A realistic method for real-time obstacle avoidance without the Calculation of Cspace Obstacles

Yongji Wang, Han Liu, Qiuming Tao
Laboratory for Internet Technologies
Institute of Software
Chinese Academy of Sciences
P. O. Box 8718, Beijing 100080, China
E-Mail: {ywang,liuhan,qmtao}@intec.iscas.ac.cn
Tel. 0086 10 62581294, Fax. 0086 10 62632340

***Abstract*— An important concept proposed in the early stage of the robot path planning field is the shrinking of a robot to a point and meanwhile the expanding of the obstacles in the workspace as a set of new obstacles. The resulting grown obstacles are called the Configuration Space (Cspace) obstacles [24, 23]. The find-path problem is then transformed into that of finding a collision-free path for a point robot among the Cspace obstacles. However, the research experiences obtained so far have shown that the calculation of Cspace obstacles is very hard when the following situations occur: 1. both the robot and obstacles are not polygons and 2. the robot is allowed to rotate. This situation is getting even worse when the robot and obstacles are three dimensional (3D) objects with various shapes [43, 54, 3]. Obviously, a direct path planning approach without the calculation of the Cspace obstacles is strongly needed. This paper presents such a new real-time robot path planning approach which, to our best knowledge, is the first one in the robotic community. Motivated by the practical requirements of obstacle representation, a generalized constrained optimization problem (GCOP) with not only intersection but also union operations was proposed and a mathematical solution to it was developed [47]. This paper inherits the fundamental ideas of inequality and optimization techniques from the previous work [45-47] and converts the real-time obstacle avoidance problem into a semi-infinite constrained optimization problem with the help of the mathematical transformation given in [47]. This leads to an efficient method for the real-time obstacle avoidance. Simulation results in 3D space have been presented to show its merits.**

***Index Terms*— autonomous underwater vehicle, mathematical transformation, non-linear programming, obstacle avoidance, path planning, real-time system, robotics, semi-infinite constrained optimization, subsea vehicle.**

I. Introduction

An autonomous robot (for example, an Unmanned Underwater Vehicle extensively used for the inspection, maintenance and repair of offshore structures in both deep and shallow water, See Fig. 1) is a typical real-time system which has strict time constraints [22,45,46,30,31]. Such a real-time system normally consists of the following tasks: 1. dynamic plant identification; 2. low level control law design; 3. real-time sensor fusion or computer vision signal processing for environmental map building; 4. path planning based on the results obtained in step 3; and 5. high level mission management or control [54,29,,37,42]. In this paper we concentrate on the path planning task for the vehicle through a cluttered environment, such as around a subsea structure. Although seemingly trivial, it has proved notoriously difficult to find techniques which work efficiently in the presence of multiple obstacles. The problem is well known as a find-path problem which can be described as follows: Given a subsea vehicle with an initial configuration, a goal configuration, and a set of obstacles located in the workspace, finds a continuous, collision-free path from the initial configuration to the goal configuration for the vehicle.

Obstacle avoidance has been an active field during the past twenty years. A significant and varied effort has been made on this seemingly simple, but, in fact, very complicated problem [5, 9, 10, 17-20,23-25,27-31,44-56]. Various methods for dealing with the basic find-path problem and its extensions, such as

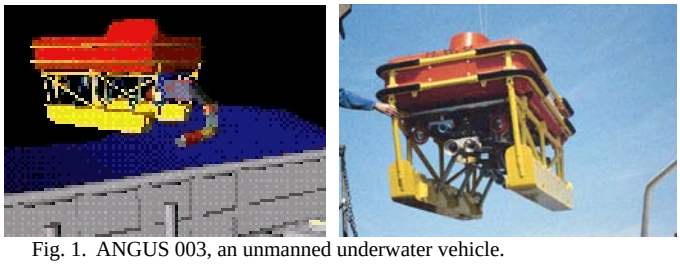

Fig. 1. ANGUS 003, an unmanned underwater vehicle.

Vgraph [24], Voronoi diagram [23], exact cell decomposition [5], approximate cell decomposition [23], potential field approach [20, 19], and optimization-based approach [45-47] have been developed. A systematic discussion on these methods can be found in references [23, 54].

Historically the Constrained Optimization and Constructive Solid Geometry (COCSG) method was first proposed in [45]

and the assumption made in that paper was that each obstacle in the workspace can be represented by one inequality. The mathematical foundations for the Constructive Solid Geometry (CSG), the Boolean operations and the approximation techniques were developed to represent the free space of the robot as a set of inequalities in [45, 34, 4, 1, 7]. The fundamental ideas used include: 1. the free Cspace of the robot is represented as a set of inequality constraints using the vehicle configuration variables; 2. the goal configuration is designed as the unique global minimum point of the objective function and the initial configuration is treated as the start point for the spatial search; 3. the numerical algorithm developed for solving nonlinear programming problem is applied to solve the robot motion planning problem and every immediate point generated in this way guarantees that it is in the free Cspace and therefore is collision free.

In any robot path planning method, robot and obstacle representation is the first thing to be considered. Obstacle avoidance problem essentially deals with how to move a 3D object (robot) among another set of 3D objects (obstacles), satisfying various constraints [44,52,53,38,55]. There are many reasons for having so many different approaches developed. For example, the different assumptions made on the shapes of the robot and obstacles, on the constraints imposed by the mechanical structure of the robot all contribute to them. The important thing for judging the reality of an approach is whether the problem formulation is realistic.

An important concept proposed in the early stage of robot path planning field is the shrinking of the robot to a point and meanwhile the expanding of the obstacles in the workspace as a set of new obstacles. The resulting grown obstacles are called the Configuration Space (Cspace) obstacles. The find-path problem is then transformed into that of finding a collision-free path for a point robot among the Cspace obstacles. This idea was first popularized by Lozano_Perez [24] in the Artificial Intelligence Lab, MIT as a basis of the spatial planning approach for the find-path and find-place problems and then extended by Latombe [23] as a basis of all motion planning approaches suitable for a point robot. However, the research experiences obtained so far have shown that the calculation of Cspace obstacles is a very hard work when the following situations occur. 1. Both the robot and obstacles are not polygons and 2. The robot is allowed to rotate. This situation is getting even worse when the robot and obstacles are 3D objects with various shapes [43, 54, 3, 14, 16]. It can be observed that all of the aforementioned approaches developed so far are based on the Cspce obstacles; therefore they are only suitable for the unrealistic situation. Obviously, a direct path planning approach without the calculation of the Cspace obstacles is strongly needed. This paper presents such a new real-time robot path planning approach which, to our best knowledge, is the first one in the robotic community.

Motivated by the practical requirements of obstacle representation, a generalized constrained optimization problem (GCOP) with not only intersection but also union operations was proposed and a mathematical solution to it was developed in the reference [47]. This paper inherits the fundamental ideas of inequality and optimization techniques from the previous work [45-47]. AS will be shown later, the ideas of inequality and optimization proposed by Wang [45-47] are still useful and will be used in this paper, enabling us to develop a new approach without the need to calculate the Cspace obstacles. It converts the real-time obstacle avoidance problem into a semi-infinite constrained optimization problem with the help of the mathematical transformation given in [45]. This leads to an efficient method for the real-time obstacle avoidance. Therefore, the work reported here is, in certain way, a natural extension of the previous work.

In this paper we will present the principle of the new real-time robot path planning method. The points outside the obstacles in the 3D workspace are represented by implicit inequalities and the points on the boundary of a 3D robot are expressed in the form of a parametric function. A sufficient and necessary condition for a collision free path for the robot and the obstacles is then derived in the form of a set of inequalities that lead to the use of efficient search algorithms.

The rest of this paper is organized as follows. Section II gives a brief description of the set operation, inequality constraints and the formulations for optimization theory. In particular, a previously-developed, generalized constrained optimization with union operations and the mathematical translation needed for its solution are also presented in this section, which forms the basis for the following sections. In section III, obstacle and robot presentation method is presented. The implicit function for representing the outside of the obstacles as inequalities and the parametric function for representing the surface points on the 3D robot as equalities are developed. In section IV, we investigate how to convert the robot path planning problem into a semi-infinite constrained optimization problem and a sufficient and necessary condition for a collision-free path for a 3D robot among a set of 3D obstacles is developed. Simulation results are presented in section V. Finally conclusions are given in section VI.

II. Set Operation, Equality And Inequality Constraints In Optimization Problems

In this section, set theory will be used to explain the logic relationship between the constraints in the optimization problems and a solution technique developed in [47] is also briefly introduced. This forms the basis for the solution to the robot obstacle avoidance problem. Here we can only give a brief introduction to the various optimization problems because the emphasis of this paper will be focused on how to convert the path planning problem into a standard semi-infinite constrained optimization problem, rather than on the details of the nonlinear programming theory which are easily found in [11,13,26,32,33,40]. The essential part of the mathematical transformation which can transfer a set of inequalities with union operations into one inequality is also introduced here.

A. Set operations

Definition 1: Let A and B be sets. The **Union** of the sets A

and B, denoted by A∪B, is the set that contains those elements that are either in A or in B, or in both. Mathematically $A \cup B = \{\boldsymbol{x} \mid \boldsymbol{x} \in A \vee \boldsymbol{x} \in B\}$, where $\boldsymbol{x}$ is an element belonging to the union of A and B.

Definition 2: Let A and B be sets. The **Intersection** of the sets A and B, denoted by A∩B, is the set containing those elements in both A and B. Mathematically $A \cap B = \{\boldsymbol{x} \mid \boldsymbol{x} \in A \wedge \boldsymbol{x} \in B\}$, where $\boldsymbol{x}$ is an element belonging to the intersection of A and B.

Definition 3: Let A and B be sets. The **Difference** of the sets A and B, denoted by A-B, is the set that contains those elements that are in A but not in B. The difference of A and B is also called the complement of B with respect to A. Mathematically $A\text{-}B = \{\boldsymbol{x} \mid \boldsymbol{x} \in A \wedge \boldsymbol{x} \notin B\}$, where $\boldsymbol{x}$ is an element belonging to the difference of A and B.

Definition 4: Let U be the universal set. The **complement** of the set A, denoted by $\overline{A}$, is the complement of A with respect to U. In other words, the complement of the set A is U-A. Mathematically $\overline{A} = \{\boldsymbol{x} \mid \boldsymbol{x} \notin A\}$, where $\boldsymbol{x}$ is an element belonging to the complement of A.

More details on set identities, see the reference [36].

B. Optimization problems

Standard optimization theory (SOT) concerns the minimization or maximization of a function subject to different types of constraints (equality or inequality) [11,13]. SOT has a very sound theoretical background and many applications in various engineering fields such as Computer Vision, Automatic Control, Computer Aided Design, Structure Optimal Design, and Transportation Engineering for finding the roots of a set of non-linear equations, for detecting the intersections of solid objects, for fitting a curve, and so on [2,6,8,39].

Optimization techniques have now become a mature subject. There are mainly four different types of optimization problem: *Linear Programming*, *Unconstrained Problems*, *Constrained Problems* and *Semi-infinite Constrained Problems,* as shown in Table I [13,15,40,57]. The last three parts together comprise the subject of *Non-linear Programming*.

C. Standard Constrained Optimization (SCO)

The standard constrained optimization problem investigated in all the literature available can be described as follows: Find an optimal point $\boldsymbol{x}^*$ which minimizes the function

$$f(\boldsymbol{x}) \tag{1}$$

subject to:

$$\begin{aligned} & g_i(\boldsymbol{x}) = 0,\ i = 1, 2, \ldots\ldots, m1 \\ & g_j(\boldsymbol{x}) \le 0,\ j = m1+1, \ldots\ldots, m2 \\ & \boldsymbol{x}_l \le \boldsymbol{x} \le \boldsymbol{x}_u \end{aligned} \tag{2}$$

where $m1$ and $m2$ are integers, $\boldsymbol{x}$ is an n-dimensional vector of unknowns, $\boldsymbol{x} = (x_1, x_2, \ldots\ldots, x_n)$, and $f, g_i, i = 1, 2, \ldots\ldots, m1$ and $g_j, j = m1+1, \ldots\ldots, m2,$ are real-valued functions of the variables $(x_1, x_2, \ldots\ldots, x_n)$. $\boldsymbol{x}_l = (x_{l1}, x_{l2}, \ldots\ldots, x_{ln})$ and $\boldsymbol{x}_u = (x_{u1}, x_{u2}, \ldots\ldots, x_{un})$ are the lower and upper bounds of $\boldsymbol{x}$, respectively. The function f is the objective function, and the equations and inequalities of (2) are constraints.

D. Generalized Constrained Optimization (GCO)

It is important to note that although not explicitly stated in the literature available, the set operation among the constraints g_i, g_j and $\boldsymbol{x}$ in (2) are ***intersection***. That is, their relationships imply the following intersection operations: $\{\boldsymbol{x} \mid g_1(\boldsymbol{x})=0 \cap \ldots\ldots \cap g_{m1}(\boldsymbol{x})=0 \cap g_{m1+1}(\boldsymbol{x}) \le 0 \cap \ldots\ldots\ldots \cap g_{m2}(\boldsymbol{x}) \le 0 \cap x_{l1} \le x_1 \le x_{u1} \cap \ldots\ldots \cap x_{ln} \le x_n \le x_{un}\}$. The problem described by (1) and (2) was named as the standard constrained optimization problem (SCOP) in [47] due to the fact that the constraints are of the set intersection operations. However, the work reported in [47] has shown that a practical robot path planning problem can be cast as a generalized constrained optimization problem of the following form:

Find an optimal point $\boldsymbol{x}^*$ which minimizes the function

$$f(\boldsymbol{x}) \tag{3}$$

subject to:

$$\begin{aligned} & \{\boldsymbol{x} \mid g_1(\boldsymbol{x})=0 \cap \ldots\ldots \cap g_{m1}(\boldsymbol{x})=0 \cap g_{m1+1}(\boldsymbol{x}) \le 0 \cap \ldots\ldots\ldots \cap g_{m2}(\boldsymbol{x}) \le 0 \cap \\ & [h_{1,1}(\boldsymbol{x}) \le 0 \cup \ldots\ldots \cup h_{1,km}(\boldsymbol{x}) \le 0] \cap [h_{2,1}(\boldsymbol{x}) \le 0 \cup \ldots\ldots \cup h_{2,km}(\boldsymbol{x}) \le 0] \cap \ldots\ldots \\ & \cap [h_{m,1}(\boldsymbol{x}) \le 0 \cup \ldots\ldots \cup h_{m,km}(\boldsymbol{x}) \le 0] \cap \{x_{l1} \le x_1 \le x_{u1} \cap \ldots\ldots \cap x_{ln} \le x_n \le x_{un}\}\} \end{aligned} \tag{4}$$

where $m1, m2, m, k1, k2, \ldots, km$ are all integers, $h_{i,j}(\boldsymbol{x})$ are real-valued functions of the variables $\boldsymbol{x}$, and $A_i = \{\boldsymbol{x} \mid h_{i,1}(\boldsymbol{x}) \le 0 \cup \ldots\ldots \cup h_{i,km}(\boldsymbol{x}) \le 0, i = 1, \ldots\ldots, m\}$ are the union operations. The problem described by (3) and (4) was named as the generalized constrained optimization problem (GCOP) because the constraints have both the intersection and the union operations. This generalized form will play an important role in the domain of real time robot path planning without the need to calculate the Cspace obstacles, as shown later in this paper.

The development of an algorithm for the solution to GCOP is important. If it is possible to transfer the constraints (4) with both the intersection and the union operations into the constraint (2) with only the intersection operation by the set identities, then the generalized problem is trivial because the algorithms developed for solving the COP can be directly applied to the GCOP. Unfortunately this is impossible [36,47].

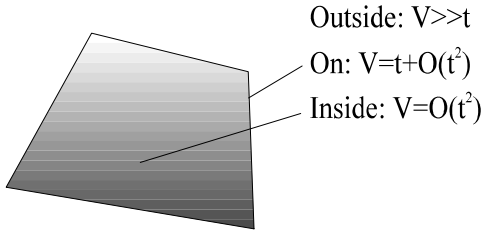

Fig. 2. Illustration of V as a function of a point $\boldsymbol{x}$ in n-dimensional space.

There are two ways to deal with the difficulty. The first is to develop some new algorithms which can directly deal with the GCOP rather than adopting the algorithms available for the COP. This seems possible. However we have not tried along that line. The second way is based on the idea of devising a mathematical transformation which is able to convert the *km* union operations $A_i=\{\boldsymbol{x} \mid h_{i,1}(\boldsymbol{x})\le 0 \cup \ldots\ldots \cup h_{i,km}(\boldsymbol{x})\le 0, i=1,\ldots,m\}$ into *one* new inequalities $A_i=\{\boldsymbol{x} \mid H_i(\boldsymbol{x})\le 0,\ i=1,2,\ \ldots,\ m\}$ for any point $\boldsymbol{x}$. As a result, the algorithms developed for the SCOP can be directly applied to the GCOP.

E. A mathematical solution to convert a set of inequalities with the union operations into one inequality

Here, we present a mathematical transformation which is able to realize the second idea in section D. Suppose there are *m* inequalities $h_i(\boldsymbol{x})<0,\ i=1,2,\ldots,m,$ combined with union operations as given in (5). From mathematical point, set operation (5) represents the point set of the inside for a generalized *n* dimensional object. In a 3D space, set operation (5) may be explained as representing the set of all the inside points for an object whose surface is represented by *m* continuous equations $h_i(\boldsymbol{x})=0,\ i=1,2,\ldots,m:$

$$\{\boldsymbol{x} \mid h_1(\boldsymbol{x})<0 \cap \ldots\ldots \cap h_m(\boldsymbol{x})<0\} \tag{5}$$

For each function $h_i(\boldsymbol{x})$, a new function of the following form is constructed ($\boldsymbol{x}$ is omitted for simplicity):

$$v_i=(h^2{}_i+t^2)^{1/2}+h_i \tag{6}$$

where *t* is a small, positive real number and satisfies $t<<1$. Note that v_i is the function of a point $\boldsymbol{x}$ and *t*. For the whole object, a function *V* of the following form is also constructed

$$V = v_1 + v_2 + \ldots + v_m = \sum_1^m v_i \tag{7}$$

Note the symbol "+" in (6) and (7) represents simply the arithmetic addition. Now let us examine the properties of the two transformations from h_i to v_i and from v_i to *V* in greater detail. First, the function v_i is **always positive** for any point $\boldsymbol{x}$ and any constant *t*, i.e. $v_i>0$ always holds, and second, it is an **increasing function** of h_i which suggests that the value of v_i at the points where $h_i>0$ is much larger than the value at the points where $h_i<0$. If $t<<1$, v_i can be approximately represented as

$$v_i=\begin{cases} 2h_i+O(t^2)>>t>0, & for \quad h_i>0 \\ \approx t, & for \quad h_i=0 \\ \approx O(t^2), & for \quad h_i<0 \end{cases} \tag{8}$$

where $O(t^2)$ represents a very small positive number with the order of t^2 for $t<<1$. (8) Indicates that except for the points located at the vicinity of the surface $h_i=0$, v_i is large compared with *t* when $h_i>0$, and small compared with *t* when $h_i<0$.

From Fig. 2 we can see that for the points located inside the object and in the vicinity of $h_i=0$, the value of all other functions h_j $(j=1,2,\ldots,m$ and $j\neq i)$ is less than zero, which leads to v_i in the order of $O(t^2)$. Substituting (8) into (7) gives

$$V=\begin{cases} >>t, & for \ (h_1>0)\cup(h_2>0)\cup\ldots\ldots\cup(h_m>0); \\ \approx t+O(t^2), & for \ (h_1=0\cap h_j\le 0, j=2,3,\ldots,m)\cup \\ & \quad (h_2=0\cap h_j\le 0, j=1,3,4,\ldots,m)\cup\ldots\ldots\cup(h_m=0\cap h_j\le 0, j=1,2,\ldots,m-1) \\ \approx O(t^2), & for \ (h_1<0)\cap(h_2<0)\cap\ldots\ldots\cap(h_m<0) \end{cases} \tag{9}$$

Consequently, from (9) we can observe that the function *V* is small, $\approx O(t^2)$, compared with *t* when all the h_i are sufficiently negative, i.e. at those points which are inside the object, $V>>t+O(t^2)$ at the set of outside points of the object where at least one of the h_i is greater than *t*, and $V\approx t$ in the vicinity of the boundaries of the object. Fig. 2 illustrates this situation.

Now let us consider the situation when $t\to 0$, which will help us to understand better why construction functions (6) and (7) are used here as the mathematical transformation. As $t\to 0$, v_i tends to

$$v_i=\begin{cases} >0, & for \quad h_i>0 \\ =0, & for \quad h_i\le 0 \end{cases} \tag{10}$$

It is well-known that the sum of two positive values is positive, the sum of a positive value and a zero is positive, and the sum of two zeroes is zero. Thus the addition operation of v_i in (7) corresponds to the union operation if we treat a positive value as a logic value "1" and a zero as a logic value "0". Thus we have

$$V=\begin{cases} >0, & for \quad h_i>0 \\ =0, & for \quad h_i\le 0 \end{cases} \tag{11}$$

This property is very attractive and is particularly suitable for our requirements and it indicates that when $t=0$ the **sufficient and necessary condition** for a point $\boldsymbol{x}$ which falls into the outside of the object is that

$$V=\sum v_i>0 \tag{12a}$$

Note that the standard form for constraints in optimization problem (1) and (2) is **less than or equal to** zero. Note that although (12a) may change to the form

$$-V=-\sum v_i<0 \tag{12b}$$

It does not allow the condition "equal to zero". However, it is not desirable for robot path planning to have the path too close to the obstacles. A small positive value Δv can be introduced to control the distance of the allowed path to the obstacle. If the following inequality is satisfied by a point $\boldsymbol{x}$

$$V = \sum v_i \geq \Delta v \text{ or } \Delta v - \sum v_i \leq 0 \quad (13)$$

then this point must be outside the obstacle determined by (5). If $\Delta v \to 0$, the boundary determined by $\Delta v - \sum v_i \leq 0$ tends to be the surface of the obstacle. In the experiments which follow, we generally chose $t=0$ and $\Delta v = 0.0001$.

In summary, we have the following result:

Theorem 1: *If the outside and the surface of an object is determined by* $(h_1 \geq 0 \cup h_2 \geq 0 \cup \ldots \cup h_m \geq 0)$, *then its outside and surface can also be determined by the inequality* $\Delta v - \sum v_i \leq 0$ *as the small positive value* $\Delta v \to 0$. *In other words, the satisfaction of the inequality* $\Delta v - \sum v_i \leq 0$ *for a point* $\boldsymbol{x}$ *guarantees that this point falls outside the object.*

Mathematically the following set operation results can be obtained:

$$\{\boldsymbol{x} / (g_1 \geq 0 \cup g_2 \geq 0 \cup \ldots\ldots \cup g_m \geq 0) = \{\boldsymbol{x} / (\Delta v - \sum_1^m (\sqrt{g^2{}_i} + g_j) \leq 0)\} \quad (14a)$$

$$\{\boldsymbol{x} / (g_1 \leq 0 \cup g_2 \leq 0 \cup \ldots\ldots \cup g_m \leq 0)\} = \{\boldsymbol{x} / (\Delta v - \sum_1^m (\sqrt{g^2{}_i} - g_j) \leq 0)\} \quad (14b)$$

A direct conclusion drawn from ***Theorem 1*** is that a GCO problem can be converted into a SCO problem by the transformations (6) and (7).

F. Semi-Infinite Constrained Optimization (SICO)

The semi-infinite constrained optimization problem is to find the minimum of a semi-infinitely constrained scalar function of several variables $\boldsymbol{x}$ starting at an initial estimate $\boldsymbol{x}_s$. This problem is mathematically stated as:

Minimize

$$f(\boldsymbol{x}), \boldsymbol{x} \in \Re^n, \quad (15)$$

subject to:

$$g_i(\boldsymbol{x}) = 0, i = 1, 2, \ldots\ldots, m1$$
$$g_j(\boldsymbol{x}) \leq 0, j = m1+1, \ldots\ldots, m2$$
$$\Phi_k(\boldsymbol{x},\boldsymbol{v}) \leq 0, k = 1, 2, \ldots\ldots, n$$
$$\boldsymbol{x}_l \leq \boldsymbol{x} \leq \boldsymbol{x}_u \text{ for all } \boldsymbol{v} \in \Re^2 \quad (16)$$

where $\Phi_k(\boldsymbol{x},\boldsymbol{v})$ is a continuous function of both $\boldsymbol{x}$ and an additional set of variables $\boldsymbol{v}$. The variables $\boldsymbol{v}$ are vectors of at most length two. The aim is to minimize $f(\boldsymbol{x})$ so that the constraints hold for all possible values of $\Phi_k(\boldsymbol{x},\boldsymbol{v})$. Since it is impossible to calculate all possible values of $\Phi_k(\boldsymbol{x},\boldsymbol{v})$, a region must be chosen for v over which to calculate an appropriately sampled set of values. x is referred to as the unknown variable and $\boldsymbol{v}$ as the independent variables.

The difference between the SCO and SICO is obvious. The latter allows an extra set of independent variables $\boldsymbol{v}$ to be included in the constraints rather than only the unknown variable x, as is the case in the former. Note that the former is a special case of the latter, and the latter can not be converted into the former. For more details, see [37].

The procedure for solving such a SICO with nonlinear constraints is as follows:

(a) assign an initial guess point for x and a region for $\boldsymbol{v}$

(b) use a search algorithm to find the optimum solution $\boldsymbol{x}^*$ and the corresponding minimum objective function $f(\boldsymbol{x}^*)$

In the subsequent sections we will gradually illustrate that the 3D path planning problem without the calculation of Cspace obstacles can be converted into a standard semi-infinite constrained optimization problem.

III. 3D Space Obstacle And Robot Representation By Set Theory And Inequality

Obviously the first thing for robot path planning is to give each of the objects including obstacles and robot in the workspace a mathematical representation. Motivated by the practical requirements of obstacle representation, a 3D object will be categorized into **two groups** according to the number of the set operations needed for representing its inside. Now let us consider this issue and see how it affects the problem formulation.

A. Obstacle classification and representation by set theory and inequality

Let $\boldsymbol{X}=(x, y, z)$ denote a **point** in **3D space.** The inside of an **obstacle** in 3D space is a set of infinite points. A 3D obstacle divides the space into three **parts**: the **inside**, the **outside**, and the **boundary**. Let us consider the mathematical representation of a 3D obstacle from the viewpoint of set theory based on the set operation number used for its inside representation. First we introduce some definitions.

Definition 5: Free space of an obstacle: The set of points on or outside of the surface of a 3D obstacle is said to be its free space.

Definition 6: First group of obstacles: if the inside of a 3D obstacle can be represented by only ***one*** implicit function inequality, the obstacle is said to be in the first group.

Let A denote the set of all points inside an obstacle in the first group, and $h(\boldsymbol{X})=0$ denote the boundary, then according to definition 6, we have:

$$A=\{\boldsymbol{X} / h(\boldsymbol{X}) < 0\} \quad (17)$$

then $\overline{A}$, the complement of the set A, is the free space of the obstacle. Thus we have

$$\overline{A} = \{\boldsymbol{X} / h(\boldsymbol{X}) \geq 0\} \quad (18)$$

obviously the free space of an obstacle in the first group can

be represented by only **one** inequality. A simple example is illustrated in Fig. 3. Some examples of the obstacles in this group in 3D space include spheres, ellipsoids, torus, superellipsoids and so on [1,2,4,12,34,45].

Definition 7: Second group of obstacles: if the inside of an obstacle must be represented by *m (m is **more than one**)* implicit function inequalities with the intersection operations, the obstacle is said to be in the second group.

Let A denote the set of all points inside an obstacle in the second group, according to definition 7 we have

$$A = \{\boldsymbol{X} \,/\, h_1(\boldsymbol{X})<0 \cap \ldots\ldots \cap h_m(\boldsymbol{X})<0\} \quad (19)$$

then, the complement of the set B, is the free space of the obstacle. Thus we have:

$$\overline{A} = \{\boldsymbol{X} \,/\, h_1(\boldsymbol{X})\geq 0 \cup \ldots\ldots \cup h_m(\boldsymbol{X})\geq 0\} \quad (20)$$

Comparing (20) with (14a), we can conclude from

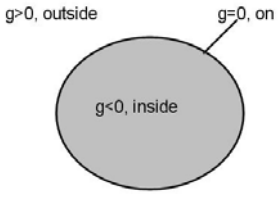

Fig. 3. First group of obstacles: inside and outside of an obstacle described by **one** inequality.

Theorem1 that the free space of an obstacle in the second group can also be represented by only **one** inequality.

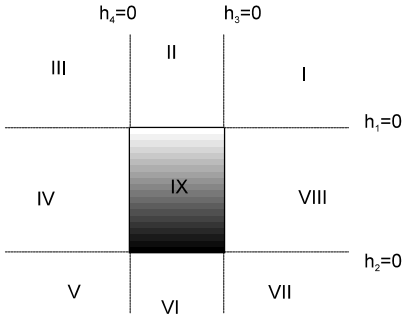

Fig. 4. Second group of obstacles: inside and outside of an obstacle described by **more than one** implicit function inequalities

For a deeper understanding of the set identities involved in (20) and the physical meaning, we use a simple example in 2D for illustration (Fig. 4). Generally speaking, the inside of the obstacle in this group is the intersection of *m (m>1)* regions A_i where each A_i is defined by an inequality $A_i=\{\boldsymbol{X}/h_i(\boldsymbol{X})<0\}$. In Fig. 4 the obstacle is a rectangle whose inside is surrounded by four lines $h_1=x-a=0$, $h_2=-x-a=0$, $h_3=y-b=0$, and $h_4=-y-b=0$, where a and b are positive values. $A_i=\{(x,y)/h_i<0\}$. The inside of the rectangle (Region IX in Fig. 4) is represented by the following set of inequalities with intersection operation:

$$(A_1 \cap A_2 \cap A_3 \cap A_4)=\{(x,y) \,/\, h_1<0 \cap h_2<0 \cap h_3<0 \cap h_4<0\} \quad (21)$$

Thus the free space (outside and boundary) of the rectangle (Region I ∪ II ∪ III ∪ IV ∪ V ∪ VI ∪ VII ∪ VIII and the boundary in Fig. 4) must satisfy the following relationship:

$$\overline{(A_1 \cap A_2 \cap A_3 \cap A_4)} = \overline{A_1} \cup \overline{A_2} \cup \overline{A_3} \cup \overline{A_4} \quad (22)$$

Since $A_i=\{(x,y)/h_i(x,y)<0\}$ and $\overline{A_i}=\{(x,y)/h_i(x,y)\geq 0\}$, thus the free space of the rectangle is represented as

$$\overline{(A_1 \cap A_2 \cap A_3 \cap A_4)} = (h_1\geq 0) \cup (h_2\geq 0) \cup (h_3\geq 0) \cup (h_4\geq 0) \quad (23)$$

Obviously (23) is a special case of (20).

From the discussion presented so far in this section we can have the following conclusion:

Theorem 2: *The free 3D space of an obstacle either in the first group or in the second group can be represented by one and only one inequality.*

Suppose a workspace is full of n obstacles that are either in the first group or in the second group. Suppose the inside for the *ith* obstacle is represented by

$$A_i=\{\boldsymbol{X}/g_i(\boldsymbol{X})<0\}, \quad i=1,2,\ldots\ldots,n \quad (24)$$

then the inside points determined by the n obstacles is described by the union operation: $A_1 \cup A_2 \cup \ldots \cup A_n$. The complement of the forbidden region is the free space, and can be represented by the complement operation: $\overline{(A_1 \cup A_2 \cup \ldots\ldots \cup A_n)}$. So the free space (on or outside the boundary surface) of the n obstacles can be finally represented as a set of inequalities with the following intersection operation:

$$\overline{(A_1 \cup A_2 \cup \ldots\ldots \cup A_n)} = \overline{A_1} \cap \overline{A_2} \cap \ldots\ldots \cap \overline{A_n} = \{\boldsymbol{X}/g_1(\boldsymbol{X})\geq 0 \cap \ldots\ldots \cap g_n(\boldsymbol{X})\geq 0\} = \{\boldsymbol{X}/-g_1()\leq 0 \cap \ldots\ldots \cap -g_n(\boldsymbol{X})\leq 0\} \quad (25)$$

Now, it's obvious that we can get the following conclusion:

Theorem 3: *The free 3D space determined by n obstacles either in the first group or in the second group can be represented by a set of n inequalities with only the intersection operations.*

B. Robot Representation

Robot representation means the expression of a point, denoted by $\boldsymbol{X}=(x,y,z)$, on the boundary of the robot as the function of its spatial position and orientation variables. To clearly model and manipulate a robot in 3D space, we must be capable of representing its spatial location, orientation, size, and shape. To this end, it is well known that six independent variables must be used to represent its spatial position and orientation. Normally a point $\boldsymbol{O}$, denoted by $\boldsymbol{O}(x_o, y_o, z_o,)$, are chosen at the geometric centre of the robot as three independent

position variables and three orientation angles, denoted by $\boldsymbol{\Theta}(\theta_1, \theta_2, \theta_3)$, (for example, a set of three Euler angles) are chosen as the other three independent variables.

There are two ways to describe the boundary of a robot moving in a 3D space. The first is the implicit function which takes the form of $g(x, y, z) = 0$ for its boundary expression. An explicit function $z = z(x, y)$ is a special case of the general implicit function $g(x, y, z) = 0$ due to the fact that $g(x,y,z)=z-z(x,y)$. The second is the parametric form. The x, y, and z are expressed as functions of two auxiliary parameters $\boldsymbol{v}=(t_1, t_2)$, so that $x=x(\boldsymbol{v})=x(t_1, t_2)$, $y=y(\boldsymbol{v})=y(t_1, t_2)$, and $z=z(\boldsymbol{v})=z(t_1,t_2)$. In the following, we use both the implicit and the parametric forms to formulate a robot.

A generalised implicit function representation is given as (26):

$$g(x, y, z, x_o, y_o, z_o, \theta_1, \theta_2, \theta_3)=0 \tag{26}$$

and its equivalent parametric form can be expressed as (27):

$$\begin{aligned} x&=x(x_o, y_o, z_o, \theta_1, \theta_2, \theta_3, \boldsymbol{v}) \\ y&=y(x_o, y_o, z_o, \theta_1, \theta_2, \theta_3, \boldsymbol{v}) \\ z&=z(x_o, y_o, z_o, \theta_1, \theta_2, \theta_3, \boldsymbol{v}) \end{aligned} \tag{27}$$

where x, y, z are boundary points of a robot and mainly responsible for the robot's collision free condition. x_o, y_o, z_o define the robot's spatial location, and θ_1, θ_2, θ_3 are orientation angles to represent the robot's spatial orientation. x_o, y_o, z_o together with θ_1, θ_2, θ_3 are responsible for determining the position and orientation of a robot and are called configuration space variables. $\boldsymbol{v}$ is a vector with two parameters (t_1, t_2) to specify the robot's boundary points.

The function we use to represent robot concretely is the superellipsoid proposed in references [1,2], which is a Constructive solid geometry (CSG) primitive for a broad family of robot and obstacles. The superellipsoid is defined by the function

$$[((x- x_o)/r_x)^{2/s2} + ((y- y_o)/r_y)^{2/s2}]^{s2/s1} + ((z- z_o)/r_z)^{2/s1} =1 \tag{28}$$

the parametric form is:

$$\begin{aligned} x&= r_x\cos^{s1}(t_1)\cos^{s2}(t_2)+ x_o \\ y&= r_y\cos^{s1}(t_1)\sin^{s2}(t_2)+ y_o \qquad -\pi/2 \leqslant t_1 \leqslant \pi/2 \\ z&= r_z\sin^{s1}(t_1)+ z_o \qquad\qquad 0 \leqslant t_2 \leqslant 2\pi \end{aligned} \tag{29}$$

where r_x, r_y, r_z define the geometric extent, s_1 and s_2 specify the shape properties, and x_o, y_o, z_o describe the spatial location respectively. s_1 is the squareness parameter in the north-south direction; s_2 is the squareness parameter in the east-west direction. The superellipsoid can be constructed from the basic slabs. Some superellipsoid shapes produced by the choice of different values for parameters s_1 and s_2 are shown in Fig. 5 when $r_x=r_y=r_z$. When $s_1=s_2=1$, the superellipsoid becomes an ellipsoid that is suitable to describe a path planning vehicle such as an autonomous submarine. When $s_1=s_2=0.5$, the superellipsoid becomes a rectangle with round angle that is suitable for describing a traditional underwater vehicle; while when $s_1=s_2=2$, the superellipsoid becomes a diamond that is suitable for describing a ship. From [1,2], we know that most kinds of robots can be simulated by the broad family of easily defined superellipsoid primitives. In addition to the supersphere shapes that can be generated using various values for parameters s_1 and s_2, other superquadratic shapes can also be combined to create more complex structures. More details about them can be found in [1,2,45].

(28) and (29) describe a 3D robot in the standard orientation with $\theta_1=\theta_2=\theta_3=0$. It's necessary to give a general representation of its spatial orientation. The concepts of Euler angle and Euler angle conversion are introduced in the following.

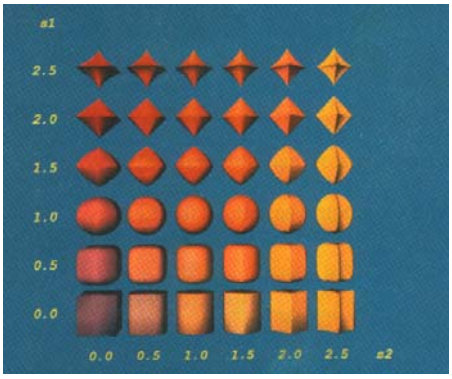

Fig. 5. Superellipsoids plotted with different values for parameters s_1 and s_2 when $r_x=r_y=r_z$.

The Euler angles (α, β, γ) comprise three arbitrary rotations in 3D space to describe the spatial orientation of an object. How the Euler angle is defined and how the rotation matrix R is obtained are briefly described, with the assumption that we start in frame S with cartesian axes, x_{old}, y_{old} *and* z_{old}. A positive (anti-clockwise) rotation of magnitude α about the z axis of S is first carried out and the resulting frame is called S'. Then it is followed by a positive rotation of magnitude β about the y' axis of frame S' and the resulting frame is called S''. Finally, a positive rotation of magnitude γ about the z'' axis of S'' is made and the resulting frame is called S'''. Fig. 6 illustrates the combined effect of these steps.

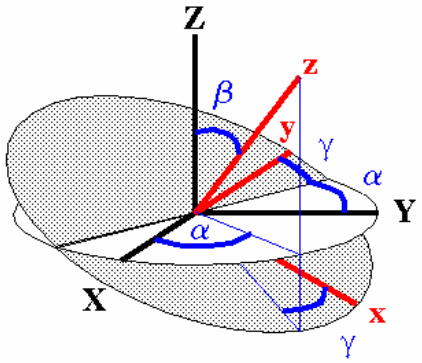

Fig. 6. Euler angle conversion.

The combined result of these three rotations is mathematically expressed by the following rotation matrix:

$$\begin{pmatrix} x_{new} \\ y_{new} \\ z_{new} \end{pmatrix} = R \cdot \begin{pmatrix} x_{old} \\ y_{old} \\ z_{old} \end{pmatrix} \tag{30}$$

where R is the rotation matrix:

$$\begin{pmatrix} \cos\alpha\cos\beta\cos\gamma - \sin\alpha\sin\gamma & \sin\alpha\cos\beta\cos\gamma + \cos\alpha\sin\gamma & -\sin\beta\cos\gamma \\ -\cos\alpha\cos\beta\sin\gamma - \sin\alpha\cos\gamma & \sin\alpha\cos\beta\sin\gamma - \cos\alpha\cos\gamma & \sin\beta\sin\gamma \\ \cos\alpha\sin\beta & \sin\alpha\sin\beta & \cos\beta \end{pmatrix}$$

(30) is equivalent with (31) as follows:

$$\begin{cases} x_{new} = x_{old}\cdot(\cos\alpha\cos\beta\cos\gamma - \sin\alpha\sin\gamma) \\ \quad + y_{old}\cdot(\sin\alpha\cos\beta\cos\gamma + \cos\alpha\sin\gamma) \\ \quad - z_{old}\cdot(\sin\beta\cos\gamma); \\ y_{new} = x_{old}\cdot(-\cos\alpha\cos\beta\sin\gamma - \sin\alpha\cos\gamma) \\ \quad + y_{old}\cdot(\sin\alpha\cos\beta\sin\gamma - \cos\alpha\cos\gamma) \\ \quad + z_{old}\cdot(\sin\beta\sin\gamma); \\ z_{new} = x_{old}\cdot(\cos\alpha\sin\beta) + y_{old}\cdot(\sin\alpha\sin\beta) \\ \quad + z_{old}\cdot\cos\beta; \end{cases} \tag{31}$$

where vector $(x_{old}, y_{old}, z_{old})$ represent the point in the first coordinate system and $(x_{new}, y_{new}, z_{new})$ represent the point in the new coordinate system. The ranges for α,β,γ are

$$0\le\alpha\le 2\pi,\ 0\le\beta\le\pi,\ 0\le\gamma\le 2\pi \tag{32}$$

The combination of rotation transformation (32) with (29) results in the parametric form for a superellipsoid robot in a general position $O(x_o, y_o, z_o,)$ and orientation $\Theta(\theta_1, \theta_2, \theta_3)$:

$$\begin{aligned} x &= (r_x\cos^{s1}(t_1)\cos^{s2}(t_2))l_1 + (r_y\cos^{s1}(t_1)\sin^{s2}(t_2))l_2 \\ &\quad + (r_z\sin^{s1}(t_1))l_3 + x_o \\ y &= (r_x\cos^{s1}(t_1)\cos^{s2}(t_2))m_1 + (r_y\cos^{s1}(t_1)\sin^{s2}(t_2))m_2 \\ &\quad + (r_z\sin^{s1}(t_1))m_3 + y_o \\ z &= (r_x\cos^{s1}(t_1)\cos^{s2}(t_2))n_1 + (r_y\cos^{s1}(t_1)\sin^{s2}(t_2))n_2 \\ &\quad + (r_z\sin^{s1}(t_1))n_3 + z_o \end{aligned} \tag{33}$$

where

$$\begin{aligned} l_1 &= \cos(\theta_1)\cos(\theta_2)\cos(\theta_3) -)\sin(\theta_1)\sin(\theta_3) \\ m_1 &= -\sin(\theta_1)\cos(\theta_3) - \cos(\theta_1)\cos(\theta_2)\sin(\theta_3) \\ n_1 &= \cos(\theta_1)\sin(\theta_2) \\ l_2 &= -\cos(\theta_1)\sin(\theta_3) + \cos(\theta_2)\sin(\theta_2)\sin(\theta_3) \\ m_2 &= -\cos(\theta_1)\cos(\theta_3) + \sin(\theta_1)\cos(\theta_2)\sin(\theta_3) \\ n_2 &= \sin(\theta_2)\sin(\theta_3) \\ l_3 &= \cos(\theta_1)\sin(\theta_2) \\ m_3 &= \sin(\theta_1)\sin(\theta_2) \\ n3 &= \cos(\theta_2) \\ -\pi/2 &\le t_1 \le \pi/2 \\ 0 &\le t_2 \le 2\pi \end{aligned}$$

where the oriental angles $\theta_1, \theta_2, \theta_3$ are specified by three Euler angles. For more details about Euler angle conversion, see [35].

IV. Converting The Robot Path Planning Problem Into The Semi-infinite Optimization Problem

A. The necessary and sufficient condition for a collision free robot in 3D space

The position of a point in a 3D space can be described by a vector :$\boldsymbol{X}=(x, y, z)$. From ***Theorem 3***, the free space determined by an obstacle whenever it belongs to group one or group two can be described by the following inequality constraint:

$$G_1(x, y, z) \le 0 \tag{34}$$

Expressing the free space as the form of inequality constraint (34) is necessary to make it consistent with the requirements of MATLAB's optimization toolbox [57]. The **necessary and sufficient condition** for a point which is collision-free with this obstacle is that it must fall into the free space. Mathematically this condition is equivalent to the satisfaction of inequalities (34).

If there are n obstacles in the workspace, then the whole free space is determined by the following set of inequality constraints with intersection operations

$$G_j(x, y, z) \le 0, j = 1, 2, ..., n \tag{35}$$

To make sure a robot is collision free, a necessary and sufficient condition is that all the points on the robot's surface must be collision free. We thus can obtain the following ***Theorem 4*** by substituting (27) into inequalities (35), which is a general form of necessary and sufficient condition for the collision-free motion of the robot among the obstacles:

Theorem 4: *The necessary and sufficient condition for a collision-free motion of the 3D robot expressed by (27) among a set of 3D obstacles expressed by (35) is the satisfaction of the following inequalities with intersection operations:*

$$G_j(x(x_o, y_o, z_o, \theta_1, \theta_2, \theta_3, \boldsymbol{v}), y(x_o, y_o, z_o, \theta_1, \theta_2, \theta_3, \boldsymbol{v}), z(x_o, y_o, z_o, \theta_1, \theta_2, \theta_3, \boldsymbol{v})) \le 0, j = 1, 2, ..., n \tag{36}$$

In path planning the configuration variables should also meet the limits of the workspace. This requirement can be described as follows:

$$x_{ol} \le x_o \le x_{ou}, y_{ol} \le y_o \le y_{ou}, z_{ol} \le z_o \le z_{ou}, \theta_{1l} \le \theta_1 \le \theta_{1u}, \theta_{2l} \le \theta_2 \le \theta_{2u}, \theta_{3l} \le \theta_3 \le \theta_{3u} \tag{37}$$

where (x_l, y_l, z_l) and (x_u, y_u, z_u) are the lower and upper bounds respectively. (36) and (37) form the inequality constraints required in the formulation of the SICO problem for path planning.

when the robot is represented by a superellipsoid, we can obtain a more specific form of necessary and sufficient condition by substituting (33) into inequalities (36):

$$G_j(\ (r_x\cos^{s1}(t_1)\cos^{s2}(t_2))\ \bullet\ (\cos(\theta_1)\cos(\theta_2)\cos(\theta_3)\text{-})\sin(\theta_1)\sin(\theta_3))\text{-}(r_y\cos^{s1}(t_1)\sin^{s2}(t_2))\ \bullet\ (\cos(\theta_1)\sin(\theta_3)\text{-}\cos(\theta_2)\sin(\theta_2)\sin(\theta_3))+(r_z\sin^{s1}(t_1))\ \bullet\ (\cos(\theta_1)\sin(\theta_2))+x_o,\text{-}(r_x\cos^{s1}(t_1)\cos^{s2}(t_2))\ \bullet\ (\sin(\theta_1)\cos(\theta_3)+\cos(\theta_1)\cos(\theta_2)\sin(\theta_3))\text{-}(r_y\cos^{s1}(t_1)\sin^{s2}(t_2))\ \bullet\ (\cos(\theta_1)\cos(\theta_3)+\sin(\theta_1)\cos(\theta_2)\sin(\theta_3))+(r_z\sin^{s1}(t_1))\ \bullet\ (\sin(\theta_1)\sin(\theta_2))+y_o,(r_x\cos^{s1}(t_1)\cos^{s2}(t_2))\ \bullet\ (\cos(\theta_1)\sin(\theta_2))+(r_y\cos^{s1}(t_1)\sin^{s2}(t_2))\ \bullet\ (\sin(\theta_2)\sin(\theta_3))+(r_z\sin^{s1}(t_1))\ \bullet\ \cos(\theta_2)+z_o)\le 0$$

$$,j=1,2,\ldots,n \qquad (38)$$

where

$-\pi/2\le t_1\le\pi/2,\ 0\le t_2\le 2\pi,\ j=1,2,\ldots,n.$

In (38), s_1 and s_2 are constants for a particular superellipsoid. It's easy to observe from the comparison of (38) with (4) that (38) is the standard form of constraints in the semi-infinite constrained optimization, with $\boldsymbol{v}=(t_1, t_2)$ and $\boldsymbol{x}=(\boldsymbol{O}, \boldsymbol{\Theta})=(x_o, y_o, z_o, \theta_1, \theta_2, \theta_3)$.

B. Design of objective function

There are many ways to design the objective function. In the nonlinear programming problem, this function must represent some meaning of the practical problem, for example, minimum time, minimum distance, minimum energy, or minimum cost. From a mathematical viewpoint, this function must have a minimum lower bound. For the path planning problem, the goal configuration must be designed as the unique global minimum of the configuration variables. We use a quadratic function of form (41) as the objective function, with the goal configuration point (x_{og}, y_{og}, z_{og}) and goal configuration angles $(\theta_{1g}, \theta_{2g}, \theta_{3g})$ being its unique global minimum point and satisfying the condition that $\min f(x_o, y_o, z_o, \theta_1, \theta_2, \theta_3) = f(x_{og}, y_{og}, z_{og}, \theta_{1g}, \theta_{2g}, \theta_{3g}) = 0$.

$$f(x_o, y_o, z_o, \theta_1, \theta_2, \theta_3)=w((x_o-x_{og})^2+(y_o-y_{og})^2+(z_o-z_{og})^2)+(1-w)((\theta_1-\theta_{1g})^2+(\theta_2-\theta_{2g})^2+(\theta_3-\theta_{3g})^2) \qquad (39)$$

where w is a non-negative weighted factor that satisfies: $0\le w\le 1$.

In (39), w is the weight to adjust the relative effects of the spatial position $(x_o-x_{og})^2+(y_o-y_{og})^2+(z_o-z_{og})^2$ and the spatial orientation $(\theta_1-\theta_{1g})^2+(\theta_2-\theta_{2g})^2+(\theta_3-\theta_{3g})^2$. When w=1, only the effect of the spatial position factor is considered. When w=0.5, both factors are considered equally. w can be adaptively adjusted and be revised during the searching process. (38) and (39) together form a semi-infinite constrained optimization problem. If we use the initial configuration variables $(x_{os}, y_{os}, z_{os}, \theta_{1s}, \theta_{2s}, \theta_{3s})$ as the initial estimate of the optimization problem, the optimum search for $(x_o^*, y_o^*, z_o^*, \theta_1^*, \theta_2^*, \theta_3^*)$ is equivalent to searching the goal configuration variables. If the algorithm is convergent and the problem has a solution, then we will find that $x_o^* = x_{og}, y_o^* = y_{og}, z_o^*=z_{og}, \theta_1^*=\theta_{1g}, \theta_2^*=\theta_{2g}, \theta_3^*=\theta_{3g}$. Simulation results will be shown later.

In summary, the fundamental idea for this approach is to represent the free space determined by the robot and obstacles as inequality constraints for a semi-infinite constrained optimization problem in 3D space. The goal configuration is designed as the unique global minimum point of the objective function. The initial configuration is treated as the first search point for the optimization problem. Then the numerical algorithm developed for solving the semi-infinite constrained optimization problem can be applied to solve the robot motion planning problem. Every point generated using the semi-infinite constrained optimization method is guaranteed to be in free space and therefore is collision free.

V. Implementation Considerations And Simulation Results

when implementation is carried out, we only consider the motion of the robot in 3D space because the case for 2D is relatively simple. The vehicle is modeled as a superellipsoid with different shapes and the obstacles are modeled by circle, ellipsoid, cylinder, tetrahedron, cuboids, and various other shapes of superellipsoid, which belong to either the first group or the second group.

A. Algorithm Implementation Consideration

The implementation of the semi-infinite optimization is based on the constrained optimization toolbox[57], but some modifications have been made. The first is the control of the output. In [57], only the final result of the vector $\boldsymbol{x}=(x_o^*, y_o^*, z_o^*, \theta_1^*, \theta_2^*, \theta_3^*)$ is provided. However, the important thing for the robot path planning problem is to generate a smooth path. Therefore, a new function which outputs current $(x_o, y_o, z_o, \theta_1, \theta_2, \theta_3)$ at every iteration has been added. The second is the control of the change of Euler Angles in the line search algorithm for every iteration. Recall that the principle of developing an algorithm in nonlinear programming is to minimize the number of function evaluations which represents the most efficient way for finding the optimum $\boldsymbol{x}^*$. Thus, the change step is automatically chosen as large as possible to minimize the objective function in the line search direction. The disadvantage of this strategy when applied to path planning is that it sometimes leads to a non-smooth path, so modification has been made. For more details on the implementation of the semi-infinite optimization algorithms, see the reference [57].

In the following the results obtained for 3D path planning will be given. In all the experiments the algorithm adopted is the Sequential Quadratic Programming (SQP) [57]. The limits of the workspace are: $\boldsymbol{O}_l=(-20, -20, -20)$ and $\boldsymbol{O}_u=(60, 60, 60)$, thus the workspace is surrounded by a cube with length 80(-20, 60), width 80(-20, 60), and height 80(-20,60). The inequality $\boldsymbol{O}_l\le\boldsymbol{O}\le\boldsymbol{O}_u$ will guarantee that the generated path must be in the workspace. Let $\boldsymbol{O}_s=(x_{os}, y_{os}, z_{os})$ and $\boldsymbol{\Theta}_s=(\theta_{1s}, \theta_{2s}, \theta_{3s})$ denote the initial configuration and $\boldsymbol{O}_g=(x_{og}, y_{og}, z_{og})$, $\boldsymbol{\Theta}_g=(\theta_{1g}, \theta_{2g}, \theta_{3g})$ denote the goal configuration. The robot's start and goal configurations are chosen as $(x_{os}, y_{os}, z_{os}, \theta_{1s}, \theta_{2s}, \theta_{3s})$=(-20, -20, -20, 0, 0, 0) and $(x_{og}, y_{og}, z_{og}, \theta_{1g}, \theta_{2g}, \theta_{3g})$=(50, 50, 50, 0, 0, 0) respectively . The objective function is defined as:

$$f=w((x_o-50)^2+(y_o-50)^2+(z_o-50)^2)+(1-w)(\theta_1^2+\theta_2^2+\theta_3^2) \quad (40)$$

where w is chosen as $0 \leqslant w \leqslant 1$.

It is well known from computer graphics that a set of 3D objects may be viewed from different angles. However it is hard to find a view angle that could clearly illustrate the collision-free path even though we have generated such a path using the approach. For this reason two or three views of the generated path from different angles are presented in all of the following experiments to assist 3D visualization.

B. *Simulation Results with the First Kind of Objects*

In order to demonstrate the convergence of the problem formulation to the goal configuration, the workspace considered in example 1 contains seven obstacles as shown in Figs. 7(a) and (b). The obstacles are represented as spheres with the following boundary expressions: $(x-20)^2 + (y-20)^2 + (z-20)^2 = 225$, $x^2 + (y-20)^2 + (z-20)^2 = 25$, $(x-40)^2 + (y-20)^2 + (z-20)^2 = 25$, $(x-20)^2 + y^2 + (z-20)^2 = 25$, $(x-20)^2 + (y-40)^2 + (z-20)^2 = 25$, $(x-20)^2 + (y-20)^2 + z^2 = 25$, $(x-20)^2 + (y-20)^2 + (z-40)^2 = 25$. The robot is represented by a superellipsoid with $r_x=5$, $r_y=4$, $r_z=3$, $s_1=s_2=1$, which is an ellipsoid. Figs. 7(a) and (b) show the same generated path, the same obstacles and the same robot, but from different view angles using the optimization toolbox in Matlab. It can be observed that the optimization algorithm does converge to the goal and a smooth path has been generated.

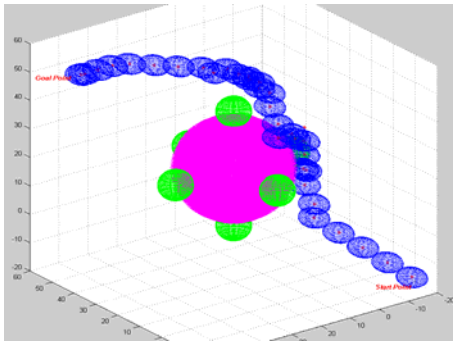

Fig. 7(a) Workspace with seven circles and the generated collision-free path, viewed from one angle.

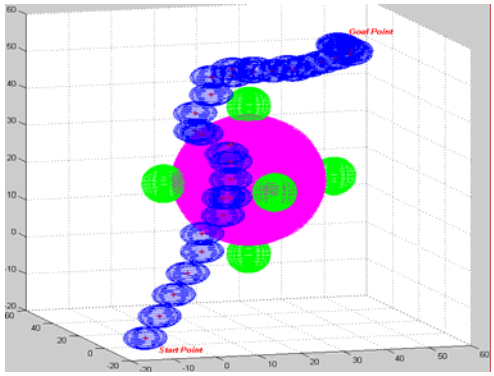

Fig. 7(b) The same result as that in Fig. 7 (a), but viewed from another angle.

In order to illustrate the suitability of the approach for different obstacle shapes, we have shown another distributed-obstacle situation as shown in Figs. 8(a) and 8(b). The results simulate a real word path planning task encountered in offshore industry. The designed workspace contains a set of cylinders which could be easily recognized as pipelines or a base of an offshore structure. In this example, we have 9 cylinders to represent obstacles with the following boundary equations: $x^2 + y^2 = 16$, $x^2 + (y-20)^2 = 16$, $x^2 + (y-40)^2 = 16$, $(x-20)^2 + y^2 = 16$, $(x-20)^2 + (y-20)^2 = 16$, $(x-20)^2 + (y-40)^2 = 16$, $(x-40)^2 + y^2 = 16$, $(x-40)^2 + (y-20)^2 = 16$, $(x-40)^2 + (y-40)^2 = 16$ where $-20 \leq z \leq 60$. The robot is represented by a superellipsoid with $r_x=5$, $r_y=4$, $r_z=3$, $s_1=s_2=1.5$, which is a superellipsoid. Fig. 8(a) is a 3D view and Fig. 8(b) is a 2D view. In this example the 2D view clearly shows the collision avoidance of the robot from the obstacles. From Fig. 8(a) and Fig. 8(b), we can also observe that the generated path passes behind the cylinder without touching it, and the plotted path avoids all the obstacles and converges to the goal in a smooth way. We have tested many other start points and the results are all quite satisfactory,

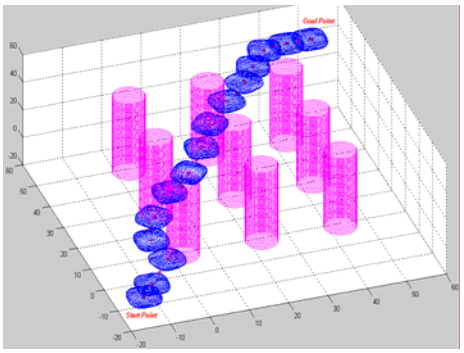

Fig. 8(a) Workspace with nine cylinders and the generated path, viewed from one angle.

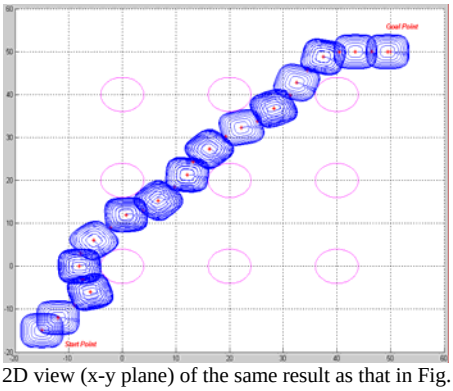

Fig. 8(b) A 2D view (x-y plane) of the same result as that in Fig. 8(a).

although they are not reported here for the limitation of space.

In addition to the simulation results shown in Figs. 7 and 8, we have carried out another test with mixed superellipsoids and cylinders as shown in Figs. 9(a), (b) and (c). In this test, five obstacles are included and the robot is represented by a superellipsoid with $r_x=8$, $r_y=8$, $r_z=6$, $s_1=s_2=0.8$. The boundary expressions for the obstacles are: 2 cylinders: $(x-40)^2 + y^2 = 144$, $30 \leq z \leq 50$, $(x-30)^2 + (y-30)^2=100$, $-5 \leq z \leq 30$, 3 superellipoids: $((x-15)/12) + ((y+10)/8 + ((z-12)/12) =1$, $((x-15)/20)^{2/3} + ((y-40)/18)^{2/3} + ((z+10)/12)^{2/3} =1$, $(x-10)^2 + (y-10)^2 + (z-10)^2 = 100$. Fig. 9(a) is a 3D view and Fig. 9(b) is a 2D view, Fig. 9(c) is an enlarged view of Fig. 9(a). This

experiment results show that the robot can adjust its orientation angles autonomously to reach the goal point.

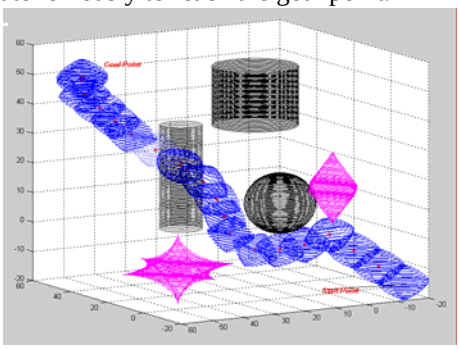

Fig. 9(a) Simulation result with mixed superellipsoids and cylinders: a 3D view.

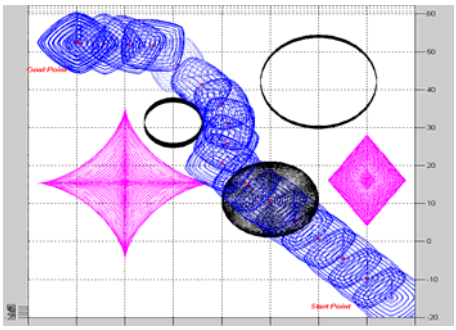

Fig. 9(b) Simulation result with mixed superellipsoids and cylinders: a 2D view

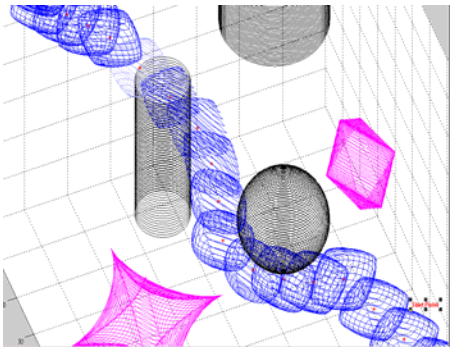

Fig. 9(c) Simulation result with mixed superellipsoids and cylinders: another 3D view

C. Simulation Results with the Both the First and the Second Kinds of Objects

Figs. 10(a), (b) and (c) show the experimental results with the environment including a triangular pyramid, a cube which belong to the second group and a cylinder-like obstacle belonging to the first group. The triangular pyramid's four vertexes are: V_0=(-5,-10,-15), V_1=(20,15,-15), V_2=(5,-15,10), V_3=(5,20,10) and its outside is represented by $\{(h_1 \geq -x-y-0.6z-14) \cup (h_2 \geq -x+0.4z+1) \cup (h_3 \geq x+0.6z-11) \cup (h_4 \geq -x+y-0.8z-7)\}$.The cube is represented by $10 \leq x \leq 25$, $25 \leq y \leq 45$, $25 \leq z \leq 40$, while the cylinder is descried as $(x-35)^2 + (y-20) \leq 64$, $-10 \leq z \leq 50$. The robot is represented by a superellipsoid with r_x=8, r_y=6, r_z=4, s_1=s_2=1.It is obvious that the path generated using the mathematical transformation clearly avoids the obstacle and converges to the goal. Figs. 10(a) and 10(b) are all 3D views form different view angles, and Fig 10(c) is a 2D view.

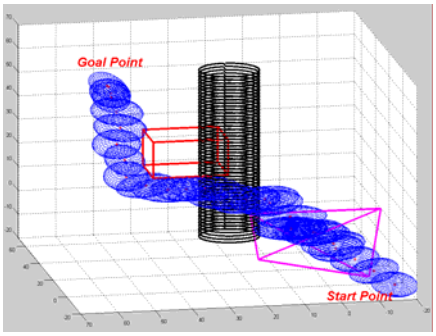

Fig. 10(a) Simulation result with both the first and the second kinds of objects: a 3D view.

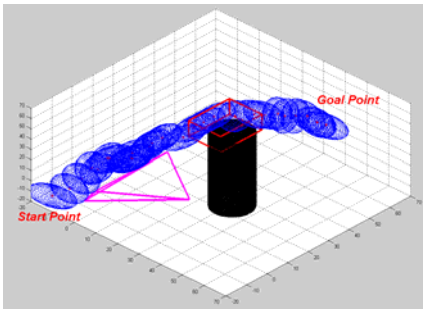

Fig. 10(b) Simulation result with both the first and the second kinds of objects: another 3D view.

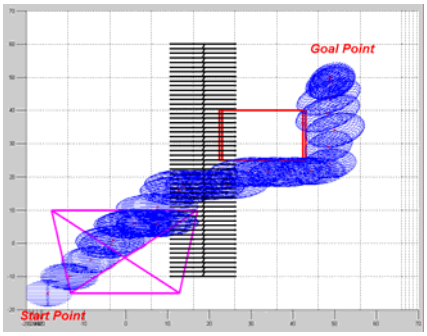

Fig. 10(c) Simulation result with both the first and the second kinds of objects: a 2D view.

Figs. 11(a) and (b) show the results simulating another typical real word path planning task encountered in offshore industry. The designed workspace contains a cylinder and several cubes which may be models for pipelines or a base of an offshore structure. The cylinder is described as follows: $(x-20)^2 + (y-20) \leq 49$, $-10 \leq z \leq 50$. The four cubes are represented as: $\{0 \leq x \leq 10, 5 \leq y \leq 10, -15 \leq z \leq 55\}$, $\{30 \leq x \leq 40, 30 \leq y \leq 35, -15 \leq z \leq 55\}$,$\{5 \leq x \leq 10, 30 \leq y \leq 40, -15 \leq z \leq 55\}$,$\{30 \leq x \leq 35, 0 \leq y \leq 10, -15 \leq z \leq 55\}$. The robot is represented by a superellipsoid with r_x=6, r_y=4, r_z=3, s_1=s_2=1.2. Figs.11 (a) and (b) show the clear avoidance of the obstacles by the robot and the convergence to the goal.

Superquadrics are a new and powerful family of parametric shapes and extend the basic quadric and solids, yielding a variety of useful forms. In this paper, extensive experiments have been performed with different object shapes, different object distributions, and different start and goal configurations. Although the results are not presented here, the path planner converges to the goal successfully, showing its excellent capability to navigate through the described workspace and the suitability to various object shapes.

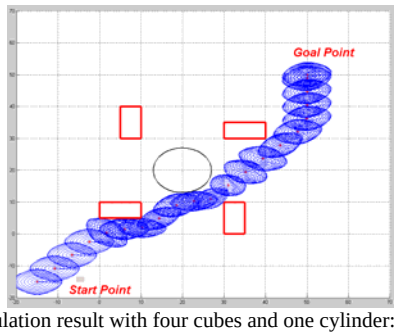

Fig. 11(a) Simulation result with four cubes and one cylinder: a 2D view.

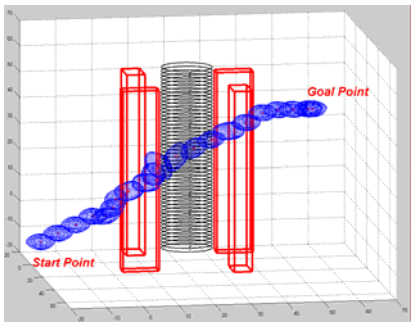

Fig. 11(b) Simulation result with four cubes and one cylinder: a 3D view.

VI. Conclusions

In this paper, set operation, inequality transformation, and semi-infinite constrained optimization techniques have been presented for the development of a realistic real-time obstacle avoidance approach for subsea vehicle. We have shown the principle of converting the path planning problem into the standard Semi-infinite Constrained Optimization problem. This direct path planning approach considers the robot's 3D shape and is totally different from the traditional approaches in the way that the calculation of the Cspace obstacles is no longer needed. This new real-time robot path planning approach, to our best knowledge, is so far the first one in the robotic community. From the viewpoint of robot path planning, this paper presents a new way of using a classical engineering approach. The generality of representing the free space of all the objects using the inequalities $g_i(\boldsymbol{x}) \leq 0$ makes this optimization-based approach suitable for different object shapes and has significantly simplified the construction of the objects by sensors and computer vision systems. The iterative nature of the search for the optimization point makes it particularly suitable for on-line sensor-based path planning. Once a new object is detected a new inequality can be added before running the next iteration. When an old obstacle is passed, its corresponding inequality may also be deleted. Every time when an inequality is added or deleted, a new optimization problem is formed. The current variable is always treated as the initial point of the optimization problem and search starts again. This mechanism explains why this approach is particularly suitable for on-line path planning.

The advantages of the approaches are that mature techniques developed in nonlinear programming theory which guarantee convergence, efficiency and numerical robustness can be directly applied to the robot path planning problem. The Semi-infinite constrained optimization approach with an adaptive objective function has the following advantages:

1. The robot does not need to be shrunk to a point and the obstacles do not need to be expended to Cspace, the entire course of path planning can be carried out in real 3D space.
2. The goal point is guaranteed to be the only global minimum of the objective function.
3. The standard search techniques which have been developed for more than thirty years in the nonlinear programming field can be used.
4. The approach is suitable for on-line task planning.

The investigation carried out in this paper has indicated that robot path planning can be formulated as semi-infinite constrained optimization problem. Although the fundamentals for the nonlinear programming theory have existed for many years, they have not attracted enough attention for such applications. The context presented in this paper covers a wide range of subjects such as robot kinematics, CAD, CAM, Computer Graphics and nonlinear programming theory, and a basic framework has been developed. Our treatment is consistent. The study presented in this paper has shown its great potential as an on-line motion planner. The future work includes the extension of the principle developed here to the obstacle avoidance problem for manipulator without the calculation of Cspace obstacles.

Acknowledgment

The authors gratefully acknowledge the financial support from Chinese Academy of Sciences, P. R. China and Royal Society of United Kingdom for the joint research project under grant No. 20030389, 2003-2006, and the financial support from Chinese Academy of Sciences and Chinese State Development Planning Commission for Bai Ren Ji Hua (BRJH), 2002-2005.

References

[1] A. H. Barr, "Superquadrics and angle-preserving transformations," *IEEE Computer Graphics and Applications*, Vol. 1, pp. 11-23, 1981.
[2] M. Berger, *Computer Graphics*, The Benjamin/Cummings Publishing Company, Inc., California, 1986.
[3] D. Blackmore, M. Leu, and L. P. Wang, "The sweep-envelope differential equation algorithm and its application to NC machining verification," *Computer Aided Design*, Vol. 29, pp. 629-637, 1997.
[4] J. L. Blechschmidt and D. Nagasuru, "The use of algebraic

functions as a solid modelling alternative: an investigation," *Proc. of the 16th ASME Design Automation Conf*., Chicago, USA, pp. 33-41, 1990.

[5] R. A. Brooks and T. Lozano-perez, "A subdivision algorithm in configuration space for findpath with rotation," *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 15, pp. 224-233, 1985.

[6] J. C. Chang and H. C. Lu, "Backing up a simulated truck via grey relational analysis," *Journal of the Chinese Institute of Engineers,* Vol. 24, pp. 745-752, 2001.

[7] H. Chiyokura, *Solid Modelling with Designbase*, Addison-Wesley Publishing Limited, New York, 1988.

[8] P. G. Comba, "A procedure for detecting intersections of three-dimensional objects," *JACM*, Vol. 15, pp. 354-366, 1968.

[9] R. A. Conn and M. Kam, "On the moving-obstacle path planning algorithm of Shih, Lee, and Gruver," *IEEE Trans. on Systems, Man and Cybernetics ---- part B: Cybernetics*, Vol. 27, pp. 136-138, 1997.

[10] I. Connolly, "Harmonic function and collision probabilities," *Int. J. Robotics Research*, Vol. 16, pp. 497-507, 1997.

[11] R. Fletcher, *Practical Methods of Optimization*, John Wiley & Sons., New York, 1987.

[12] W. R. Franklin and A. H. Barr, "Faster calculation of superquadric shapes," *IEEE Computer Graphics and Application*, Vol. 1, 41-57, pp. 1981.

[13] P. E. Gill, W. Murray and M. H. Wright, *Practical Optimization*, Academic Press, London, 1981.

[14] M. Hall and J. Warren, "Adaptive polygonalization of implicitly defined surfaces," *IEEE Computer Graphics & Applications*, Vol. 10, pp. 33-42, 1990.

[15] S. P. Han, "A globally convergent method for nonlinear programming," *J. Optimization Theory and Applications*, Vol. 22, pp. 297-235, 1977.

[16] E. J. Haug, F. A. Adkins, and D. Cororian, "Domains of mobility for planar body moving among obstacles," *Transactions of the ASME, Journal of Mechanical Design*, Vol. 120, pp. 462-467, 1988.

[17] T. C. Hu, A. B. Kahng and G. Robins, "Optimal robust path planning in general environment," *IEEE Trans. Robotics and Automation*, Vol. 9, pp. 755-774, 1993.

[18] H. P. Huang and P. C. Lee, "A real-time algorithm for obstacle avoidance of autonomous mobile robots," *Robotica*, Vol. 10, pp. 217-227, 1992.

[19] Y. K. Hwang and N. Ahuja, "A potential field approach to path planning," *IEEE Trans. on Robotics and Automation*, Vol. 8, pp. 23-32, 1992.

[20] O. Khatib, "Real-time obstacle avoidance for manipulator and mobile robots," *Int. J. Robotics Research*, Vol. 5, pp. 90-98, 1986.

[21] E. Kreyszig, *Advanced Engineering Mathematics*, Seventh Edition, John Wiley & Sons, New York, 1993.

[22] C. M. Krishna and K. G. Shin, *Real-time Systems*, McGraw-Hill, New York, 1997.

[23] J. C. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, Boston, 1991.

[24] T. Lozano-perez, "Spatial planning: A configuration space approach," *IEEE Trans. on Computer,* Vol. 32, pp. 108-120, 1983.

[25] H. C. Lu and M. E. Yeh, "Robot path planning based on modified grey relational analysis," Cybernetics and Systems, Vol. 33, pp. 129-159, 2002.

[26] D. G. Luenberger, *Linear and Nonlinear Programming*, Addison-Wesley Publishing Company, 2nd ed. London, 1984.

[27] V. J. Lumelsky, "A comparative study on path length performance of maze-searching and robot motion planning," *IEEE Trans. on Robotics and Automation*, Vol. 7, pp. 57-66, 1991.

[28] G. Oriolo, G. Ulivi, M. Vendittelli, "Real-time map building and Navigation for autonomous robots in unknown environments," *IEEE Trans. on Systems, Man and Cybernetics, PartB: Cybernetics*, Vol. 28, pp. 316-333, 1998.

[29] T. J. Park, J. W. Ahn, and C. S. Han, "A path generation algorithm of an automatic guided vehicle using sensor scanning method," *KSME International Journal,* Vol. 16, pp. 137-146, 2002.

[30] Y. R. Petillot, T. I. Ruiz, D. M. Lane, Y. Wang, E. Trucco, and N. Pican, "Underwater vehicle path planning using a multi-beam forward looking sonar," *IEEE Oceanic Engineering Society, OCEANS'98, 1998, Nice, France,* pp. 1194-1199, 1998.

[31] Y. Petillot, I. T. Ruiz, and D. M. Lane, "Underwater vehicle obstacle avoidance and path planning using a multi-beam forward looking sonar," *IEEE Journal of Oceanic Engineering,* Vol. 26, pp. 240-251, 2001.

[32] E. Polak and D. Q. Mayne, "Control system design via semi-infinite optimization: a review," *Proceedings of IEEE*, Vol. *72*, pp. 1777-1793, 1984.

[33] S. S. Rao, *Optimization theory and applications*, Wiley Eastern Ltd, Second Edition, New Delhi, 1984.

[34] A. Ricci, "A constructive geometry for computer graphics," *The Computer Journal*, Vol. 16, pp. 157-160, 1973.

[35] E. M. Rose, *Elementary Theory of Angular Momentum*, John Wiley & Sons, New York, 1957.

[36] K. H. Rosen, *Discrete mathematics and Its Applications*, McGraw-Hill, Inc., Second Edition, New York, 1991.

[37] T. Ruiz, D. M. Lane DM, and M. J. Chantler, "A comparison of inter-frame feature measures for robust object classification in sector scan sonar image sequences," *IEEE Journal of Oceanic Engineering*, Vol. 24, pp. 458-469, 1999.

[38] C. L. Shih, J. T. Jeng, "Stabilization of non-holonomic chained systems by gain scheduling," *International Journal of Systems Science*, Vol. 30, pp. 441-449, 1999.

[39] S. Sundar, and S. Shiller, "Optimal obstacle avoidance based on the HJB equation," *IEEE Trans. on Robotics and Automation*, 13, pp. 305 –310, 1997.

[40] Y. Tanak, M. Fukushima, and T. Ibaraki, "A comparative study of several semi-infinite nonlinear programming algorithms," *European Journal of Operational Research*, Vol. 36, pp. 92-100, 1988.

[41] K. Tang, M. Wang, L. Chen, S. Chou, T. Woo, and R. Janardane, "Computing planar swept polygons under translation," *Computer Aided Design*, Vol. 29, pp. 825-836, 1997.

[42] E. Trucco, Y. R. Petillot, I. T. Ruiz, K. Plakas, D. M. Lane, "Feature tracking in video and sonar subsea sequences with applications," Computer Vision and Image Uuderstanding, Vol. 79, pp. 92-122, 2000.

[43] W. P. Wang and K. K. Wang, "Geometric Modeling for swept volume of moving solids," *IEEE CG&A*, Vol. 12, pp. 8-17, 1986.

[44] Yongji Wang, "A note on "solving the find-path problem by good representation of free space," *IEEE Trans. on Systems, Man and Cybernetics ---- part B: Cybernetics*, Vol. 27, pp. 723-724, 1997.

[45] Yongji Wang, and D. M. Lane, "Subsea vehicle path planning using nonlinear programming and constructive solid geometry," *IEE Proc., Part D, Control theory and applications,* Vol. 144, pp. 143-152, 1997.

[46] Yongji Wang, D. M. Lane and G. J. Falconer, "Two novel approaches for unmanned underwater vehicle path planning: constrained optimization and semi-infinite constrained optimization," *Robotica*, Vol. 18, pp. 123-142, 2000.

[47] Yongji Wang and David M. Lane, "Solving a generalized

constrained optimization problem with both logic AND and OR relationships by a mathematical transformation and its application to robot path planning," *IEEE Trans. on Systems, Man and Cybernetics, Part C*: *Application and Reviews,* Vol. 30, pp. 525-536, 2000.

[48] Yongji Wang and M. P. Cartmell, "A new overtaking model for Passing Sight Distance, PDS) on two-lane highways," *ASCE J. of Transportation Engineering,* Vol. 124, pp. 536-545, 1998.

[49] Yongji Wang and M. P. Cartmell, "Autonomous vehicle parallel parking design using function fitting approaches" , *Robotica*, Vol. 16, pp. 159-170, 1998.

[50] Yongji Wang and M. P. Cartmell, "Trajectory generation for four-wheel steering tractor-trailer system: a two step method," *Robotica*, Vol. 16, pp. 381-386, 1998.

[51] Yongji Wang and J. A. Linnett, "Vehicle kinematics and its application to highway design," *ASCE J. of Transportation Engineering*, Vol. 121, pp. 63-74, 1995.

[52] Yongji Wang, J. A. Linnett and J. W. Roberts, "Motion feasibility of a wheeled vehicle with a steering angle limit," *Robotica*, Vol. 12, pp. 217-226, 1994.

[53] Yongji Wang, J. A. Linnett, and J. Roberts, "Kinematics, kinematic constraints and path planning for wheeled mobile robots," *Robotica*, Vol. 12, pp. 391-400, 1994.

[54] Yongji Wang, *Kinematics, motion analysis and path planning for four kinds of wheeled mobile robots,* Ph. D Thesis, Department of Mechanical Engineering, Edinburgh University, 1995.

[55] W. L. Xu, B. L. Ma, "Polynomial motion of non-holonomic mechanical systems of chained form," *Mathematical Methods in the Applied Sciences*, Vol. 22, pp. 1153-1173, 1999.

[56] Y. Zhang, and K. P. Valavanis, "A 3-D potential panel method for robot motion planning," *Robotica,* Vol. 15, pp. 421-434, 1997.

[57] The Math Works Inc., MATLAB, 1993, Optimization Toolbox User's Guide.