajectory candidates more times, which seems a natural consequence of its sub-optimal exploratory behaviour.

TABLE I

TRIAL RESULTS

	Phase 1 View Count	Phase 2 View Count	Grasp Results		Collision Probabiltiy		Grasp Views
Objects	NBV & Random	NBV & Random	NBV	Random	NBV	Random	t of Γ_t
bowl	4	4	success	fail	0.005009	1.0	4
bowl small	3	4	success	success	0.001044	1.0	3
bucket	5	4	success	success	0.000035	0.007403	8
coke	2	5	success	success	0.002384	1.0	2
cup1	3	5	success	fail	0.001015	1.0	7
dustpan	3	3	success	success	0.002267	1.0	3
glass2	4	4	fail	fail	1.0	1.0	4
guttering	3	4	success	success	0.001526	0.0050009	3
jug	3	5	success	success	0.000507	1.0	7
mrmuscle	3	4	success	success	0.003768	1.0	6
mug1	3	4	success	fail	0.0009	1.0	6
rennie	3	3	success	success	0.00313	1.0	3
stand2	3	3	success	fail	0.006257	1.0	3
toothpaste	4	4	fail	success	0.00491	0.003807	7
		Success Rate	0.857	0.642		Average Views	4.7

One such example is highlighted by Fig 6, in which the final trajectory executed with probability of collision 1.0 and, indeed, makes the robot hand collide with a part of the mug not involved in the grasp, finally failing for safety reasons. We also note that our collision probability appears to be over-sensitive, the random approach also succeeded for various cases in which the probability of collision was 1.0. Nonetheless, even for successful grasps as the one depicted in Fig 2, grasps with probability 1.0 tended to collide prematurely with different parts of the object. In addition, we also noted that for the case of the toothpaste, the trivial solution of a grasp with as few collisions as possible might yield grasps with very poor grip. This indicates future work towards a middle ground between these two extremes.

Finally, as shown by Table I and in Fig 8, our approach had equivalent success rate to prior work done by by Kopicki and Wyatt et al [10]. In our experiments our approach used on average 4.7 views for grasp planning, as compared with 7 in [10]. Additional views were only used to assess safety.

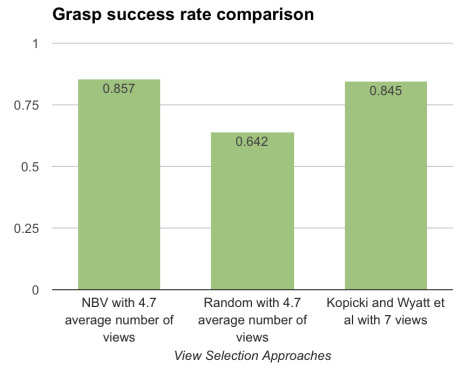

Fig. 8. Grasp success rate comparison.

V. CONCLUSIONS

We have proposed an effective approach for view selection comprising two stages. The first stage guides gaze by planned contacts, seeking a low noise point cloud near those contacts. If no contact points are available, we guide gaze so as to minimise the entropy of the 3D occupancy grid map around our object cloud Γ_t. After candidate grasps are found, we gaze to assess the safety of the reach-to-grasp trajectory candidate. We showed that this yields a better success rate compared to a random strategy, and that we slightly improve on [10], while using fewer views for grasp planning.

REFERENCES

[1] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. Active vision. *International Journal of Computer Vision*, 1(4):333–356, 1988.

[2] Ruzena Bajcsy. Active perception. In *Proc IEEE, 76:996–1005*, 1988.

[3] H. Ben Amor, O. Kroemer, U. Hillenbrand, G. Neumann, and J. Peters. Generalization of human grasping for multi-fingered robot hands. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[4] Jeannette Bohg, Matthew Johnson-Roberson, Beatriz Len, Javier Felip, Xavi Gratal, N Bergstrom, Danica Kragic, and Antonio Morales. Mind the gap-robotic grasping under incomplete observation. In *IEEE International Conference on Robotics and Automation*, pages 686–693. IEEE, 2011.

[5] Shengyong Chen, Youfu Li, and Ngai Ming Kwok. Active vision in robotic systems: A survey of recent developments. *I. J. Robotic Res.*, 30:1343–1377, 2011.

[6] Noel Curtis and Jing Xiao. Efficient and effective grasping of novel objects through learning and adapting a knowledge base. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2252–2257. IEEE, 2008.

[7] Renaud Detry and Justus Piater. Unsupervised learning of predictive parts for cross-object grasp transfer. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013.

[8] Martin Hjelm, Renaud Detry, Carl Henrik Ek, and Danica Kragic. Representations for cross-task, cross-object grasp transfer. In *IEEE International Conference on Robotics and Automation*, 2014.

[9] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: an efficient probabilistic 3d mapping framework based on octrees. *Autonomous Robots*, 34(3):189–206, 2013.

[10] Marek Kopicki, Renaud Detry, Maxime Adjigble, Rustam Stolkin, Ales Leonardis, and Jeremy L. Wyatt. One-shot learning and generation of dexterous grasps for novel objects. *The International Journal of Robotics Research*, 2015. first published on September 18, 2015.

[11] Marek Kopicki, Renaud Detry, Florian Schmidt, Christoph Borst, Rustam Stolkin, and Jeremy L. Wyatt. Learning dextrous grasps that generalise to novel objects by combining hand and contact models. In *IEEE International Conference on Robotics and Automation*, pages 5358–5365. IEEE, 2014.

[12] Michael Krainin, Brian Curless, and Dieter Fox. Autonomous generation of complete 3d object models using next best view manipulation planning. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 5031–5037. IEEE, 2011.

[13] H. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2, pages 116–121, Mar 1985.

[14] Alexander Rietzler, Renaud Detry, Marek Kopicki, Jeremy L. Wyatt, and Justus Piater. Inertially-safe grasping of novel objects. In *Cognitive Robotics Systems: Replicating Human Actions and Activities (Workshop at IROS 2013)*, 2013.

[15] A. Saxena, L. Wong, and A.Y. Ng. Learning grasp strategies with partial shape information. In *Proceedings of AAAI*, pages 1491–1494. AAAI, 2008.

[16] Michael Stark, Philipp Lies, Michael Zillich, Jeremy Wyatt, and Bernt Schiele. Functional object class detection based on learned affordance cues. In *Computer Vision Systems*, pages 435–444. Springer, 2008.

[17] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.