International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:2, No:1, 2008

Self-protection Method for Flying Robots to Avoid Collision

Guosheng Wu, Luning Wang, Changyuan Fan, and Xi Zhu

***Abstract*—**This paper provides a new approach to solve the motion planning problems of flying robots in uncertain 3D dynamic environments. The robots controlled by this method can adaptively choose the fast way to avoid collision without information about the shapes and trajectories of obstacles. Based on sphere coordinates the new method accomplishes collision avoidance of flying robots without any other auxiliary positioning systems. The Self-protection System gives robots self-protection abilities to work in uncertain 3D dynamic environments. Simulations illustrate the validity of the proposed method.

***Keywords*—**Collision avoidance, Mobile robots, Motion-planning, Sphere coordinates, Self-protection

I. Introduction

COLLISION avoidance is one of the key issues in path planning research on a robot finding a way from the starting point to goal in the presence of obstacles. There are various approaches have been discussed for static obstacles in known or unknown environment, such as C-space method [1], artificial potential field method [2], genetic algorithms [3] and dynamic window [4-5]. These methods may obtain perfect results under some certain conditions. But in fact, robots usually work in dynamic uncertain environment including static obstacles with unknown position and dynamic obstacles with uncertain trajectory. And the collision-free navigation of mobile robots in dynamic uncertain environment is so complex that it is still an intractable topic by far.

Although many algorithms have been proposed based on 2D environment to avoid the dynamic obstacles for mobile robots on the ground [6], most of them are not suitable for flying robots which can move in 3D space, because they are facing collisions from every direction. In this paper, a new method based on the sensors' information for motion planning of flying robots in uncertain dynamic 3D space environments is proposed. This algorithm gives the robots self-protection ability which can protect themselves from any obstacles' impact as Asimov I defined that the robot should own self-protection ability which do not conflict with that the robot should not let human beings injured and should obey the order from human beings [7].

Using this method, the robot not only can avoid the obstacles but also can sidestep anything rushing towards. This algorithm also can be used on the aircrafts and any other machines which can fly in the air or even in the outer-space. The self-protection ability can let these machines protect themselves from hurts especially protect the entire things inside them. This self-protection method based on 3D sphere coordinates, which can directly provide the desired acceleration for flying robots, has been studied very little so far. Simulation experiments are given to illustrate it.

Guosheng Wu is with the School of Automation Engineering at University of Electronic Science and Technology of China, Chengdu, CO 610054 P. R. China (Phone: +86-13084445761; E-mail: wgsheng@gmail.com).

Luning Wang is with the Department of Computer Science at Beijing Institute of Technology, Beijing, CO 100081, P. R. China (Phone: +86-13810267537 ; E-mail: evangelium@gmail.com)

Changyuan Fan is an associate professor in the Department of Control Engineering at Chengdu University of Information Technology, Chengdu, CO 610041, P. R. China. (Phone: +86-13881863183; E-mail: xiongxiong123@cuit.edu.cn)

Xi Zhu is with Computer Science at Beijing University of Aeronautics and Astronautics, Beijing, CO 100083, P.R.China.(E-mail:zhux1984@yahoo.com.cn)

II. SELF-PROTECTION SYSTEM

The Self-protection system was developed based on an LMS291 laser range finder (SICK AG, Waldkirch, Germany). Shown as Fig. 1, this laser range finder (LRF) employs a laser optical scanner to detect the distance of an object of interest by measuring the "time-of-flight" of laser light pulses. During the measurement, the scanner emits pulsed laser beams and receives the beams being reflected from a detected object. The distance of the detected object is determined by the time interval between the emission and reception of the laser beams. In addition, the LRF can be used to survey the 3D working environment for robots [8-9].

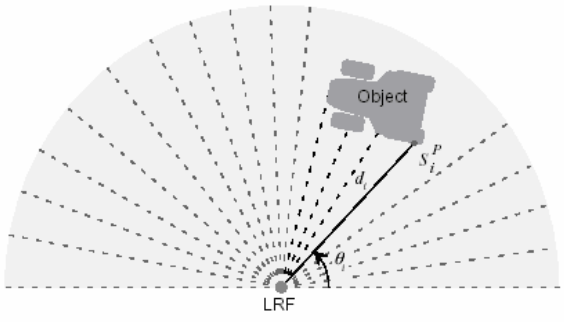


Fig. 1.Principle of the Sensor Scanning

The flow of Self-protection System is showed in Fig. 2. After initiated, the sensor collects data from 3D space for CPU, and then CPU calculates Relative Velocity of Obstacles

to robot. Using these data, whether the robot in the collision area or not can be computed. If necessary, Self-protection System can be activated to avoid the obstacles.

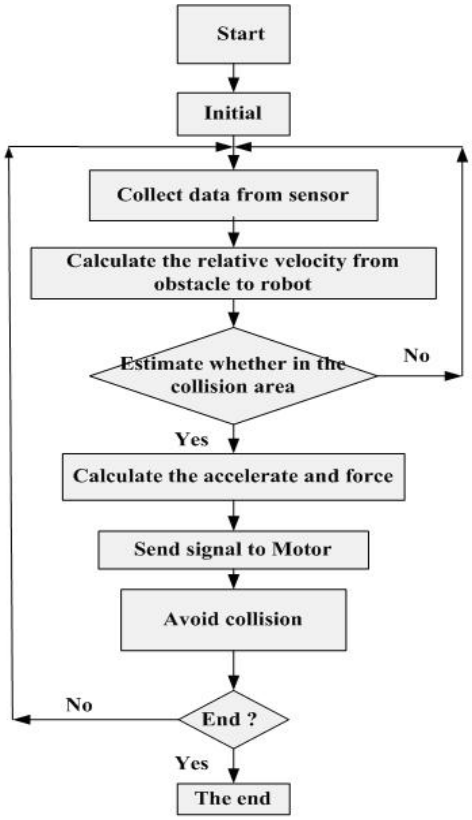


Fig. 2 Flow of Self-protection System

III. ALGORITHM OF SELF-PROTECTION SYSTEM

When running by the highest speed in space, the robot can only change the angle of Velocity. So we can suppose:

$$ {}^{R}V = \begin{bmatrix} V_R \\ 0 \\ 0 \end{bmatrix} = V_R \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} $$

$$ {}^{o}V = \begin{bmatrix} V_o \cos\theta_o \cos\varphi_o \\ V_o \sin\theta_o \cos\varphi_o \\ V_o \sin\varphi_o \end{bmatrix} = V_o \begin{bmatrix} \cos\theta_o \cos\varphi_o \\ \sin\theta_o \cos\varphi_o \\ \sin\varphi_o \end{bmatrix} $$

$$ {}^{R}V_{want} = V_R \begin{bmatrix} \cos\theta_v \cos\varphi_v \\ \sin\theta_v \cos\varphi_v \\ \sin\varphi_v \end{bmatrix} $$

$$ \Delta V_{want} = V_\Delta \begin{bmatrix} \cos\theta_\Delta \cos\varphi_\Delta \\ \sin\theta_\Delta \cos\varphi_\Delta \\ \sin\varphi_\Delta \end{bmatrix} $$

$$ \overrightarrow{\Delta V_{O\to R}} = \begin{bmatrix} V_o \cos\theta_o \cos\varphi_o - V_R \\ V_o \sin\theta_o \cos\varphi_o \\ V_o \sin\varphi_o \end{bmatrix} = V_{\Delta v} \begin{bmatrix} \cos\theta_{\Delta v} \cos\varphi_{\Delta v} \\ \sin\theta_{\Delta v} \cos\varphi_{\Delta v} \\ \sin\varphi_{\Delta v} \end{bmatrix} $$

${}^{R}V$: The velocity of robot to the ground.

${}^{o}V$: The velocity of obstacle to the ground.

${}^{R}V_{want}$: The velocity of robot we want to change to, if collision happens.

ΔV_{want} : The relative velocity we want to change to, if collision happens.

$\overrightarrow{\Delta V_{O\to R}}$: The relative velocity from obstacle to robot.

V_R , V_o , V_Δ and $V_{\Delta v}$: The magnitude of ${}^{R}V$, ${}^{o}V$, ${}^{R}V_{want}$, ΔV_{want} and $\overrightarrow{\Delta V_{O\to R}}$.

θ_o , θ_v , θ_Δ and $\theta_{\Delta v}$ denote the level angle of ${}^{o}V$, ${}^{R}V_{want}$, ΔV_{want} and $\overrightarrow{\Delta V_{O\to R}}$ while φ_o , φ_v , φ_Δ and $\varphi_{\Delta v}$ present the vertical angle of them.

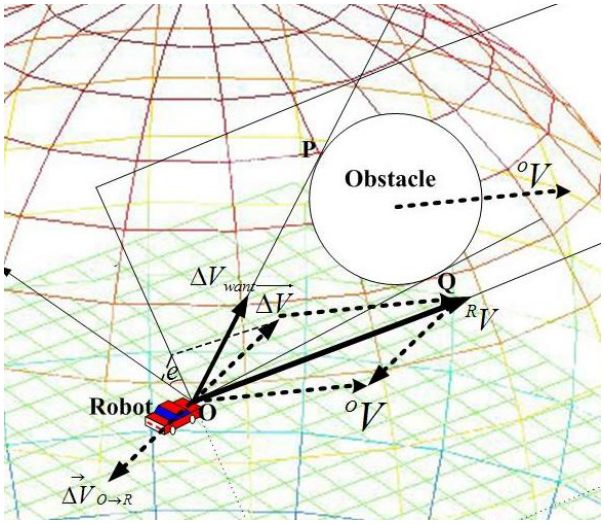


Fig. 3 Model of Collision in 3D Space

The model of Collision in 3D space is shown in Fig.3. $\overrightarrow{\Delta V_{O\to R}}$ Can be calculated from the data of sensor

and ${}^{R}V_{want}$ is known. Because ΔV_{want} and $\vec{OP}$ on the same line, θ_Δ and φ_Δ can be reached from the sensor's data.

Although we do not know V_Δ, yet we can deduce it from

$$V_o{}^2 + V_\Delta{}^2 - 2V_o V_\Delta \cos z = V_R{}^2 \quad (1)$$

In (1) z presents the angle from ${}^{o}V$ to ΔV_{want}. Because V_Δ can not be negative, so we have

$$V_\Delta = V_o \cos z + \sqrt{V_o{}^2(\cos 2z - 1) + V_R{}^2} \quad (2)$$

We also have

$${}^{R}V_{want} = {}^{o}V - \Delta V_{want} = \vec{\Delta V}_{O\to R} + {}^{R}V - \Delta V_{want} \quad (3)$$

That is

$$V_R \begin{bmatrix} \cos\theta_v \cos\varphi_v \\ \sin\theta_v \cos\varphi_v \\ \sin\varphi_v \end{bmatrix} = V_{\Delta v} \begin{bmatrix} \cos\theta_{\Delta v} \cos\varphi_{\Delta v} \\ \sin\theta_{\Delta v} \cos\varphi_{\Delta v} \\ \sin\varphi_{\Delta v} \end{bmatrix} + V_R \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} - V_\Delta \begin{bmatrix} \cos\theta_\Delta \cos\varphi_\Delta \\ \sin\theta_\Delta \cos\varphi_\Delta \\ \sin\varphi_\Delta \end{bmatrix} \quad (4)$$

We can get the solution of (4)

$$\begin{bmatrix} \theta_v \\ \varphi_v \end{bmatrix} = \begin{bmatrix} a\cos \dfrac{V_R - V_\Delta \cos\theta_\Delta \cos\varphi_\Delta + V_{\Delta v} \cos\theta_{\Delta v} \cos\varphi_{\Delta v}}{V_R \sqrt{1 - (\dfrac{V_{\Delta v} \sin\varphi_{\Delta v} - V_\Delta \sin\varphi_\Delta}{V_R})^2}} \\ a\sin \dfrac{V_{\Delta v} \sin\varphi_{\Delta v} - V_\Delta \sin\varphi_\Delta}{V_R} \end{bmatrix} \quad (5)$$

Because ${}^{R}V$, ${}^{o}V$, ${}^{R}V_{want}$ and $\overrightarrow{\Delta V_{O\to R}}$ in the same plane, so we can find a line which pass the zero point and is perpendicular with the plane. Assume the line is

$${}^{R}k = \begin{bmatrix} \cos\theta_k \cos\varphi_k \\ \sin\theta_k \cos\varphi_k \\ \sin\varphi_k \end{bmatrix} = \begin{bmatrix} a_k \\ b_k \\ c_k \end{bmatrix} \quad (6)$$

$$\begin{bmatrix} a_k & b_k & c_k \end{bmatrix} \bullet \begin{bmatrix} i \\ j \\ k \end{bmatrix} = \begin{vmatrix} i & j & k \\ 1 & 0 & 0 \\ \cos\theta_{\Delta v} \cos\varphi_{\Delta v} & \sin\theta_{\Delta v} \cos\varphi_{\Delta v} & \sin\varphi_{\Delta v} \end{vmatrix} \quad (7)$$

a_k, b_k and c_k can be reached from (7):

$$\begin{bmatrix} a_k \\ b_k \\ c_k \end{bmatrix} = \begin{bmatrix} 0 \\ -\sin\varphi_{\Delta v} \\ \sin\theta_{\Delta v} \cos\varphi_{\Delta v} \end{bmatrix} \quad (8)$$

Suppose that $d\varphi$ presents the angle from ${}^{R}V$ to ${}^{R}V_{want}$, we have

$$\cos d\varphi = \cos\theta_v \cos\varphi_v \quad (9)$$

Through (9) we can get

$$d\varphi = a\cos \frac{V_R - V_\Delta \cos\theta_\Delta \cos\varphi_\Delta + V_{\Delta v} \cos\theta_{\Delta v} \cos\varphi_{\Delta v}}{V_R} \quad (10)$$

Based on all above, we finally know $J(k, d\varphi)$ from (11) which showed at the end of the page. Then there exists

$${}^{R}V_{want} = J(k, d\varphi)\,{}^{R}V \quad (12)$$

The acceleration and the power on robot are

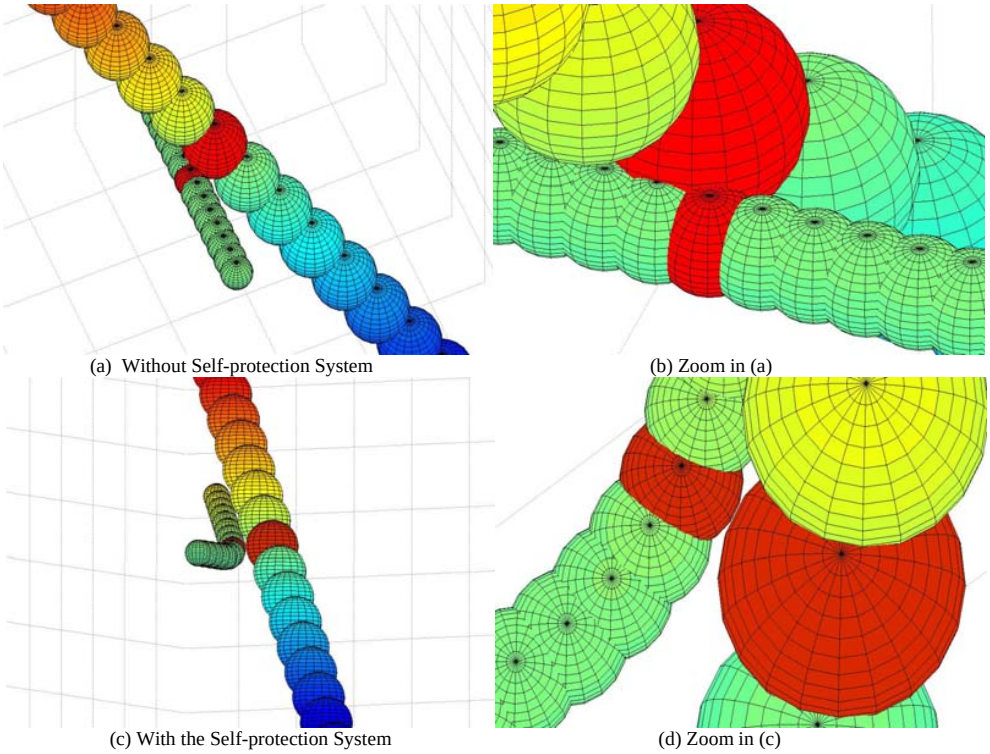
$${}^{R}a = {}^{R}V_{want} - {}^{R}V = V_R \begin{bmatrix} \cos\theta_v \cos\varphi_v - 1 \\ \sin\theta_v \cos\varphi_v \\ \sin\varphi_v \end{bmatrix} \quad (13)$$

$$F = m\,{}^{R}a = m\,{}^{R}V_R \begin{bmatrix} \cos\theta_v \cos\varphi_v - 1 \\ \sin\theta_v \cos\varphi_v \\ \sin\varphi_v \end{bmatrix} \quad (14)$$

So if we can get θ_v, φ_v, $d\varphi$, with (2), (5), (10), (12) and (14), Self-protection can be realized.

$$J(k, d\varphi) = \begin{bmatrix} a_k{}^2(1-\cos d\varphi) + \cos d\varphi & b_k a_k(1-\cos d\varphi) - c_k \sin d\varphi & c_k a_k(1-\cos d\varphi) + b_k \sin d\varphi \\ a_k b_k(1-\cos d\varphi) + c_k \sin d\varphi & b_k{}^2(1-\cos d\varphi) + \cos d\varphi & c_k b_k(1-\cos d\varphi) - a_k \sin d\varphi \\ a_k c_k(1-\cos d\varphi) - b_k \sin d\varphi & b_k c_k(1-\cos d\varphi) + a_k \sin d\varphi & c_k{}^2(1-\cos d\varphi) + \cos d\varphi \end{bmatrix}$$

$$= \begin{bmatrix} \cos d\varphi & \sin\theta_{\Delta v} \cos\varphi_{\Delta v} \sin d\varphi & -\sin\varphi_{\Delta v} \sin d\varphi \\ \sin\theta_{\Delta v} \cos\varphi_{\Delta v} \sin d\varphi & \sin^2\varphi_{\Delta v}(1-\cos d\varphi) + \cos d\varphi & -\sin\theta_{\Delta v} \cos\varphi_{\Delta v} \sin\varphi_{\Delta v}(1-\cos d\varphi) \\ \sin\varphi_{\Delta v} \sin d\varphi & -\sin\varphi_{\Delta v} \sin\theta_{\Delta v} \cos\varphi_{\Delta v}(1-\cos d\varphi) & \sin^2\theta_{\Delta v} \cos^2\varphi_{\Delta v}(1-\cos d\varphi) + \cos d\varphi \end{bmatrix} \quad (11)$$



(a) Without Self-protection System (b) Zoom in (a)

(c) With the Self-protection System (d) Zoom in (c)

Fig. 4 Simulation Results

IV. SIMULATION

The simulation is based on Mat lab, version 7.0, and the result can be observed from every visual angle. The string formed of small spheres shows the robot's track while the string formed of big spheres presents obstacle's track. In Fig. 4, (a) and (b) show the condition that the robot do not own the self-protection system but robot in (c) and (d) possess the ability to avoid the dynamic obstacle. From Fig. 4 we can see that the robot changed its angle of velocity to avoid impact, which proves that the algorithm is correct and feasible. By this method, flying robot has the ability to protect itself from collision.

V. CONCLUSION

Aimed at developing an effective collision avoidance method in uncertain 3D dynamic environments, this paper provides a new method to realize self-protection ability of flying robots, by which the robot can effectively avoid some damages while moving in the space. Through the deduction and simulation of the algorithm, we can see that this method provide an effective approach for flying robots to avoid obstacles in 3D uncertain environments. Since the algorithm is based on sphere coordinates, the robot can move without any other auxiliary positioning systems to locate the obstacles and avoid them. This algorithm has been proved to be correct and feasible to realize on flying robots, and the sphere coordinate is easy to be established, thus lays a strong mathematic foundation for this algorithm.

REFERENCES

[1] Y.Wang, L. Han, M. Li, Q.Wang, J. Zhou, M. Cartmell, "A real-time path planning approach without the computation of Cspace obstacles," *Robotica*, vol. 22, pp. 173–187, 2004.
[2] O.Khatib, "Real-time obstacle for manipulators and mobile robot," *The Int. J. Robotic Research*, vol. 5, no. 1, pp. 90–99, 1986.
[3] Holland J.H., "Genetic algorithms and the optimal allocations of trails," *SIAM J. of Computing*, vol. 2, no. 2, pp. 88–105, 1973.
[4] Fox, D., Burgard, W. and Thrun, S., "The Dynamic Window Approach to Collision Avoidance," *IEEE J. Robotics and Automation*, vol. 4, no. 1, pp. 23–33, 1997.
[5] O. Brock, O. Khatib, "High-speed navigation using the global dynamic window approach," *Proc. the IEEE Int. Conf. on Robotics and Automation*, vol. 1, pp. 341–346, 1999.
[6] Xing-Jian Jing, "Behavior dynamics based motion planning of mobile robots in uncertain dynamic environments," *Robotics and Autonomous Systems*, vol. 53, pp. 99–123, 2005.
[7] Asimov I, Fredkel K A, *Robots: Machines in Man's Image*. New York: Harmony Books, 1985.
[8] M. Kise, Q. Zhang, N. Noguchi, "An Obstacle Identification Algorithm for a Laser Range Finder-Based Obstacle Detector," *Trans. the ASAE*, vol. 48, no. 3, pp. 1269–1278, 2005.
[9] Arras, K. O., N. Tomatis, B. T. Jensen and R. Siegwart, "Multisensor on-the-fly localization: Precision and reliability for applications," *Robotics and Autonomous Systems*, vol. 34, no. 2-3, pp. 131-143, 2001.

Guosheng Wu was born in Shandong Province, P. R. China, in 1981. Currently, he is a student in School of Automation Engineering at the University of Electronic Science and Technology of China. His current research interests include collision avoidance, motion planning and Robotics.

Luning Wang is a student in the Department of Computer Science at Beijing Institute of Technology, Beijing, CO100081, P. R. China. His current research interests include Robotics, Graphical User Interface.