Sensor Planning for 3D Object Search

by

Yiming Ye and **John K. Tsotsos**

Department of Computer Science

University of Toronto

Toronto Ontario

Canada M5S 1A4

yiming@vis.toronto.edu tsotsos@vis.toronto.edu

Tel: (416) 978-6986 (416) 978-3619

FAX: 416-978-1455

Abstract

The task of sensor planning for object search is formulated and a strategy for this task is proposed. The searcher is assumed to be a mobile platform equipped with an active camera and a method of calculating depth, like a stereo or laser range finder. The formulation casts sensor planning as an optimization problem: the goal is to maximize the probability of detecting the target with minimum cost. The search region is thus characterized by the probability distribution of the presence of the target. The control of the sensing parameters depends on the current state of the search region and the detecting abilities of the recognition algorithm. In order to efficiently determine the sensing actions over time, the huge space of possible actions is reduced to a finite set of actions that must be considered. The result of each sensing operation is used to update the status of the search space.

List of Symbols

a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z

A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z

0,1,2,3,4,5,6,7,8,9,0

Ω, $\bigcup$, $\neq$, $\sum$, α, β, $\neg$, $\ldots$, $\prod$, $\bigcap$, $\bigcap$, $\times$, Δ, (,), [,], {, }, θ, δ, $>$, $<$, $\hat{}$, ϕ, σ, π

Contents

1 Introduction

The research described in this paper conforms to the task of efficiently searching for a given 3D object in a given 3D environment by a mobile platform equipped with a camera and, if the environment configuration is not known, a method of calculating depth, like a stereo or laser range finder. It is clear that exhaustive, brute-force blind search will suffice for its solution; however, our goal is the design of efficient strategies for search because exhaustive search is computationally and mechanically prohibitive for non-trivial situations. In general, this task consists of three subtasks. The first subtask is the selection of the sensing parameters so as to bring the target into the field of view of the sensor with sufficient quality of the image that it can be detected by the recognition algorithms. This is called the sensor planning problem for object search, and it is the main concern of this paper. The second subtask is the manipulation of the hardware so that the sensing operators can reach the state specified by the planner. The recent availability of sophisticated hardware for sensor/platform control and real-time image processing ([43],[4], [49]) makes this task feasible. The third subtask involves searching for the target within the image. This is the object recognition and localization problem, which attracts a lot of attention within the computer vision community (see [13] for a review).

Sensor planning for object search is very important if a robot is to interact intelligently and effectively with its environment. For example, a robot often needs to use sensory information to determine the position of the objects in its workspace in order to manipulate them. It is generally impractical to determine the location of the object before it is needed. Because of the camera's limited field of view, occlusion, the limited depth of field of lenses, the lighting conditions and other factors, it is often necessary to obtain many images using many camera configurations and viewpoints (see [49] for further discussion). Also, a robot navigating in an industrial environment often needs to search the environment for known landmarks in order to estimate its own position. The above two examples illustrate the critical requirement for efficient directing of the sensor. Although sensor planning for object search is very important in practice, it is interesting to note that there is little research on this subject within the computer vision community ([43], [51], [18], [32], [37]). Garvey [18] proposes the idea of indirect search for the target: first the sensor is directed to search for an "intermediate" object that commonly participates in a spatial relationship with the target, and then the sensor is directed to examine the restricted region specified by this relationship. Wixson and Ballard [50] [51] present a mathematical model of search efficiency and analyze the efficiency of indirect search over a wide range of situations that vary the spatial structure of the domain as well as recognition performances. They conclude that indirect search can improve efficiency in many situations. The drawbacks of indirect search are that the spatial relationship may not always exist and detection of the "intermediate" object may not always be easier than detection of the target. Connell et al. [14] constructs a robot that roams an area searching for and collecting soda cans. The robot is designed to follow the walls of the room and the sensor only searches the area immediately in front of the robot. This may not be very efficient since the likelihood of the presence of the target is not considered. It is interesting to note that the operational research community has extensively studied the problem of optimal search [27]. Their approach, however, is too general and thus it is inappropriate to directly apply their results here.

Object search is a task that is particularly suited to the active vision approach. Thus the study of object search may produce results that will lead to an understanding of various aspects of an active vision system, especially the role of a sensor planning component in a task oriented system and its relationship to the vision component. The active vision approach argues that rather than simply analyze a set of pre-recorded images, the observer should actively control its image acquisition process with the purpose that the acquired images be at least relevant or useful to the task at hand. From the aspect of efficiency of the action selection process, there are two extremes in the calculation spectrum. At one end is Brooks's theory on reactive planning, in which no time is spent on calculating the action sequence. At the other end is the strategy of reconstructing every detail of the environment and then selecting an action sequence that is best according to given criteria. The goal of the sensor planning system is to find a balance between these opposite ends such that the task can be best performed within given limits on effort. In the relation between sensor planning and image analysis, previous research has generally placed much more emphasis on image analysis than on sensor planning. The sensor planning problem, however, is very important, since it is the sensor's state parameters for the most part that determine the quality of the resulting image. For some task oriented vision systems, e.g., object search, sensor planning is definitely required. Recently, more and more work on sensor planning for computer vision has appeared. For example, Krotkov[28][29] considers the problem of how to determine the focus motor position providing the sharpest focus on an object point at an unknown distance and how to compute the distance to a sharply focused object point. Cowan et al. [15] and Tarabanis et al. [40] present approaches to automatically generating the possible camera locations, orientations, and optical settings for observing an object. Their approaches, which use models of the object and the camera, is based on meeting the requirements that the spatial resolution be above a minimum value, all surface points be in focus, all surfaces lie within the sensor field of view, and no surface points be occluded. There is also work on sensor planning with respect to task oriented systems. For example, Rimey and Brown [37] use composite Bayes net and utility decision rules to plan the sensor in their task-oriented system TEA. The goal of their system is to answer qualitative questions based on the scene of a dinner table, such as whether it is an informal or fancy meal. In order to verify the presence and position of an object on the table, the sensor is directed to the center of mass of the expected area for the object based on the belief value on the net. Grimson [46] proposes an optimal sensing strategy for disambiguating among alternative interpretations of scenes containing multiple polyhedral objects (multiple objects in known poses or a single object with several consistent poses). His goal is to use a minimum amount of sensory data in the interaction of sensing and action required for a robot to recognize and locate objects in its workspace.

This paper provides the first formalization of the sensor planning task for object search and proposes a practical sensor planning strategy. Instead of examining different aspects of object search individually and in some degree of isolation, the formulation emphasizes more the relationship between action and image processing and integrates the two into a complete system. The sensor planning strategy decomposes the huge space of possible sensing actions into a finite set of actions that must be considered, thus greatly simplifying the sensor planning task. Both simulation experiments and real experiments are presented in support of the concept.

2 Problem Formulation

At first glance object search seems easy —– turn the camera around and check every part of the environment for the target. Close examination reveals several difficulties. First, the search agent usually has some prior knowledge of the location of the target, this knowledge should be used to guide the search process. Second, there is no perfect recognizer. Even if the target is within the field of view of the camera, the recognition algorithm might fail to detect it. Third, occlusion might hinder the search process. These factors need to be incorporated in any formulation of the object search process.

In the following, we first explain the environment, the model of the search agent, and some basic concepts used in the discussion. Then we formulate the task of sensor planning for object search.

The search region Ω can be of any form, and it is assumed that we know the boundary of Ω exactly, although we do not know its internal configuration. In practice, we tessellate the region Ω into a series of elements c_i, $\Omega = \bigcup_{i=1}^{n} c_i$ and $c_i \bigcap c_j = 0$ for $i \neq j$. In the rest of the paper, we assume the search region is an office-like environment, and we tessellate the space into little cubes of equal size.

The model of the search agent is based on the ARK robot (Figure 1(b)) [35], which is a mobile platform equipped with a special sensor: the Laser Eye [22] (Figure 1(a)). The Laser Eye is mounted on a robotic head with pan and tilt capabilities. It consists of a camera with controllable focal length (zoom), a laser range-finder and a mirror. The mirror is used to ensure collinearity of effective optical axes of the camera lens and the range finder. With this mirror, the laser range finder can measure the distance from the center of the camera

(a) (b)

Figure 1:

(a) The Laser Eye. At the top is the Optech laser range finder, at the bottom is the zoom and focus controlled lens. The two mirrors are used to ensure collinearity of effective optical axes of the camera lens and the range finder. The pan-tilt unit is operated by two DC servocontrolled motors from Micromo Electronics, equipped with gearboxes and optical encoders.
(b) The ARK mobile platform, which is a 3 wheeled Cybermotion K2A platform equipped with a small number of sonar transducers.

to the object along the camera viewing axis. The camera's image plane is assumed to be always coincident with its focal plane. There are three coordinate systems used to describe the geometric configuration of the search agent: the world coordinate system (WCS), the head coordinate system (HCS) and the camera coordinate system (CCS). The WCS is fixed and corresponds to the search region Ω. The HCS is attached to the robot head with its origin at the center of the robot head. The direction of the HCS is the same as that of the WCS. The CCS corresponds to the camera and has its origin at the nodal point of the camera. Here we assume the origin of the CCS and the origin of the HCS are the same. The direction of the Z axis of the CCS is the same as the direction of the view axis of the camera. The direction of the X axis of the CCS is parallel to the horizontal direction of the image plane. The direction of the Y axis of the CCS is parallel to the vertical direction of the image plane. The direction of the camera viewing axis Z can be specified by the pan and tilt (p, t) with respect to the HCS. The pan p is the amount of counterclockwise rotation needed to bring the viewing axis of the camera into the XZ plane of the HCS; tilt t is the angle between the viewing axis of the camera and the Z axis of the HCS. The range for p and t are $0 \leq p < 2\pi$ and $0 \leq t < \pi$. Thus, the state of the search agent is uniquely determined by 7 parameters $(x_c, y_c, z_c, w, h, p, t)$, where (x_c, y_c, z_c) is the position of the camera center in WCS, w, h are the width and height of the solid viewing angle of the camera. The position (x_c, y_c, z_c) can be adjusted by moving the mobile platform. The viewing angle size $< w, h >$ can be adjusted by the zoom lens of the camera. Pan and tilt (p, t) can be adjusted by the motors on the robotics head.

An operation $\mathbf{f} = \mathbf{f}(x_c, y_c, z_c, p, t, w, h, a)$ is an action of the searcher within the region Ω, where a is the recognition algorithm used to detect the target. An operation $\mathbf{f}$ entails two steps: (1) take a **perspective** projection image according to the camera configuration of $\mathbf{f}$, and then (2) search the image using the recognition algorithm a.

The target distribution can be specified by a probability distribution function $\mathbf{p}$. The term $\mathbf{p}(c_i, t_v)$ gives the probability that the center of the target is within cube c_i at time t_v. Usually this distribution is assumed to be known at the beginning of the search process, and it is determined by our knowledge of the world. If we know nothing about the target probability distribution, then we can assume a uniform distribution at the beginning. Note, we use $\mathbf{p}(c_o, t_v)$ to represent the probability that the target is outside the search region at time t.

The detection function on Ω is a function $\mathbf{b}$, such that $\mathbf{b}(c_i, \mathbf{f})$ gives the conditional probability of detecting the target given that the center of the target is located within c_i, and the operation is $\mathbf{f}$. This function is further simplified as the conditional probability of detecting the target when the target center is located at the center of c_i. For any operation, if the projection of the center of the cube c_i is outside the image, we assume $\mathbf{b}(c_i, \mathbf{f}) = 0$; if the cube is occluded or too far from the camera or too near, we also have $\mathbf{b}(c_i, \mathbf{f}) = 0$. In general [53] [52], $\mathbf{b}(c_i, \mathbf{f})$ is determined by various factors, such as intensity, occlusion, and orientation, etc. It is obvious that the probability of detecting the target by applying action $\mathbf{f}$ is given by

$$P(\mathbf{f}) = \sum_{i=1}^{n} \mathbf{p}(c_i, t_{\mathbf{f}})\mathbf{b}(c_i, \mathbf{f}) \tag{1}$$

where $t_{\mathbf{f}}$ is the time just before $\mathbf{f}$ is applied. Let Ψ be the set of all the cubes that are within the field of view of $\mathbf{f}$ and that are not occluded, then we have

$$P(\mathbf{f}) = \sum_{c \in \Psi} \mathbf{p}(c, t_{\mathbf{f}})\mathbf{b}(c, \mathbf{f}) \tag{2}$$

The reason that the term $t_{\mathbf{f}}$ is introduced in the calculation of $P(\mathbf{f})$ is that the probability distribution needs to be updated whenever an action fails. Here we use Bayes' formula. Let α_i be the event that the center of the target is in cube c_i, and α_o be the event that the center of the target is outside the search region. Let β be the event that after applying a recognition action, the recognizer successfully detects the target. Then $P(\neg\beta \mid \alpha_i) = 1 - \mathbf{b}(c_i, \mathbf{f})$ and $P(\alpha_i \mid \neg\beta) = \mathbf{p}(c_i, t_{\mathbf{f}+})$, where $t_{\mathbf{f}+}$ is the time after $\mathbf{f}$ is applied. Since the above events $\alpha_1, \ldots, \alpha_n, \alpha_o$ are mutually complementary and exclusive, we get the following update rule

$$\mathbf{p}(c_i, t_{\mathbf{f}+}) \leftarrow \frac{\mathbf{p}(c_i, t_{\mathbf{f}})(1 - \mathbf{b}(c_i, \mathbf{f}))}{\mathbf{p}(c_o, t_{\mathbf{f}}) + \sum_{j=1}^{n} \mathbf{p}(c_j, t_{\mathbf{f}})(1 - \mathbf{b}(c_j, \mathbf{f}))} \tag{3}$$

where $i = 1, \ldots, n, o$.

The cost $\mathbf{t_o}(\mathbf{f})$ gives the total time needed to (1)manipulate the hardware to the status specified by $\mathbf{f}$; (2)take a picture; (3)update the environment and register the space; and (4) run the recognition algorithm. We assume that: (2) and (3) are same for all the actions; (4) is known for any recognition algorithm and is constant for a given recognition algorithm.

Let $\mathbf{O}_{\mathbf{\Omega}}$ be the set of all possible operations that can be applied. The effort allocation $\mathbf{F} = \{\mathbf{f}_1, \ldots, \mathbf{f}_k\}$ gives the ordered set of operations applied in the search, where $\mathbf{f}_i \in \mathbf{O}_{\mathbf{\Omega}}$. It is clear that the probability of detecting the target by this allocation is:

$$P[\mathbf{F}] = P(\mathbf{f}_1) + \ldots + \{\prod_{i=1}^{k-1}[1 - P(\mathbf{f}_i)]\}P(\mathbf{f}_k) \tag{4}$$

The total cost for applying this allocation is (following [44]):

$$T[\mathbf{F}] = \sum_{i=1}^{k} \mathbf{t_o}(\mathbf{f}_i) \tag{5}$$

Suppose K is the total time that can be allowed in the search, then the task of sensor planning for object search can be defined as finding an allocation $\mathbf{F} \subset \mathbf{O_\Omega}$, which satisfies $T(\mathbf{F}) \leq K$ and maximizes $P[\mathbf{F}]$.

3 The Complexity of the Sensor Planning Task and a Simplified Version

Complexity level analysis of robotics and vision problems is important because it can reveal basic insights into the structure of the problem and delimit the space of permissible solutions in a formal and theoretical fashion. In this section, we first pinpoint some important properties of the task and then outline the proof that the sensor planning task is NP-complete.

From the definition of the sensor planning task, we can see that the probability of detecting the target by an effort allocation is represented by a complex formula with many intermediate probability distributions $\mathbf{p}(c, t_v)$. Since this distribution is updated every time an action is applied, it is hard to see any regularities in the expression $P[\mathbf{F}]$ defined above. Suppose $\mathbf{p}^{[0]}(c_1), \mathbf{p}^{[0]}(c_2), \ldots, \mathbf{p}^{[0]}(c_n), \mathbf{p}^{[0]}(c_o)$ represent the initial probability distribution, $P^{[0]}(\mathbf{f}_i) = \sum_{j=1}^{n} \mathbf{p}^{[0]}(c_j)\mathbf{b}(c_j, \mathbf{f_i})$ represents the probability of detecting the target by applying the action $\mathbf{f}_i$ when no action has been applied before. If we can represent $P[\mathbf{F}]$ just by the initial probability distributions, we might gain some insight and deeper understanding of the properties of $P[\mathbf{F}]$. The following lemma illustrates this idea.

Lemma 1 *For a given allocation* $\mathbf{F} = \{\mathbf{f}_1, \ldots, \mathbf{f}_q\}$*, we have*

$$P[\mathbf{F}] \;=\; \sum_{i_1=1}^{q} P^{[0]}(\mathbf{f}_i)$$

$$+(-1)^{2+1} \sum_{1 \le i_1 < i_2 \le q} \Big(\sum_{c \in \Omega(\mathbf{f}_{i_1}) \bigcap \Omega(\mathbf{f}_{i_2})} \mathbf{p}^{[0]}(c)\mathbf{b}(c, \mathbf{f}_{i_1})\mathbf{b}(c, \mathbf{f}_{i_2}) \Big)$$

$$\ldots$$

$$+(-1)^{r+1} \sum_{1 \le i_1 < i_2 < \ldots < i_r \le q} \Big(\sum_{c \in \Omega(\mathbf{f}_{i_1}) \bigcap \Omega(\mathbf{f}_{i_2}) \bigcap \ldots \bigcap \Omega(\mathbf{f}_{i_r})} \mathbf{p}^{[0]}(c)\mathbf{b}(c, \mathbf{f}_{i_1})\mathbf{b}(c, \mathbf{f}_{i_2}) \ldots \mathbf{b}(c, \mathbf{f}_{i_r}) \Big)$$

$$\ldots$$

$$+(-1)^{q+1} \Big(\sum_{c \in \Omega(\mathbf{f}_1) \bigcap \Omega(\mathbf{f}_2) \bigcap \ldots \bigcap \Omega(\mathbf{f}_q)} \mathbf{p}^{[0]}(c)\mathbf{b}(c, \mathbf{f}_1) \ldots \mathbf{b}(c, \mathbf{f}_q) \Big) \tag{6}$$

where $\Omega(\mathbf{f}) = \{c \mid \mathbf{b}(c, \mathbf{f}) \neq 0\}$ *is called the influence range of action* $\mathbf{f}$ *(See [55] [52] for proofs).*

From Lemma 1 it is easy to see that $P[\mathbf{F}]$ is not influenced by the order of the applied actions. This result agrees with common knowledge, because for a given set of actions, the expected probability of detecting the target should be same regardless of the order of applications.

As a direct result of Lemma 1, we have the following lemma which can be used to prove that the sensor planning task is NP-complete.

Lemma 2 *For an allocation* $\mathbf{F} = \{\mathbf{f}_1, \ldots, \mathbf{f}_q\}$, *if we restrict the available actions such that no cube belongs to any two influence ranges, i.e.,* $\Omega(\mathbf{f}_i) \bigcap \Omega(\mathbf{f}_j) = \phi$ *for* $i \neq j$, *then* $P[\mathbf{F}]$ *can be calculated by the following:*

$$P[\mathbf{F}] = \sum_{i=1}^{q} P^{[0]}(\mathbf{f}_i) \tag{7}$$

To prove that the sensor planning task for object search is *NP-complete*, we first change the maximization problem into the equivalent decision problem. It is obvious that the resulting problem belongs to NP. After restricting the resulting problem by only allowing instances in which any two available operations do not have common influence range, we are able to prove that the restricted problem is equivalent to the $KNAPSACK$ problem. Thus the sensor planning for object search task is NP-complete. Please refer to [55] [52] for detail.

Since the sensor planning problem is NP-complete and usually the number of the available candidate actions is large, it is necessary to simplify the original problem. Instead of looking for an algorithm that always generates an optimal solution, we simply use heuristics that will generate a feasible solution for the original problem. Here, we use the greedy strategy. The greedy method suggests that one can devise an algorithm which works in stages, considering one input at a time. At each stage, based on some optimization measure, the next candidate is selected and is included into the partial solution so far. Suppose we have already executed q ($q \geq 1$) actions $\mathbf{F}_q = \{\mathbf{f}_1, \ldots, \mathbf{f}_q\}$. We now want to find the next action $\mathbf{f}_{q+1}$ to execute, with the hope that our strategy of finding the next action may lead to an **approximate** solution for the object search task.

When we execute a next action $\mathbf{f}$, the effort allocation becomes $\mathbf{F}_{q+1} = \{\mathbf{f}_1, \ldots, \mathbf{f}_q, \mathbf{f}\}$. The expected probability of detecting the target is $P[\mathbf{F}_{q+1}] = P[\mathbf{F}_q] + \Delta_P(\mathbf{f})$, where $\Delta_P(\mathbf{f}) = \{\prod_{j=1}^{q}[1 - \sum_{i=1}^{n} \mathbf{p}(c_i, t_{\mathbf{f_j}})\mathbf{b}(c_i, \mathbf{f_j})]\} \times [\sum_{i=1}^{n} \mathbf{p}(c_i, t_{\mathbf{f}})\mathbf{b}(c_i, \mathbf{f})]$ (Note: $t_{\mathbf{f_k}}$, $1 \leq k \leq q$ refers to the time just before the application of $\mathbf{f_k}$). The total cost becomes $T[\mathbf{F}_{q+1}] = T[\mathbf{F}_q] + \Delta_T(\mathbf{f})$, where $\Delta_T(\mathbf{f}) = \mathbf{t_o}(\mathbf{f})$. Our strategy for selecting the next action is as follows: the next action $\mathbf{f}_{q+1}$ should be selected that maximizes the term $\frac{\Delta_P(\mathbf{f})}{\Delta_T(\mathbf{f})}$. Since the term $\{\prod_{j=1}^{q}[1 - \sum_{i=1}^{n} \mathbf{p}(c_i, t_{\mathbf{f_j}})\mathbf{b}(c_i, \mathbf{f_j})]\}$ is fixed no matter what next action $\mathbf{f}$ is selected, our strategy becomes to select the next action that maximizes the term

$$\mathbf{E}(\mathbf{f}) = \frac{\sum_{i=1}^{n} \mathbf{p}(c_i, t_{\mathbf{f}})\mathbf{b}(c_i, \mathbf{f})}{\mathbf{t_o}(\mathbf{f})} \tag{8}$$

In some situations, the simple greedy strategy may even generate optimal answers. By using Lemma 2, we can prove the following result.

Lemma 3: *Suppose the available operations* $\mathbf{O_\Omega} = \{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_q\}$ *satisfy:*

(1) $\mathbf{t_o}(\mathbf{f}_i) = \mathbf{t_o}(\mathbf{f}_j)$, $1 \leq i, j \leq q$.

(2) $\Omega(\mathbf{f}_j) \bigcap \Omega(\mathbf{f}_i) = \phi$, $1 \leq i, j \leq q, i \neq j$.

Then, the above greedy strategy generates optimal answer.

It is interesting to note that the greedy strategy can sometimes minimize the expected time to detect the target. For a given effort allocation $\mathbf{F} = \{\mathbf{f}_1, \ldots, \mathbf{f}_k\}$, the expected time to detect the target by applying $\mathbf{F}$ is defined as

$$
\begin{aligned}
\Theta[\mathbf{F}] &= \mathbf{t}(\mathbf{f}_1) \times P(\mathbf{f}_1) \\
&+ \Big(\mathbf{t}(\mathbf{f}_1) + \mathbf{t}(\mathbf{f}_2)\Big) \times [1 - P(\mathbf{f}_1)]P(\mathbf{f}_2) \\
&+ \ldots \\
&+ \Big(\mathbf{t}(\mathbf{f}_1) + \ldots + \mathbf{t}(\mathbf{f}_k)\Big) \times \{\prod_{i=1}^{k-1}[1 - P(\mathbf{f}_i)]\}P(\mathbf{f}_k) \qquad (9)
\end{aligned}
$$

For $\Theta[\mathbf{F}]$ defined above, we have the following result (please refer to [52] for proofs).

Property 1 *For a group of actions* $\{\mathbf{f_1}, \ldots, \mathbf{f_q}\}$*, if there is no cube belonging to any two action ranges:* $\Omega(\mathbf{f}_j) \bigcap \Omega(\mathbf{f}_i) = \phi$*, if* $i \neq j$*, then the application order of the actions selected by our strategy minimize the expected time of detecting the target with respect to this group.*

For the model of search agent discussed in this paper, the cost of moving the robot is much higher than the cost of changing the camera direction and visual angle size. Thus, we further simplify the "one step look ahead" problem into two separate phases: the "where to look next" phase and the "where to move next" phase. The "where to look next" task selects the parameters p, t, w, h, a for the next action to be executed when the robot position is fixed. The "where to move next" task selects the next robot position when the benefit of simply adjusting the state parameters of the camera is too low.

4 Detection Function

From Section 2 and Section 3, we can see that the detection function $\mathbf{b}(c, \mathbf{f})$ is frequently used in calculating the probability of detecting the target and in updating the environment probability distributions. Obviously we cannot generate the value of $\mathbf{b}(c, \mathbf{f})$ at run time, so we should obtain the detection function values before the search process and retrieve these values whenever needed.

The standard detection function $\mathbf{b_0}((\theta, \delta, l), < a, w, h >)$ gives a measure of the detecting ability of the recognition algorithm a when no previous action has been applied. In this formula, $< w, h >$ is the viewing angle size of the camera; (θ, δ, l) is the relative position of the center of the target to the camera; $\theta = arctan(\frac{x}{z})$, $\delta = arctan(\frac{y}{z})$ and $l = z$, where (x, y, z) are the coordinates of the target center in CCS. We assume that the target is within the field of view of the camera, thus $-\frac{w}{2} \leq \theta \leq \frac{w}{2}$, $-\frac{h}{2} \leq \delta \leq \frac{h}{2}$, and $l > 0$. The value of $\mathbf{b_0}((\theta, \delta, l), < a, w, h >)$ can be obtained by experiment. We can first put the target at (θ, δ, l) and then perform experiments under various conditions, such as light intensity, background situation, and the relative orientation of the target with respect to the camera center. The final value will be the total number of successful recognitions divided by the total number of experiments. Usually, we tessellate the space of θ, δ, l into grids and record the approximate detection function value for each grid. These values can be stored in a look up table indexed by θ, δ, l and retrieved when needed. Sometimes we may approximate these values by analytic formulas.

It is not necessary to record the detection function values of all the different sizes of the solid viewing angles. We only need the detection values of one camera angle size (we call it the reference angle), and those of the other camera angle sizes can be obtained approximately by transforming them into those of the known camera angle size.

Suppose we know the detection function values for viewing angle size $< w_0, h_0 >$. We want to find the detection function values for viewing angle size $< w, h >$. To get the value of $\mathbf{b}(< \theta, \delta, l >, < w, h >)$ for a given $< \theta, \delta, l >$, we need to find the values of $(\theta_0,\ \delta_0,\ l_0)$ for angle size $< w_0, h_0 >$ that make the following approximation true:

$$\mathbf{b}(< \theta, \delta, l >, < w, h >) \approx \mathbf{b}(< \theta_0, \delta_0, l_0 >, < w_0, h_0 >) \tag{10}$$

The approximation relation (Formula (10)) means that when we use the recognition algorithm to analyze the picture taken with parameters $(< \theta, \delta, l >, < w, h >)$ and the picture taken with parameters $(< \theta_0, \delta_0, l_0 >, < w_0, h_0 >)$, we should get almost the same result. To guarantee this, the images of the target object should be almost the same for both imaging parameters, i.e., they must be approximately equal in at least two geometric factors, namely the scale factor and the position factor. The scale factor refers to the size of the projection of the target object on the image plane. The position factor refers to the position on the image plane of the projection of the center of the target object.

We use the scale factor to find the value of l_0 when l is given. The sizes of the projection of the target object on the image planes for $(<\theta, \delta, l>, <w, h>)$ and $(<\theta_0, \delta_0, l_0>, <w_0, h_0>)$ are approximately determined by l and l_0, respectively. The scale factor tells us that for a target patch that is parallel to the image plane, the area of its projection on the image plane for $(<\theta, \delta, l>, <w, h>)$ should be same as the area of its projection on the image plane for $(<\theta_0, \delta_0, l_0>, <w_0, h_0>)$. Suppose that the width of the image plane is W and the height of the image plane is H. Since the size of the image plane remains constant for different focal lengths, W and H will be same for any focal length.

Suppose the area of the target patch is S; the area of the projected target image for $(<\theta, \delta, l>, <w, h>)$ is S'; and the area of the projected target image for $(<\theta_0, \delta_0, l_0>, <w_0, h_0>)$ is S_0'. From the similarity relation between the target patch and its projected image, it is easy to show that

$$S' = \frac{f^2}{l^2} S \tag{11}$$

Since

$$tan(\frac{w}{2}) = \frac{\frac{W}{2}}{f}$$

and

$$tan(\frac{h}{2}) = \frac{\frac{H}{2}}{f}$$

we have

$$S' = \frac{f^2}{l^2} S = \frac{WH}{4l^2 tan(\frac{w}{2}) tan(\frac{h}{2})} S \tag{12}$$

Similarly,

$$S_0' = \frac{WH}{4l_0^2 tan(\frac{w_0}{2}) tan(\frac{h_0}{2})} S \tag{13}$$

To guarantee $S' = S_0'$, we get:

$$l_0 = l\sqrt{\frac{tan(\frac{w}{2})tan(\frac{h}{2})}{tan(\frac{w_0}{2})tan(\frac{h_0}{2})}} \quad (14)$$

We use the position factor to find the values of θ_0, δ_0 when θ and δ are given. Suppose D is the center of target patch with respect to $(<\theta, \delta, l>, <w, h>)$; $D^{'}(x^{'}, y^{'}, z^{'})$ is the image of D on the image plane with respect to $(<\theta, \delta, l>, <w, h>)$; D_0 is the center of target patch with respect to $(<\theta_0, \delta_0, l_0>, <w_0, h_0>)$; and $D_0^{'}(x_0^{'}, y_0^{'}, z_0^{'})$ is the the image of D_0 on the image plane with respect $(<\theta_0, \delta_0, l_0>, <w_0, h_0>)$, where $(x^{'}, y^{'}, z^{'})$ and $(x_0^{'}, y_0^{'}, z_0^{'})$ are in CCS. Then we have

$$x^{'} = ftan(\theta) = \frac{\frac{W}{2}}{tan(\frac{w}{2})}tan(\theta) = \frac{W}{2}\frac{tan(\theta)}{tan(\frac{w}{2})}. \quad (15)$$

Similarly,

$$y^{'} = \frac{H}{2}\frac{tan(\delta)}{tan(\frac{h}{2})} \quad (16)$$

$$x_0^{'} = \frac{W}{2}\frac{tan(\theta_0)}{tan(\frac{w_0}{2})} \quad (17)$$

$$y_0^{'} = \frac{H}{2}\frac{tan(\delta_0)}{tan(\frac{h_0}{2})} \quad (18)$$

To guarantee $x^{'} = x_0^{'}$ and $y^{'} = y_0^{'}$, we get

$$\theta_0 = arctan[tan(\theta)\frac{tan(\frac{w_0}{2})}{tan(\frac{w}{2})}] \quad (19)$$

and

$$\delta_0 = arctan[tan(\delta)\frac{tan(\frac{h_0}{2})}{tan(\frac{h}{2})}] \quad (20)$$

So, when we want to find the detection function value for parameters $<\theta, \delta, l>$ with respect to the camera angle size $<w, h>$, we can first find the corresponding $<\theta_0, \delta_0, l_0>$, then retrieve the detection function value for $\mathbf{b}(<\theta_0, \delta_0, l_0>, <w_0, h_0>)$ from the look up table or from the analytical formula.

It is interesting to note that the above calculations are consistent with respect to the misfocus property of the two visual angle sizes: the area of the blur circle for $(<\theta, \delta, l>, <w, h>)$ is the same as the area of the blur circle for $(<\theta_0, \delta_0, l_0>, <w_0, h_0>)$. Suppose the diameter of the blur circle produced by $(<\theta, \delta, l>, <w, h>)$ is c, the focus distance for l is v, and the diameter of aperture is a. We have

$$\frac{c}{a} = \frac{v-f}{v} \tag{21}$$

$$\frac{1}{l} + \frac{1}{v} = \frac{1}{f} \tag{22}$$

Thus, $c = \frac{af}{l}$. Similarly, we have $c_0 = \frac{af_0}{l_0}$, where c_0 is the diameter of the blur circle produced by $(<\theta_0, \delta_0, l_0>, <w_0, h_0>)$.

Since

$$l_0 = l\sqrt{\frac{tan(\frac{w}{2})tan(\frac{h}{2})}{tan(\frac{w_0}{2})tan(\frac{h_0}{2})}} = l\frac{f_0}{f}, \tag{23}$$

we have

$$c_0 = \frac{af_0}{l_0} = \frac{af_0}{l\frac{f_0}{f}} = \frac{af}{l} = c \tag{24}$$

It is also interesting to note that when the configurations of two operations are very similar, they might be correlated with each other —– the detection function values of the second operation are influenced by the previous applied actions. For example, if we apply the same operation **f** two times, and the first application fails, then the detection function value for **f** in its second application is 0, instead of that calculated from the above described method. The reason is that the camera configurations and the surrounding environment for the two applications are same. Since this situation makes the first application of **f** fail, it will certainly make the second application of **f** fail. In order to reduce the error caused by the action correlation, we should only select those actions whose configurations are not similar. Later in this paper, we will introduce the concept of a layered sensed sphere. The actions to be selected are based on the layered sensed sphere and the configurations of these actions are quite different. Thus, the error caused by the correlation between actions can be reduced.

5 Sensed Sphere

A representation of the non-occluded space around the camera is needed. Any lines emanating from the center of the camera will hit a solid object in the environment. The environment around the camera can thus be represented by the union of all these lines. This representation can be discretized by dividing the space around the camera with a series of solid angles. Each solid angle is associated with a radius which is the length of the line from the camera center in the direction of the central axis of the solid angle. The environment can thus be represented by the union of these solid angles. This representation is called the **sensed sphere** [43]. We can use a laser to construct the sensed sphere. In order to do this, we need to **tessellate** the surface of the unit sphere so as to get a series of uniform or near uniform patches which partition the surface. Then we can ping the laser at the center of each patch to construct the sensed sphere. There are many ways to tessellate the surface. One interesting way is to use geodesic domes. However, for our concern, this is not the best choice because the centers of the triangular facets are not regular, which necessitates frequent changes of the pan and tilt together in order to ping the next center. This is not efficient when the pan and tilt operations cannot overlap in time, since the mechanical operation might waste time.

In order to make the tessellation as uniform as possible and to make the number of mechanical operations as small as possible, we use the following method. First, we tessellate the range $[0, \pi]$ of tilt uniformly by a factor $2m$, where m is integer. This tessellation in general depends on the complexity of the environment. Thus, the tilt ranges are $[0, \alpha), \ldots, [(i-1)\alpha, i\alpha), \ldots, [\pi - \alpha, \pi)$, where $\alpha = \frac{\pi}{2m}$. Each tilt range $[t_{b_i}, t_{e_i})$ corresponds to a horizontal circular surface slice. The range of pan corresponding to each slice is $[0, 2\pi)$. Then, for each tilt range $[t_{b_i}, t_{e_i})$ except $[0, \alpha)$ and $[\pi - \alpha, \pi)$, we tessellate the range $[0, 2\pi)$ of pan. Each pan range $[p_{b_{i,j}}, p_{e_{i,j}})$ and its corresponding tilt range $[t_{b_i}, t_{e_i})$ defines a surface patch whose pan and tilt is confined by $t_{b_i} \leq tilt < t_{e_i}$ and $p_{b_{i,j}} \leq pan < p_{e_{i,j}}$. The goal of our tessellation is to make the areas of the resulting surface patches approximately equal. At the equator, the change of pan is α, which is the same as the change of tilt. The area of patches at the equator is approximately α^2. However, the area of each surface patch for other *tilt*s will differ with this tessellation scheme. In order to keep patch areas approximately equal, we adjust the amount of pan for every different *tilt*. Figure 2(a) shows a surface patch on the equator and a surface patch that is not on the equator. Our strategy is to select the pan such that $| CD |=| AB |$. We then obtain that the change of pan Δ_i for tilt range $[(i-1)\alpha, i\alpha)$ is

$$\Delta_i = \begin{cases} 2arcsin(\frac{sin\frac{\alpha}{2}}{sin(i\alpha)}), & \text{if } i\alpha \leq \frac{\pi}{2} \\ 2arcsin(\frac{sin\frac{\alpha}{2}}{sin[(i-1)\alpha]}), & \text{if } (i-1)\alpha \geq \frac{\pi}{2} \end{cases} \tag{25}$$

So, for tilt range $[(i-1)\alpha, i\alpha)$, the pan ranges are $[0, \Delta_i)$, $[\Delta_i, 2\Delta_i)$, ..., $[n_i\Delta_i, 2\pi)$. The length of each pan range is Δ_i except for the last range $[n_i\Delta_i, 2\pi)$. Figure 2(b) shows a side view of a tessellation with $m = 10$. The sensed sphere constructed with the tessellation scheme described above can be concisely represented as the union of all solid angles

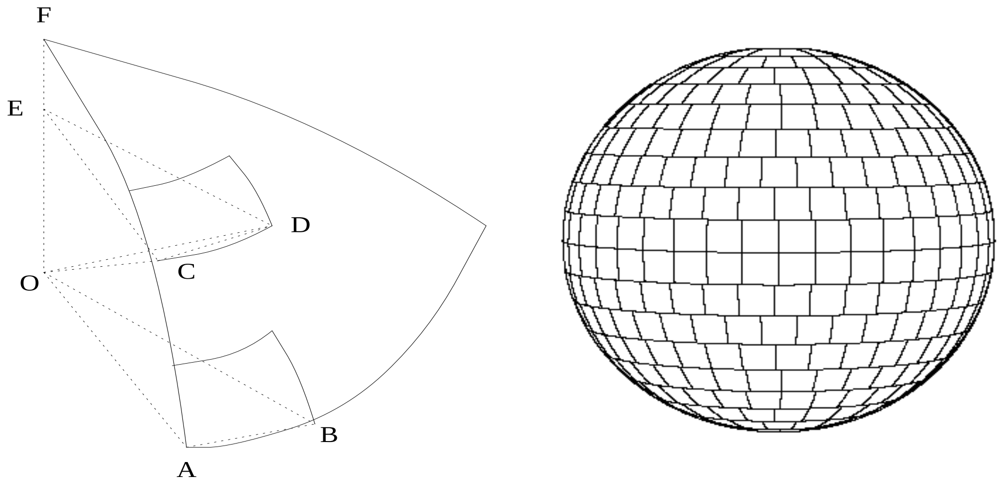
$$\bigcup_{[t_{b_i}, t_{e_i})=[(i-1)\alpha, i\alpha), i=1,\ldots,2m} \left\{ \bigcup_{[p_{b_{i,j}}, p_{e_{i,j}})=[0,\Delta_i),[\Delta_i,2\Delta_i),\ldots,[n_i\Delta_i,2\pi)} A_{ij}(t_{b_i}, t_{e_i}; p_{b_{i,j}}, p_{e_{i,j}}; r_{ij}) \right\}, \tag{26}$$

where r_{ij} is the length of the radius along the direction $tilt = \frac{t_{b_i}+t_{e_i}}{2}$, $pan = \frac{p_{b_{i,j}}+p_{e_{i,j}}}{2}$. The solid angle A_{ij} is bounded by $t_{b_i} \leq tilt < t_{e_i}, p_{b_{i,j}} \leq pan < p_{e_{i,j}}$, and r_{ij}.

The sensed sphere can be used to register the solidity property of the search region. If the distance of the center of a cube to the camera center is almost the same as the corresponding radius of the solid angle containing this cube and the distance between the cube center and the point hit by the laser is less than a given threshold, then the **solidity** of this cube is registered as *solid*. Otherwise, the **solidity** of this cube is registered as *unknown*.

6 Where to Look Next

We need to select w, h, p, t, a such that the term $\mathbf{E}(\mathbf{f})$ defined in Equation (8) is maximized. We first consider the situation where only one recognition algorithm is available. Since the cost for any operation with the same recognition algorithm is the same (see assumptions in Section 3), our task becomes the selection of w, h, p, t such that $\mathbf{P}(\mathbf{f})$ is maximized. This



(a) (b)

Figure 2:

Tessellation of the unit sphere.

(a) The change of pan for a given tilt. (b) The side view of a tessellated sphere with $\alpha_0 = 0.157$ *(9 degrees)*

task is time consuming because of the huge number of possible actions, as determined by the total number of camera configurations that can be achieved by the hardware. In our case, the number of configurations is the product of: (1) the number of different viewing angle sizes; (2) the number of different pan angles; and (3) the number of different tilt angles. To make the problem tractable, we must decompose this huge space of possible sensing actions into a finite set of actions to be tried and select the next action among the resulting limited set of actions.

6.1 Determine the Necessary Viewing Angle Size

We first select the necessary camera angle sizes. For a given robot position and a given recognition algorithm, there are many possible viewing angle sizes. However, the whole search region can be examined with high probability of detection using only a small number of them. The ability of the recognition algorithm and the value of the detection function are influenced by the image size of the target. Usually the recognizer can successfully recognize the target only when the image size of the target is within a certain range such that the whole target can be brought into the field of view of the camera and the features can be detected with the required precision. For an operation with a given recognition algorithm and a fixed viewing angle size, the probability of successfully recognizing the target is high only when the target's distance is within a certain range. Therefore, different sizes of the viewing angle $< w, h >$ will be associated with different **effective ranges** of distance. Our purpose here is to select those angles whose effective ranges will cover the entire depth D of the search region, and at the same time there will be no overlap of their effective ranges.

Suppose that the biggest viewing angle for the camera is $< w_0 \times h_0 >$, and its effective range is $[N_0, F_0]$. We can use geometric constraint to find other required viewing angles $< w_1 \times h_1 >, \ldots, < w_{n_0} \times h_{n_0} >$ and their corresponding effective ranges $[N_1, F_1], \ldots, [N_{n_0}, F_{n_0}]$, such that $[N_0, F_0] \bigcup \ldots \bigcup [N_{n_0}, F_{n_0}] \supseteq [N_0, D]$ and $[N_i, F_i) \bigcap [N_j, F_j) = \emptyset$ if $i \neq j$. These $n_0 + 1$ angle sizes are enough to examine the whole depth of the search space with high probability.

Since the effective range $[N_{i+1}, F_{i+1}]$ of the next viewing angle $< w_{i+1} \times h_{i+1} >$ should be adjacent to the effective range of the current viewing angle $< w_i \times h_i >$, we have $N_{i+1} = F_i$. To guarantee that the areas of the images of the target patch of $< w_i \times h_i >$ at N_i and F_i are equal to the areas of the images of the target patch of $< w_{i+1} \times h_{i+1} >$ at N_{i+1} and F_{i+1}, respectively, we obtain (using Equations (14)):

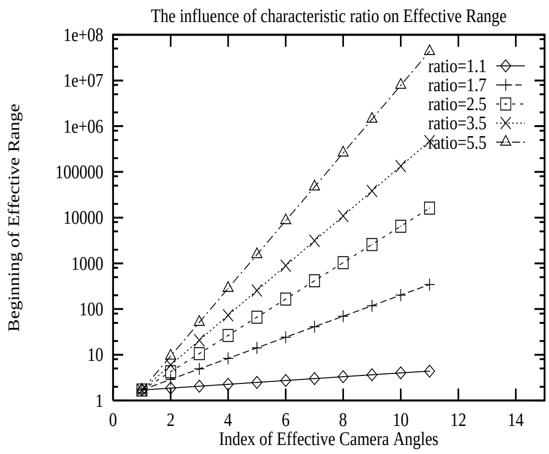
$$w_i = 2arctan[(\frac{N_0}{F_0})^i tan(\frac{w_0}{2})] \tag{27}$$

$$h_i = 2arctan[(\frac{N_0}{F_0})^i tan(\frac{h_0}{2})] \tag{28}$$

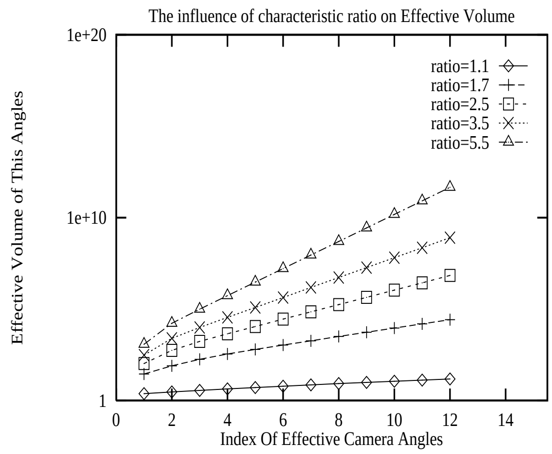
$$N_i = F_0(\frac{F_0}{N_0})^{i-1}; F_i = F_0(\frac{F_0}{N_0})^i \tag{29}$$

Since $N_i \leq D$, we obtain $i \leq \frac{ln(\frac{D}{F_0})}{ln(\frac{F_0}{N_0})} - 1$. Let $n_0 = \lfloor \frac{ln(\frac{D}{F_0})}{ln(\frac{F_0}{N_0})} - 1 \rfloor$, then the angles to be considered by the sensing actions are $< w_0 \times h_0 >, < w_1 \times h_1 >, \ldots, < w_{n_0} \times h_{n_0} >$.

Figure 3(a) shows that for large characteristic ratio $\frac{F_0}{N_0}$, the beginning of the effective range will increase very quickly as the index of the effective camera angle increases (if the



(a)



(b)

Figure 3:

The Influence of Characteristic Ratio when $N_0 = 1.7$, $w_0 = h_0 = 1.046$ *radian* (60^o)

hardware can provide the corresponding angle size). Figure 3(b) shows that the total effective volume (the $3D$ space within the effective range and the viewing volume) will increase as the index of the effective camera angle increases. This suggests that when there is no occlusion, it is better to remain in one place and search the space as far as the camera can see, rather than drive the robot near the unsearched space.

The sensed sphere can be further divided into several layers according to the effective viewing angle sizes derived above. This layered sensed sphere LSS can be represented as

$$LSS = \bigcup_{<w_k,h_k>, k=0,\ldots,n_0} LSS_{<w_k,h_k>}, \tag{30}$$

where $LSS_{<w_k,h_k>}$ is the layer corresponding to viewing angle size $<w_k, h_k>$,

$$LSS_{<w_k,h_k>} = \bigcup_{i,j} LA_{ij}^{<w_k,h_k>}. \tag{31}$$

The layered solid angle slice $LA_{ij}^{<w_k,h_k>}$ is the intersection of the solid angle A_{ij} of the sensed sphere and the current layer $LSS_{<w_k,h_k>}$. There is a probability $p_{ij}^{<w_k,h_k>}$ associated with $LA_{ij}^{<w_k,h_k>}$, which gives the sum of the probabilities of all the cubes that belong to $LA_{ij}^{<w_k,h_k>}$. In other words, $p_{ij}^{<w_k,h_k>}$ is the sum of the probability for each cube which satisfies the following: (1) it is within A_{ij}; (2) the distance to the camera center is less than r_{ij}; and (3) the distance to the camera center is within the effective range of $< w_k, h_k >$, which means that the distance should be greater than $N_{<w_k,h_k>}$ and less than $F_{<w_k,h_k>}$.

6.2 Determining the Necessary Viewing Directions for a Given Angle Size

For a given effective angle size $< w, h >$ selected by the above method, there are a huge number of viewing directions that can be considered. Each direction (p, t) corresponds to a rectangular pyramid which is the viewing volume determined by parameters $< w, h, p, t >$. Within this viewing volume, only a slice of the pyramid can be examined with high detection probability by the given recognition algorithm, i.e. those parts that are within $LSS_{<w,h>}$. We call this slice of pyramid the **effective volume** for parameter $< w, h, p, t >$. The union of the effective volumes of all the possible $< p, t >$ with respect to the given $< w, h >$ will cover the given layer $LSS_{<w,h>}$. But to examine $LSS_{<w,h>}$, it is not necessary to try every possible (p, t) one by one, we only need to consider those directions such that the union of their effective volumes cover the whole layer $LSS_{<w,h>}$ with little overlap.

The selection of the candidate directions is similar to the tessellation of the sensed sphere, but not identical. The surface patterns cut by the camera's viewing volume are different from the surface patterns obtained from our previous tessellation method. Let $\alpha = min\{w, h\}$. We study in the following how to select the necessary camera viewing directions for angle size $< \alpha, \alpha >$ such that the selected actions can cover the whole sphere.

Suppose the viewing direction of the camera (pan and tilt) is $< p, t >$. We want to determine the range of surface patch on the sphere that can be covered by the viewing volume of the camera. This covered surface patch should be specified by a tilt range $t_b \leq tilt \leq t_e$ and a pan range $p_b \leq pan \leq p_e$. Our purpose in the following is to find the values of t_b, t_e, p_b, and p_e such that the corresponding surface patch can be covered by the camera viewing

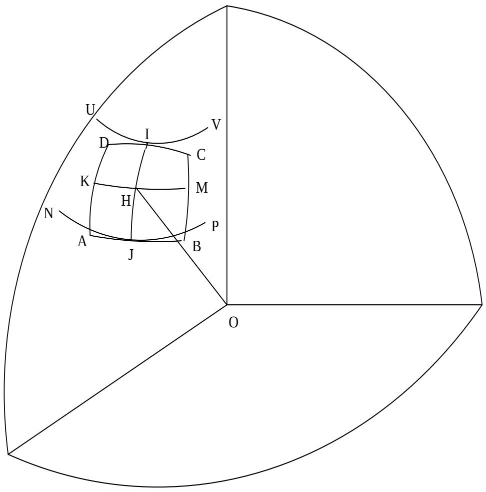


Figure 4:

The surface patch that is contained within the camera's viewing volume, where OH is the viewing direction of camera. $\widehat{IJ}$ is the big circle arc which is cut by the vertical plane of the viewing volume (YZ plane in Camera Coordinate System). $\widehat{KM}$ is the big circle arc which is cut by the horizontal plane of the viewing volume. $\widehat{AB}$, $\widehat{BC}$, $\widehat{CD}$ and $\widehat{DA}$ are the big circle arcs cut by the bounding plane of the viewing volume. $\widehat{NJP}$ is part of the locus of points on the sphere whose tilt is $t + \frac{\alpha}{2}$, and $\widehat{UIV}$ is part of the locus of points on the sphere whose tilt is $t - \frac{\alpha}{2}$.

volume with direction (p, t).

First, we need to find the range of pan $\Delta_{pan} = p_e - p_b$ for the given viewing direction of the camera. From Figure 4, we can see that any point on arc $\widehat{BMC}$ has a corresponding point on arc $\widehat{AKD}$ such that they have the same value of tilt. The smallest difference of pan for a point on $\widehat{BMC}$ and its corresponding point on $\widehat{AKD}$ occurs at point B and point A. We take the difference of the range of pan for points B and A as the value of Δ_{pan}. After a detailed calculation, we obtain

$$\Delta_{pan} = 2arctan\{\frac{sin(\frac{\alpha}{2})}{sin(t + \frac{\alpha}{2})}\}. \tag{32}$$

But for this change of pan, the viewing volume cannot cover all the surface patch whose tilt value is between $t - \frac{\alpha}{2}$ and $t + \frac{\alpha}{2}$. From Figure 4, we can see that the big circle arc $\widehat{IC}$ gradually goes down (decreases in Z value) as the point moves from I to C. So, there is a part between big circle arc $\widehat{DIC}$ and arc $\widehat{UIV}$ that can not be covered by the camera's viewing volume when the intended pan range is Δ_{pan}. We need to determine the tilt value of a point as it approaches C from I along the big circle arc $\widehat{IC}$ with the change of pan equal to $\frac{\Delta_{pan}}{2}$. After some calculations, we obtain the tilt value

$$arccos\{\frac{cos(t - \frac{\alpha}{2})}{\sqrt{1 + \frac{sin^2(\frac{\alpha}{2}) sin^2(t - \frac{\alpha}{2})}{sin^2(t + \frac{\alpha}{2})}}}\}.$$

From above calculations, it is easy to show that the following sphere surface patch can be covered by the viewing volume with direction (p, t),

$$p - arctan\{\frac{sin(\frac{\alpha}{2})}{sin(t + \frac{\alpha}{2})}\} \leq pan \leq p + arctan\{\frac{sin(\frac{\alpha}{2})}{sin(t + \frac{\alpha}{2})}\} \tag{33}$$

$$arccos\{\frac{cos(t - \frac{\alpha}{2})}{\sqrt{1 + \frac{sin^2(\frac{\alpha}{2}) sin^2(t - \frac{\alpha}{2})}{sin^2(t + \frac{\alpha}{2})}}}\} \leq tilt \leq t + \frac{\alpha}{2} \tag{34}$$

Suppose the final viewing direction set is $S_{candidate}$. According to the above calculations, we can get the following algorithm for enumerating the viewing directions to cover the whole sphere.

1. $S_{candidate} = \phi$
2. $p \longleftarrow 0$, $t \longleftarrow 0$, $S_{candidate} = S_{candidate} \bigcup < p, t >$.
3. $t_e \longleftarrow \frac{\pi}{2}$
4. $t_b \longleftarrow arccos\{\frac{cos((t_e - \frac{\alpha}{2}) - \frac{\alpha}{2})}{\sqrt{1 + \frac{sin^2(\frac{\alpha}{2}) sin^2((t_e - \frac{\alpha}{2}) - \frac{\alpha}{2})}{sin^2((t_e - \frac{\alpha}{2}) + \frac{\alpha}{2})}}}\}$
5. Cover the slide on the sphere whose tilt is within the range of $[t_b, t_e]$ and the slide on the sphere whose tilt is within the range of $[\pi - t_e, \pi - t_b]$.

(a) Let $t \longleftarrow t_e - \frac{\alpha}{2}$.

(b) let $\Delta_{pan} \longleftarrow 2arctan\{\frac{sin(\frac{\alpha}{2})}{sin((t-\frac{\alpha}{2})+\frac{\alpha}{2})}\}$

(c) Use Δ_{pan} to divide $[0, 2\pi]$ for the given slide. So, we can get a series of $[p_b, p_e]$. They are $[0, \Delta_{pan}]$, $[\Delta_{pan}, 2\Delta_{pan}]$, ..., $[k\Delta_{pan}, 2\pi]$. Note: the length of the last interval may not be Δ_{pan}.

(d) For each division, let $p \longleftarrow \frac{p_b+p_e}{2}$. Then perform $S_{candidate} = S_{candidate} \bigcup < p, t >$ and $S_{candidate} = S_{candidate} \bigcup < p, \pi - t >$.

6. Let $t_e \longleftarrow t_b$

7. If $t_e \leq \alpha$, stop the process. Otherwise Goto 4.

6.3 Selecting the Next Action

For a given recognition algorithm a, we can find a list of candidate viewing angle sizes $w_0 \times h_0, w_1 \times h_1, \ldots, w_{n_0} \times h_{n_0}$. For each candidate angle size $< w_k, h_k >$ $(0 \leq k \leq n_0)$, we can find a series of camera viewing directions that must be considered. To select the best directions among these candidate directions, we need to use $\mathbf{P}(\mathbf{f})$ to compare them. The region that is the intersection of the sensed sphere, the corresponding layer of the given viewing angle size $< w_k, h_k >$, and the viewing volume determined by $< p, t >$ is the effective volume for $< w_k, h_k, p, t >$, denoted as $V(< w_k, h_k, p, t >)$. Since only those cubes that belong to the effective volume make major contribution to $\mathbf{P}(\mathbf{f})$, we need only take these cubes into consideration when calculating $\mathbf{P}(\mathbf{f})$. In practice, we may use the probability of the presence of the target object within the effective volume

$$\bigcup_{LA_{ij}^{<w_k,h_k>} \in V(<w_k,h_k,p,t>)} p_{ij}^{<w_k,h_k>}$$

to compare among candidate directions.

For each candidate angle size $< w_k, h_k >$ $(0 \leq k \leq n_0)$, we find a best direction $< p_k, t_k >$. Thus, we obtain $n_0 + 1$ candidate actions $\mathbf{f}_0 = \mathbf{f}(p_0, t_0, w_0, h_0, a)$, $\mathbf{f}_1 = \mathbf{f}(p_1, t_1, w_1, h_1, a)$, ..., $\mathbf{f}_{n_0} = \mathbf{f}(p_{n_0}, t_{n_0}, w_{n_0}, h_{n_0}, a)$. Then we calculate the probability of detection for each $\mathbf{f}_k$,

$0 \leq k \leq n_0$ according to $\mathbf{P}(\mathbf{f})$ and select the one with highest value as the best candidate action with respect to the recognition algorithm a.

For any recognition algorithm a, we obtain a best candidate action $\mathbf{f_a}$. Suppose $a_1, a_2, \ldots, a_m$ are the recognition algorithms available for the search task, then we obtain m candidate actions $\mathbf{f_{a_1}}, \ldots, \mathbf{f_{a_m}}$. For each candidate action, we calculate its utility function value according to $\mathbf{E}(\mathbf{f})$. Then the action with the highest utility value will be taken as the next action to be executed.

After applying each action, we will update the probability of any cubes within the search space. Since the updating process might take time, we can select several operations under the current world state, and update the state after these operations are applied. The probability of presence within the current sensed sphere SS is $Prob_{SS} = \sum_{c_i \in SS} \mathbf{p}(c_i)$. This probability will be decreased each time an operation is applied. When the ratio of $Prob_{SS}$ and the probability of presence within the search space $\frac{Prob_{SS}}{1-\mathbf{p}(c_o)}$ is less than a certain threshold, or when the probability of detecting the target by the selected action is too small, we will want the robot to change position.

7 Where to Move Next

7.1 Selecting the Next Robot Position

The goal of the "where to move next" algorithm is to select the next camera position such that, at the new position, the camera can examine those parts of the region that have high probability of presence of the target but are occluded if viewed from the previous positions. In order to perform this task, first we need to decide which candidate positions the camera **can reach**. Then we need to know which one is the **most beneficial** among these reachable positions. The strategy with respect to this task is greatly influenced by the hardware that control the camera. With respect to the hardware used in this paper, the goal becomes to select the robot's next position such that the sensed sphere at the new position has the highest probability presence of the target, with the constraint that the movement of the robot from the current position to the next position is on **straight line**.

In general, the robot can be at any position where there is no solid object within the volume occupied by the robot. Many of the possible positions are not necessary, however. If we know the geometric configuration of the search space, then we can select a set of candidate positions such that the robot can examine the whole search region without occlusion. If we have no knowledge of the geometric configuration of the search space, then we can tessellate the horizontal plane of the search region into squares and take the centers of the squares as the candidate positions to be considered. The side length of the square should be carefully designed so as to get a reasonable number of candidate positions.

The candidate positions discussed above can be further reduced by considering whether the robot can reach these positions from its current position. When the robot moves from its current position to a candidate position, there will be a volume of space that is traversed by the robot. According to the current registration of the space, if there are no solid cubes present within this volume, then we assume the robot can reach this candidate position; otherwise, we assume the robot cannot reach this candidate position. We only consider those candidate positions that the robot can reach by the above criteria.

Now we will select the next robot position among these reachable candidate positions. To do this, we need to know the approximate unoccluded region that can be seen by the camera with respect to each reachable candidate position. We call this region the expected sensed sphere SS_e. If we know the geometric configuration of the search space, we can directly calculate the SS_e. If we do not know the geometric configuration, we can calculate SS_e by using the boundary of the search region and the solidity property of the cubes in the region. First, we calculate the expected sensed sphere by just taking the boundary of the search region Ω (assumed to be known) as solid and **assume** that we ping the laser around in the calculated directions from this candidate position. Thus, we can **calculate** the radius for each solid angle of SS_e. Then we update these radii according to the current registration of the search space Ω. For each solid cube and its associated solid angle with respect to the expected sensed sphere SS_e, we attempt to update the radius as follows. If the distance from this cube to the center of the new camera position is less than the radius of the solid angle, we replace the radius of the solid angle by this distance; otherwise, the radius of the solid angle is not updated. In this way, we construct the expected sensed sphere SS_e for the current reachable candidate position.

Each candidate position is associated with an expected sensed sphere constructed by the above method. For each expected sensed sphere, we can calculate the probability of presence of the target by summing up the probabilities of all the cubes that are within this expected sensed sphere $\sum_{c_i \in SS_e} \mathbf{p}(c_i)$. Thus, for each candidate position, we can get its corresponding probability of presence of the target. We will then select the position whose corresponding expected sensed sphere has the maximum probability of presence of the target as our next position.

After we find the next position, we drive the robot to this position and begin our new search. Since the solidity property of the cube obtained so far may not correctly represent the real situation, the robot may stop in the middle of the journey. If it cannot reach the next position, then it will begin the "where to look next" phase at the place it has stopped.

At this point we must consider the inaccuracy of robot movement, since we need to know the robot position with respect to the world coordinate system after each movement. In general, we cannot solely depend on dead reckoning approaches, because they are not sufficiently reliable over long distances and the position error is compounded over sequences of movements as the robot traverses the environment. Here we use the method of external referencing. That is, we assume there are fixed, easily detectable landmarks or beacons in the environment. The locations of these beacons or landmarks, in the form of coordinates in the world coordinate system, serve as the reference for computing the robot position. This approach has the advantage that the position errors are bounded. Since position errors apply only to the horizontal direction (the robot height is fixed and the floor is horizontal) and are bounded, we can incorporate the associated uncertainty into the probability distribution as a blurring or convolution operation performed on the probability distribution of the cubes on each horizontal layer. For example, if we assume the robot position uncertainty is uniform and within a bounded error range, the following method can be used to incorporate this uncertainty into the probability distribution. When we move to a new place, the probability of presence of each cube will be the sum of the probabilities of presence for the cubes in the same layer within the uncertainty range centered at the corresponding cube divided by the number of these cubes. Note, this operation does not change the total sum of the probabilities of the cubes, and thus there is no need for normalization.

7.2 The Solidity Property

During the search process, the robot must acquire information about the solidity property of the cube. This solidity property is used to calculate the radius of the expected sensed sphere. Since the movement uncertainty is bounded, the error of the calculated radius of the expected sensed sphere is also bounded. As a result, the calculated probability of presence of the expected sensed sphere with respect to each candidate position is not influenced too much by the uncertainty of the robot movement. Therefore, we ignore movement uncertainty when updating the solidity of the cubes.

After the robot drives to a new position, we use the laser to construct the sensed sphere again. After the construction, we register the space according to this new sensed sphere. We combine the new and old registration to update the registration file as shown in Table 1.

Old	*New*	*Combined*
solid	solid	solid
solid	unknown	solid
unknown	solid	solid
unknown	unknown	unknown

Table 1: The combination of the solidity property

At the new position, the robot will begin the “where to look next” process again.

7.3 When to Stop Searching

The stopping criterion is determined by assessing the benefit of continued searching. If the total probability that the target is inside the search region $(1 - p(c_o))$ is less then a certain threshold, or the maximum probability of presence of the next robot position (Section 7.1) is less then a certain threshold, then we stop the search process.

8 Testing the Model

Both simulation experiments and real experiments are performed to test the sensor planning strategy proposed in this paper. The search model is based on the ARK robot. We assume only one recognition algorithm is available. The algorithm used in the experiments is as follows:

1. Determine the necessary angle sizes according to the diameter of the search region.
2. Generate the layered sensed sphere corresponding to those angle sizes obtained in Step 1.
3. Find the top directions for each angle size.
4. Use $\mathbf{P}(\mathbf{f})$ to compare the best directions of different angle sizes, and select top actions among them.
5. Apply these selected top actions and update the environment.
6. Goto Step 2 when the probability within current sensed sphere is **not** small enough.
7. Stop search when the probability that the target is within the search region is small enough.
8. Select the next best position and drive the robot to that position.
9. Combine the search region of the two positions (Section 7.2).
10. Go to Step 2 and begin the search at new position.

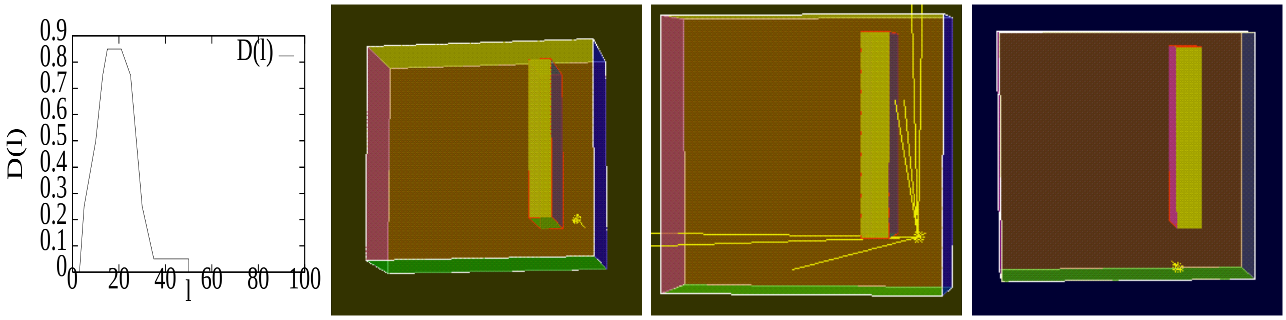
8.1 Simulation

The first simulation experiment is used to test the general scheme of our algorithm. The biggest angle for the sensor is $\frac{\pi}{4} \times \frac{\pi}{4}$. The detection function is $\mathbf{b_0}((\theta, \delta, l), < a, \frac{\pi}{4}, \frac{\pi}{4} >) = D(l)(1 - \frac{1}{6}\frac{\theta}{\frac{\pi}{4}})(1 - \frac{1}{6}\frac{\delta}{\frac{\pi}{4}})$, where $D(l)$ is shown in Figure 5(a). The search region is shown in Figure 5(b). Only three angle sizes are needed to examine the search region with respect to the first robot position. They are $\frac{\pi}{4} \times \frac{\pi}{4}$, 0.376×0.376 and 0.165×0.165, respectively. Their effective ranges are $[11, 27]$, $[27, 66]$ and $[66, 162]$, respectively. We assume the outside probability is 0.5, and that the distribution within the room is uniform at the beginning. Figure 5(a) shows that the actions selected by our algorithm are used only to examine the unoccluded region. Figure 5(b) shows that the next robot position is selected to examine the region that is occluded.

The second simulation experiment is used to compare the efficiency of our strategy with the brute force strategy. The environment, the robot position and the sensor model for this simulation are the same as those of the first, except that there is no obstacle in the room. The target distribution satisfies a 3-variate normal distribution $N(\mu, \mathbf{\Sigma})$, where the mean vector $\mu = (25, 25, 15)^T$, and the covariance matrix $\mathbf{\Sigma} = diag(\sigma^2, \sigma^2, (\frac{\sigma}{2})^2)$. Thus, the initial probability distribution is given by

$$\mathbf{p}(x, y, z) = \frac{1}{(2\pi)^{\frac{3}{2}} \times 25 \times 25 \times 15} e^{-\frac{1}{2}[\frac{(x-25)^2}{\sigma^2} + \frac{(y-25)^2}{\sigma^2} + \frac{(z-15)^2}{(\frac{\sigma}{2})^2}]} \tag{35}$$

We use $P[\mathbf{F}]$ to compare the effects of our sensor planning strategy with the effects of the Non-Planning strategy. The Non-Planning strategy means that the action application sequence is not influenced by the agent's initial knowledge of the target probability distribution. Here, the Non-Planning strategy refers to the strategy of first selecting the actions belong to the first layer, then selecting the actions belong to the second layer, and finally, selecting the actions belong to the third layer. Figure 6 gives the effects of the two strategies for different initial probability distributions. Notice that the number of actions needed to reach the detection limit for our strategy is much smaller than that for Non-planning strategy. This illustrates that the planning strategy is more efficient.



(a) (b) (c) (d)

Figure 5:

Simulation 1.

(a) The $D(l)$ for detection function;

(b) The environment for simulation one. The size of the room is length $\times$ width $\times$ height $= 50\times50\times25$. There is an obstacle with size $5\times37\times19$ in the environment. The robot (represented by the white blob in the figure) is at position $(45, 10)$. The height of the robot is 15.

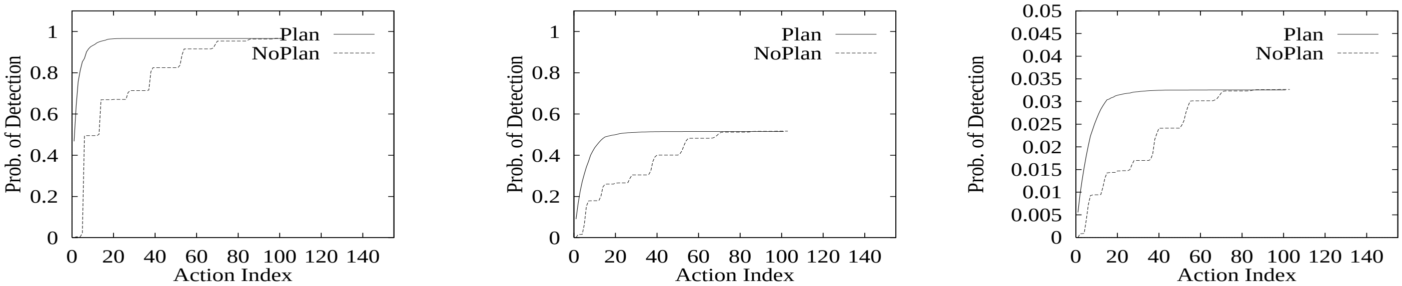
(c) Top 7 actions selected at $(45, 10)$. The short and long lines correspond to the viewing axis of actions with angle size $\frac{\pi}{4} \times \frac{\pi}{4}$, and 0.316×0.316 respectively.

(d) The next robot position $(36, 1)$ selected by the sensor planning system.

8.2 Real Experiments

Real experiments are performed using the Laser Eye and the ARK robot. The search region (Figure 7) is a part of the Vision Lab of Department of Computer Science at University of Toronto. The task is to search for a white baseball within this region. Four threshold criteria are used in the recognition algorithm: intensity I_0, blob size B_{min}, B_{max}, and roundness percentage R_0. The steps of the recognition algorithm are as follows:

1. Generate a binary image from the camera's grey image using a threshold. Pixels with intensity value bigger than I_0 receive intensity value 255. Pixels with intensity value less than or equal to I_0 receive intensity value 0.



(a) (b) (c)

Figure 6:

The detection probabilities $P[\mathbf{F}]$ *of the planning strategy (Plan) and those of the Non-planning strategy (NoPlan).*

(a)$\sigma = 5$.

(b)$\sigma = 15$.

(c)$\sigma = 50$.

2. Perform morphological operations on the image resulting from Step 1: first erosion then dilation.

3. Perform region growing on the output image from Step 2 and label white blobs.

4. Calculate the center and size (number of pixels) of each blob.

5. According to the center and size from Step 4, calculate the the corresponding circle with respect to each blob (i.e., the circle that has the same center as the blob and contains the same number of pixels as the blob), and calculate the percentage of the blob that falls into the circle.

6. If a blob size is between B_{min} and B_{max}, and more than R_0 percent of its pixels fall within the calculated circle, then this blob is taken as an image of the ball.

The first experiment tests the "where to look next" strategy. In the experiment, the parameters for the threshold are: $I_0 = 119$, $B_{min} = 250$ pixels, $B_{max} = 1375$ pixels, and

(a) (b)

Figure 7:

(a) The configuration of the search environment (size $6m \times 5.5m \times 2.5m$). Table surfaces are labeled A, B, C, and E. The Laser Eye is at the position F of table E. D is a region on the floor.

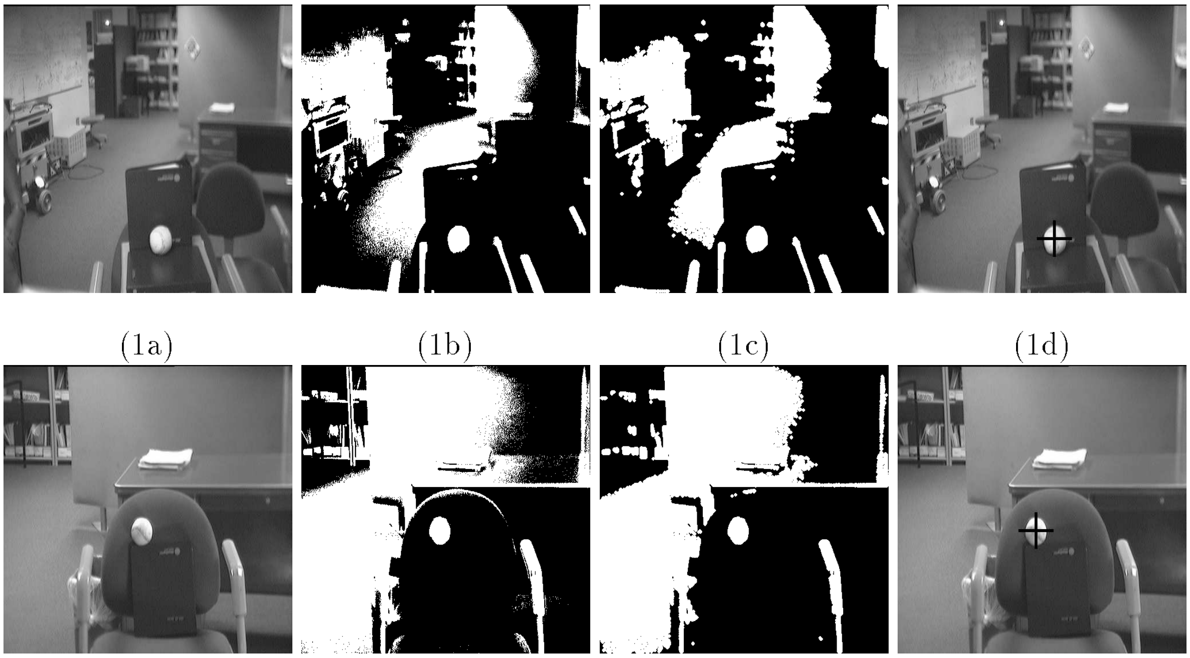
(b) Composite image of the region from position G of (a).

$R_0 = 91\%$ respectively. The Laser Eye is at F on table E (See Figure 7)(a)). We give the surface of table A, the surface of table B, and the surface of table C high target probability distribution at the beginning. The target is on table C.

First, we select the necessary angle sizes that must be used to examine the whole depth of the environment. We take the biggest angle size of the hardware $< w_0, h_0 >=< 41^o, 39^o >$ as the first angle size. The effective ranges for this angle size are obtained by experiments. The nearest distance for which the blob size is smaller than B_{max} is the value of N_0. The farthest distance for which the blob size is bigger than B_{min} is the value of F_0. We experimentally determined the effective range for $w_0 \times h_0$ as $N_0 = 1.47m$, $F_0 = 3.01m$. From equations (27), (28), and (29), we obtain the next effective angle size as $20.7^o \times 19.7^o$, and the next effective ranges are $N_1 = 3.01m$, $F_1 = 6.16m$. The output of the recognition algorithm is listed below in Table 2,

	Blob Size	Percentage		Blob Size	Percentage
N_0	1366 pixels	93.56%	N_1	1332 pixels	91.14%
F_0	273 pixels	91.58%	F_1	271 pixels	90.77%

Table 2: The output of the recognition algorithm.



(1a) (1b) (1c) (1d)

(2a) (2b) (2c) (2d)

Figure 8:

(1a) The image when angle size is $w_0 \times h_0$. *Target is 1.47m from the camera.*

(1b) The resultant binary image.

(1c) The image after the morphological operation.

(1d) The final result: target is detected.

(2a) The image when angle size is $w_1 \times h_1$. *Target is* $3.01m$ *from the camera.*

(2b), (2c), (2d) The recognition process applied to the image in (2a).

From Table 2 we can see that the values of the blob size for the first angle at N_0 and F_0 are quite close to the values of the blob size for the second angle at N_1 and F_1, respectively.

The process of running the recognition algorithm for angle size $< w_0, h_0 >$ at distance N_0 and angle size $< w_1, h_1 >$ at distance N_1 are shown in Figure 8.

Figure 9 shows experimental result of our search strategy. Figure 9(a) shows the sensed sphere constructed by the Laser Eye. Figure 9(b)(c)(d) are the top actions selected by our algorithm. The angle size for the first action is $41^o \times 39^o$, and the angle size for the second and third actions is $20.7^o \times 19.7^o$. Although the baseball appeared in the first image, the algorithm failed to detect it because it is outside the effective range of the action. The third action detects the target.

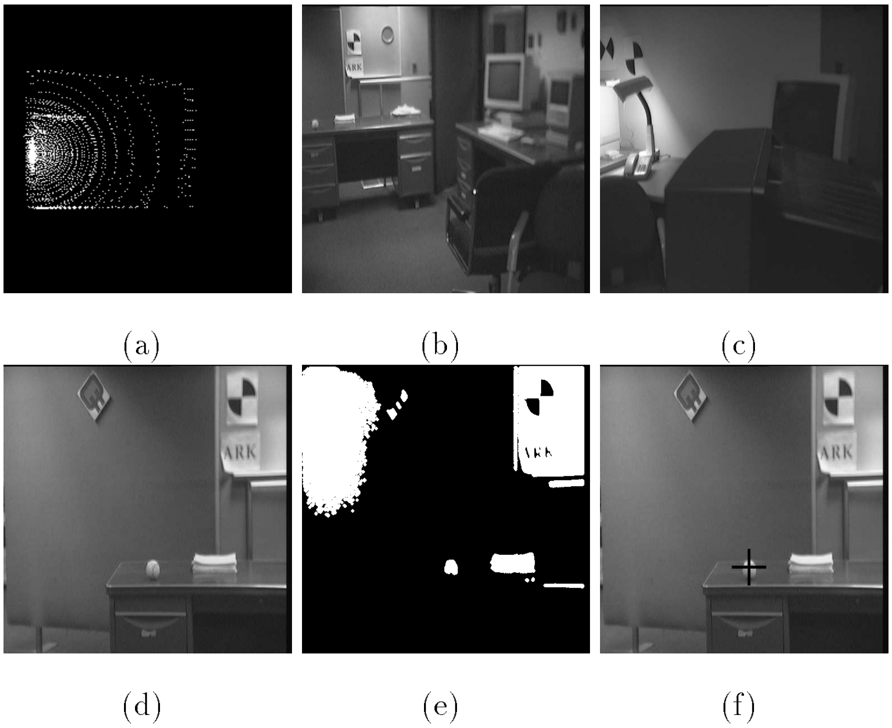


Figure 9:

(a) Sensed sphere from Laser Eye.

(b), (c), (d) The first three image sequences of the real experiment by using our planning strategy.

(e) The image of (d) after region growing.

(f) The result of the image analysis of (e), where the target is detected.

Note that the initial target distribution has a great influence on the performance of our sensor planning system. The reason is that the probability distribution is used to guide the sensor planning process. The more accurate is our knowledge about the initial distribution, the more efficient is our algorithm. We have performed experiments to examine the influence of the initial knowledge on the effectiveness of our sensor planning strategy. Experimental results are listed in Table 3 as the average number of actions needed to bring the ball into the the effective viewing volume of the camera for planning and non-planning strategies. For example, the data in the second column refers to the situation when the target is on table A. Then the number of actions needed to bring the ball into the effective volume of the camera is 1 when we give the table surface A a high probability distribution. The number of actions needed to bring the target into the effective volume of the camera by using Non-planning strategy is 8. The number of actions needed to bring the ball into the effective volume of the camera is 7 when we give the table surface C a high probability distribution (Note: the target is on A). Similar explanations can be applied to other columns of the table.

From the second and third row of Table 3, we can see that the planning strategy is much more efficient when our knowledge about the distribution is reasonable at the beginning. From the fourth row of Table 3 we can see that the performance of the planning strategy is not very satisfactory when we have the misleading information at the beginning.

Target Pos.	A	B	C	A or B	A or C	A, B or C
Plan$\left(\frac{correct}{knowledge}\right)$	(A)1	(B)1	(C)1	(AB)1.5	(AC)1.5	(ABC)2.5
No Plan	8	9	10	8.5	9	9
Plan$\left(\frac{incorrect}{knowledge}\right)$	(C)7	(A)8.5	(B) 11.5	(C)6	(B)9	(C)4.3

Table 3: The comparison of planning strategy and non-planning strategy

A second experiment is used to test the whole sensor planning strategy. The searcher is the ARK robot equipped with a camera that is different from the previous camera. The recognition algorithm is the same as that of the previous experiment, except that the threshold for the parameter values are different. Here we use $I_0 = 110$, $B_{min} = 170$ pixels, $B_{max} = 5500$ pixels, and $R_0 = 0.91$. The first viewing angle size is: $w_0 = h_0 = 30^o$. The effective range is, $N_0 = 1m$, $F_0 = 5.5m$. The outputs of the recognition algorithm are: Blob Size — 5478 pixels for N_0 and 171 pixels for F_0; Percentage — 92.37% for N_0 and 94.74% for F_0. Before the search process, we give the table top **B**, table top **C**, and the floor region **D** high probability of presence. The real position of the baseball is within the region **D**. When the probability of detecting the target is less then 0.0065, the robot is required to change the position.

Initially, the robot is at position $(1.6m, 2.9m)$ (P_1 of Figure 7). For this position, one angle size is sufficient. The sensed sphere constructed by the real laser at P_1 is shown in Figure 10.

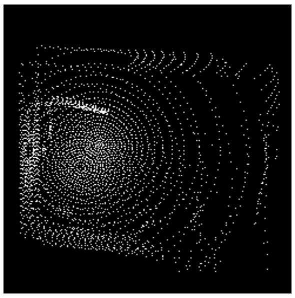


Figure 10:

The sensed sphere constructed by the robot.

Table 4 lists the actions selected by the algorithm and the corresponding detection probabilities.

Tilt	**Pan**	**Probability of Detection**
1.801100	0.000000	0.037403
2.315700	5.655361	0.036140
2.315700	4.948441	0.036654
1.286500	5.366405	0.006105

(a) (b) (c) (d)

Figure 11:

The four actions generated by the sensor planning algorithm. Images are shown for

(a) $tilt = 1.801100$ *and* $pan = 0.000000$.

(b) $tilt = 2.315700$, $pan = 5.655361$.

(c) $tilt = 2.315700$, $pan = 4.948441$.

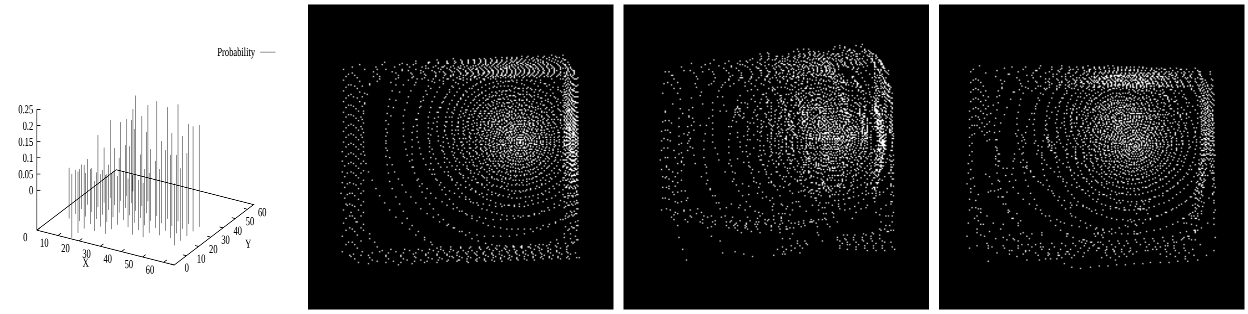
(d) $tilt = 1.286500$, $pan = 5.366405$.

Table 4: The actions selected by the algorithm and the corresponding detection probabilities.

Figure 11 shows the four images taken by the head according to the four actions listed above. We can see that the first action examines table **C**, the second and third actions examine table **B**. The fourth action simply points upward, and none of the specified regions is examined. The probability of detecting the target by the fourth action is very low, because none of the high probability regions is brought into the effective volume of the camera. No action is selected to check region **D**, because **D** is occluded by the office wall.

Since the probability of detection of the last action is 0.006105, which is less than 0.0065, the robot will quit the "where to look next" process and begin the "where to move next" process. The candidate positions are those for which $x = 0.1m$, $0.6m$, $1.1m$, ..., $6.1m$ and $y = 0.1m$, $0.6m$, $1.1m$, ..., $5.6m$. Figure 12 (a) shows the probability of the expected sensed sphere for each reachable candidate position. Among them, the position $(4.6m, 3.6m)$ (P_2 in

Figure 7(b) has the maximum probability. Figure 12 (b) shows the expected sensed sphere for position $(4.6m, 3.6m)$, where the white points represent the calculated intersecting points of the emanating lines from the camera center and the boundary of the search region. Figure 12(c) shows the sensed sphere when constrained by the solid cubes at this position, where the white points represent the calculated intersecting points of the emanating lines from the camera center and the environment after updating the radius using solid cubes. When the robot drives to the next position $(4.6m, 3.6m)$, it constructs the sensed sphere and repeats the search process. Figure 12(d) shows the sensed sphere constructed by the robot, where the white points represent the intersections of the real environment and the laser beams emanating from the camera center.



(a) (b) (c) (d)

Figure 12:

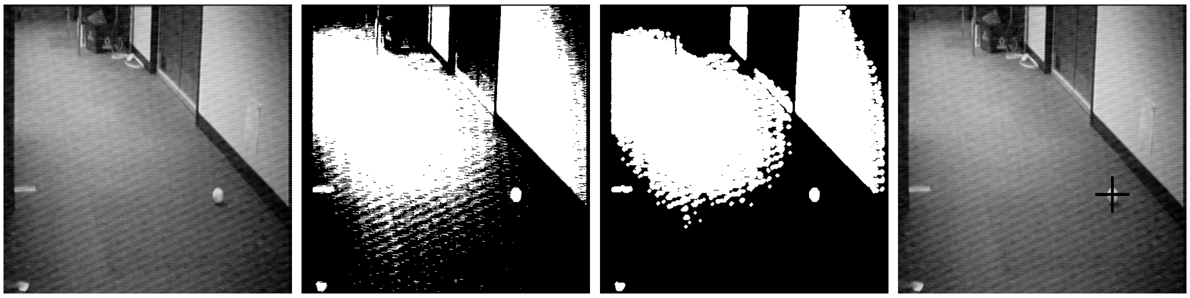
(a) The probability of presence for each reachable candidate position. The units of X, Y are decimeters.

(b) The expected sensed sphere at position $(4.6m, 3.6m)$ *by only considering the boundary of the search region.*

(c) The expected sensed sphere after constraining using the solid cubes.

(d) The real sensed sphere constructed at the position $(4.6m, 3.6m)$.

At the new position, the robot begins the search process again. The first action is: $tilt = 2.315700$ and $pan = 3.534601$. This action find the target. Figure 13 shows the image of this action and the image processing results.



(a) (b) (c) (d)

Figure 13:

(a) The image of the first action.

(b) The generated binary image.

(c) The image after the morphological operation.

(d) The final result: the target is detected (blob size is 372*, percentage is* 91.40%*).*

9 Conclusion

An interesting phenomenon of most past computer vision research is that the purely theoretical side of computer vision is too heavily emphasized. Consequently, many theoretical results cannot be applied to real situations, because they are obtained by making unrealistic assumptions. We think it is beneficial to put more emphasis on the "task oriented vision" framework as an alternative to the classic reconstructionist paradigm. For this new framework, the vision system is just one component of the agent embedded in the world, and it should provide only the minimum perceptual functionality the agent needs to perform whatever it is doing at the moment.

Object search is a typical task for this new framework. For this task, the vision module involved is object recognition. In order to find the target, the agent has to purposefully change the sensing parameters so as to bring the target into the field of view of the camera and to make the image of the target easily recognized by the available recognition algorithms. The study of the object search task may generate insight into existing vision problems. For example, the problem of object recognition may be less difficult when the image is intelligently grabbed by a controlled camera and the image analysis is interpreted within specific contexts. Research shows that the object recognition problem can be simpler if the camera is intelligently controlled [49]. Of course, the simplicity obtained in the image analysis phase is at the cost of the complexity encountered in the camera controlling phase. Much work has been done on image analysis. It is now time to study the problems related to the control of the camera. This is one reason why the sensor planning task for object search should be studied and comprises one of the motivations of this paper.

In this paper, we study the problem of sensor planning for object search from both the aspect of theory and that of practical implementation. By introducing the concept of a detection function for the object recognition algorithm and the concept of a probability distribution for the target within the search region, we formulate the sensor planning task as an optimization problem: the goal is to maximize the probability of detecting the target within a given time constraint. By analyzing the formulation of the problem, we obtain some important properties of this task and find that this task is NP-complete in general. In order to make the problem tractable, we propose to use the greedy strategy, and point out that under some special conditions this strategy may generate an optimal answer. By considering the detection ability of operations, we are able to decompose the enormous space of possible sensing actions into a limited finite set of actions that must be tried, thus greatly reducing the complexity in the action selection process. Various simulation experiments and real experiments have been performed based on our strategy. Experimental results show that our strategy is more efficient than those strategies with fixed action sequences (non-planning). It is interesting to note here that Wixson [51] has examined the "where to move next" problem in $2D$ situations by simulation and concluded that simpler model-free methods can find objects without significantly more effort than a map-based method. His result does not contradict our result because the probability distribution is not considered in his experiments.

Our theory has been applied using the ARK robot and the Laser Eye. Experiments so far have been successful as a proof of concept. A future direction is the design of a search strategy when the target is able to move. Another direction is the design of a search strategy when several robots are available.

References

[1] A. Abbott and N. Ahuja. Surface reconstruction by dynamic integration of focus, camera vergence, and stereo. In *Second International Conference on Computer Vision*, pages 532–543, Florida, USA, December 1988.

[2] A.L.Abbott. Selective fixation control for machine vision: A survey. In *IEEE Intervnational Conference on System, Man and Cybernetics*, pages 1–6, August 1991.

[3] Y. Aloimonous, I. Weiss, and A. Bandyopadhyay. Active vision. In *First International Conference on Computer Vision*, pages 35–54, London, England, June 1987.

[4] R. Bajcsy. Active perception vs. passive perception. In *Third IEEE Workshop on Vision*, pages 55–59, Bellaire, 1985.

[5] R. Bajcsy and M. Campos. Active and exploratory perception. *CVGIP: Image Understanding*, 56(1):31–40, 1992.

[6] A. Balke and A. Yuille. *Active Vision*. The MIT Press, Cambridge, Mass., 1992.

[7] D. Ballard. Animate vision. *Artificial Intelligence*, 48:57–86, 1991.

[8] D. H. Ballard. Task frames in visuo-motor coordination. In *Proceedings IEEE Workshop on Computer Vision: Representation and Control*, pages 3–10, Bellaire MI., 1985.

[9] D. H. Ballard and A. Ozcandarli. Eye fixation and early vision: Kinetic depth. In *Second International Conference on Computer Vision*, pages 524–531, Florida, USA, December 1988.

[10] R. Bolle, A. Califano, and R. Kjeldsen. Data and model driven foveation. Technical Report RC 15096, IBM Research Division, T.J. Watson Research Center, NY, October 1989.

[11] M. Brand. An eye for design: Why, where and how to look for causal structure in visual scenes. *SPIE.Intelligent Robots and Computer Vision*, 1825(XI):180–188, 1992.

[12] C. Brown. Issues in selective perception. In *11th International Conference on Pattern Recognition*, pages 21–30, 1992.

[13] R. Chin and C. Dyer. Model-based recognition in robot vision. *Computing Serveys*, 18(1):67–108, March 1986.

[14] Connel. *An Artificial Creature.* PhD thesis, AI Lab, MIT, 1989.

[15] C. K. Cowan and P. D. Kovesi. Automatic sensor placement from vision task requirements. *IEEE Transactions on Pattern analysis and Machine Intelligence*, 10(3):407–416, May 1988.

[16] S. Culhane and J. Tsotsos. An attentional prototype for early vision. In *Second European Conference on Computer Vision*, pages 551–560, Santa Margherita Ligure, Italy, May 1992.

[17] T. Dean. Sequential decision making for active perception. In *Image Understanding Workshop*, pages 889–894, Pennsylvania, USA, September 1990.

[18] T. D. Garvey. Perceptual strategies for purposive vision. Technical Report Technical Note 117, SRI International, 1976.

[19] J. Gibson. *The perception of the visual world.* Houghton-Mifflin, Boston, 1950.

[20] J. Gibson. *The Ecological Approach to Visual Perception.* Houghton-Mifflin, Boston, 1979.

[21] D. Hubel. *Eye, Brain and Vision.* Volume 22 of Scientific American Library W.H.Freeman and Company, 1988.

[22] P. Jasiobedzki, M. Jenkin, E. Milios, B. Down, and J. Tsotsos. Laser eye - a new 3d sensor for active vision. In *Intelligent Robotics and Computer Vision: Sensor Fusion VI. Proceedings of SPIE. vol. 2059*, pages 316–321, Boston, Sept. 1993.

[23] T. K., P. Allen, and R. Tsai. A survey of sensor planning in computer vision. Technical report, Comp. Sci. Dept., Columbia University, 1992.

[24] J. Kender and Freudenstein. What is a "degenerate" view? In *Image Understanding Workshop*, pages 589–598, California, February 1987.

[25] H. Kim, R. Jain, and R. A. Volz. Object recognition using multiple views. In *1985 IEEE Intervnational Conference on Robotics and Automation*, pages 28–33, July 1985.

[26] J. Koenderink and A. van Doorn. The internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32:211–216, 1979.

[27] B. O. Koopman. *Search and Screen: general principles with historical applications.* Pergaman Press, Elmsford, N.Y, 1980.

[28] E. Krotkov. Focusing. *International Journal of Computer Vision*, 1:223–237, 1987.

[29] E. Krotkov. *Active computer vision by cooperative focus and stereo.* Springer-Verlag, New York, 1989.

[30] K. Kutulakos and Dyer. Recovering shape by purposive viewpoint adjustment. In *Proceedings of Computer Vision and Pattern Recognition*, pages 16–22, Illinois,USA, June 1992.

[31] S. Li. Realizing active vision by a mobile robot. In *IEEE Workshop on Visual Motion*, pages 205–209, 1991.

[32] J. Maver and R. Bajcsy. How to decide from the first view where to look next. In *Proceedings of the DARPA Image Understanding Workshop*, September 1990.

[33] J. Maver and R. Bajcsy. Occlusions and the next view planning. In *IEEE 1992 International Conference on Robotics and Automation*, pages 1806–1811, April 1992.

[34] R. C. Nelson. Imtroduction: Vision as intelligent behavior: An introduction to machine vision at the university of rochester. *International Journal Computer Vision*, 7(1), 1991.

[35] S. Nickerson, M. Jenkin, E. Milios, B. Down, P. Jasiobedzki, A. Jepson, D. Terzopoulos, J. Tsotsos, D. Wilkes, N. Bains, and K. Tran. Ark: Autonomous navigation of a mobile robot in a known environment. In *Intelligent Autonomous Systems-3*, pages 288–296, Pennsylvania, USA, February 1993.

[36] R.A.Brooks and J.H.Connell. Asynchronous distributed control system for a mobile robot. In *Mobile Robots:Proc. SPIE 727*, California, 1987.

[37] R. Rimey and C. Brown. Where to look next using a bayes net: Incorporating geometric relations. In *Second European Conference on Computer Vision*, pages 542–550, Santa Margherita Ligure, Italy, May 1992.

[38] A. Shmuel and Werman. Active vision:3d from an image sequence. In *10th International Conference on Pattern Recognition*, pages 48–54, New Jersey, USA, June 1990.

[39] M. Swain and M. Stricker. Promising directions in active vision. Technical Report CS 91-27, Univ. Of Chicago Technical Report, 1991.

[40] K. Tarabanis, R. Y. Tsai, and P. K. Allen. Automatic sensor placement from vision task requirements. *CVGIP: Image Understanding*, 10.(3):340–358, May 1988.

[41] J. Tsotsos. A complexity level analysis of immediate vision. *International Journal Computer Vision*, 4(1):303–320, 1988.

[42] J. Tsotsos. Analyzing vision at the complexity level. *The behavioral and brain science*, 13:423–469, 1990.

[43] J. Tsotsos. 3d search strategy. *Internal ARK Working Paper, Department of Computer Science, Univrsity of Toronto*, 1992.

[44] J. Tsotsos. Active verses passive perception, which is more efficient? *IJCV*, 1992.

[45] J. Tsotsos. On the relative complexity of active vs. passive visual search. *International Journal of Computer Vision*, 7:127–141, 1992.

[46] W.E.Grimson. Sensing strategies for disambiguating among multiple objects in known poses. *IEEE Journal of Robotics and Automation*, RA-2(4):196–213, December 1986.

[47] P. Whaite and F. P. Ferrie. From uncertainty to visual exploratory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1308–1049, 1991.

[48] D. Wilkes. Active object recognition. Technical report, Department of Computer Science, University of Toronto, 1994.

[49] D. Wilkes and J. Tsotsos. Active object recognition. In *Proceedings of Computer Vision and Pattern Recognition*, pages 136–141, Illinois, USA, June 1992.

[50] L. Wixson and D. Ballard. Using intermediate object to improve the efficiency of visual search. *International Journal on Computer Vision*, 18(3):209–230, 1994.

[51] L. E. Wixson. *Gaze Selection for Visual Search.* PhD thesis, Department of Computer Science, University of Rochester, 1994.

[52] Y. Ye. *Sensor planning in 3D object search.* PhD thesis, Department of Computer Science, University of Toronto, Toronto, Ontario, Canada M5S 1A4, 1996.

[53] Y. Ye and J. K. Tsotsos. The detection function in object search. In *Proceedings of the fourth international conference for young computer scientist*, pages 868–873, Beijing, 1995.

[54] Y. Ye and J. K. Tsotsos. Where to look next in 3d object search. In *1995 IEEE International Symposium for Computer Vision*, Florida, U.S.A, November 19-21 1995.

[55] Y. Ye and J. K. Tsotsos. Sensor planning in 3d object search: its formulation and complexity. In *The 4th International Symposium on Artificial Intelligence and Mathematics*, Florida, U.S.A, January 3-5 1996.

[56] S. Yi. Automatic sensor and light source positioning for machine vision. In *10th International Conference on Pattern Recognition*, pages 55–59, New Jersey, USA, June 1990.

[57] J. Zheng, F. Kishino, Q. Chen, and S. Tsuji. Active camera controlling for manipulation. In *Proc. Computer Vision and Pattern Recognition*, Washington, 1991.

[6] [29] [21] [19] [20] [42] [45] [41] [34] [46] [39] [10] [36] [33] [31] [57] [8] [30] [49] [12] [56] [38] [24] [9] [17] [37] [16] [4] [1] [2] [25] [26] [5] [47] [11] [7] [23] [27] [35] [43] [44] [42] [49] [48] [54] [3]