Online ISSN 1848-3380, Print ISSN 0005-1144
ATKAFF 57(2), 428–440(2016)

Lei Liu, Yan Gao, Yuepeng Wu

Speed Optimization Control for Wheeled Robot Navigation with Obstacle Avoidance Based on Viability Theory

DOI 10.7305/automatika.2016.10.1274
UDK 007.52-026.731:656.052; 681.532:517.93

Original scientific paper

The navigation efficiency of wheeled robots needs to be further improved. Although related research has proposed various approaches, most of them describe the relationship between the robot and the obstacle roughly. Viability theory concerns the dynamic adaptation of evolutionary systems to the environment. Based on viability, we explore a method that involves robot dynamic model, environmental constraints and navigation control. The method can raise the efficiency of the navigation. We treat the environment as line segments to reduce the computational difficulty for building the viability condition constraints. Although there exists lots of control values which can be used to drive the robot safely to the goal, it is necessary to build an optimization model to select a more efficient control value for the navigation. Our simulation shows that viability theory can precisely describe the link between robotic dynamics and the obstacle, and thus can help the robot to achieve radical high speed navigation in an unknown environment.

Key words: Wheeled robot, Navigation, Obstacle avoidance, Viability

Optimizacija upravljanja brzinom mobilnog robota s izbjegavanjem prepreka zasnovana na teoriji vijabilnosti. Postoji potreba za unaprijeđenjem učinkovitosti navigacije mobilnih robota. Iako su vezana istraživanja predložila različite pristupe, većina ne opisuje precizno odnos između robota i prepreke. Teorija vijabilnosti istražuje dinamičke adaptacije evolucijskih sustava njihovoj okolini. U članku istražujemo metodu koja može povećati učinkovitost navigacije, zasnovanu na vijabilnosti koja uključuje dinamički model robota, ograničenja okoline robota i samu navigaciju. Radna okolina predstavljena je ravnim crtama kako bi se smanjila računska složenost izgradnje ograničenja. Iako postoji veliki broj iznosa upravljačkih veličina koje bi sigurno uputile robota prema cilju, potrebno je izraditi optimizacijski model koji bi odabrao učinkovitiju upravljačku vrijednost za navigaciju. Izvedene simulacije pokazuju da teorija vijabilnosti može precizno opisati vezu između prepreke i dinamike robota te na taj način pomoći robotu da postigne radikalno veće brzine pri navigaciji u nepoznatim prostorima.

Ključne riječi: mobilni robot, navigacija, izbjegavanje prepreka, vijabilnost

1 INTRODUCTION

Navigation control is an important topic for autonomous robots exploring tasks in unknown environments. In this study, the robot is expected to move from the starting location to the goal location without collision. The key component of robot navigation is the coordination between sensor data and robot driving control. There are two categories of navigation methods, namely deliberative methods and reactive methods. Deliberative methods are also known as global path planning. These methods are used for robots in environments where complete information about stationary obstacles and slowly moving obstacles are known in advance. Fundamental path planning methods are A*,D* searching methods, see for instance [20]. When complete information about the environment is not available in advance, mobile robots use the current sensor information to decide the next action, which has strong adaptability without prior knowledge of the environment. This is known as reactive methods, see [7]. We focus on the reactive methods since they are more suitable for unknown environment navigation.

Artificial Potential Fields(APF) is an early classical approach for navigation, it is widely used in mobile robotics [19]. The robot is driven by the negative gradient of a sum of potentials from different obstacles and the goal. After APF approach, the Vector Field Histogram (VFH) is developed in [5]. VFH+ uses less parameters than VFH to drive robots more smoothly and safely [30]. VFH-like methods transform obstacles into a histogram description. All free directions of the histogram are candidates for obstacle avoidance. An artificial weighted cost function is used to evaluate the candidate directions for goal guidance.

Then, optimal direction and velocity for navigation are obtained. The obtained direction and velocity need to be implemented by a driving control method, such as Proportion Integration Differentiation (PID) control or other nonlinear controls. Navigation approaches mentioned above belong to classical Nonmodel Based Method, and are not adequate for taking internal robot constraints like shape and dynamics into account, see [24]. It is rather difficult to control a robot traveling in a high speed with dynamic and mechanical constraints. If the acceleration and deceleration are limited, an attempt to turn at a high speed behind an obstacle easily results in collision. Another nonmodel based reactive method - fuzzy control is also widely used, see for instance [1]. This method is based on the human behavior, can be used for controlling many kinds of robots, but it is not precise enough.

In order to control robot safely in a higher speed, the robot dynamic model must be taken into account. Existing model based methods include Curvature Velocity Method(CVM), Dynamic Window Approach(DWA) and Lane Curvature Method, etc., see for instance [12, 28, 29]. All approaches mentioned above evaluate an objective function and assume that the robot moves along circular arcs. By limiting the searching space of admissible velocities, the problem induced by robot kinematic and dynamic constraints can be solved. While dynamic constraints have been taken into account, these methods are still lacking of accuracy. Taking DWA for instance, the limitation of the robot acceleration is determined by the operator's minds and adjustments. On the other hand, when encountering an obstacle, the robot should move in circular and have the ability to stop when it contacts the obstacle for safety. But in some situations, safety can also be realized without moving in circular and stopping at obstacle. For instance, if the robot moving direction is controlled to be parallel with the outline of the obstacle before crashing with it, safety can also be ensured, what's more, with higher efficiency.

Nowadays, evolutionary methods combined with classic approaches are increasingly used for navigation. Because classical approaches are effective but tend to be trapped in local minima when environment is complex. Evolutionary methods include Genetic Algorithm(GA), Ant Colony Optimization(ACO), Particle Swarm Optimization(PSO), etc. These methods are suitable for path planning in complex environment but act slowly. The combination of classical approaches and evolutionary methods can overcome each other's drawbacks. So improving the efficiency of the classical methods would help to improve the performance of the combined methods. And other studies of navigation, such as humanoid or bionic robot navigation [25, 34], machine vision aided navigation [11], multi robots cooperative navigation [8], 3-dimensions navigation [4],and so on, would benefit on it. Because these studies are more or less based on the classic methods. Besides evolutionary navigation methods, searching methods are also designed to overcome local minima, such as VFH* [31], DWA* [10], etc. For these two methods, A* searching is used to look ahead more steps in the local sensitive range for local minima avoidance. But the computation amount of these methods are large. In practice, we need to extract specific information from sensor data at first, then use these useful information for navigation.

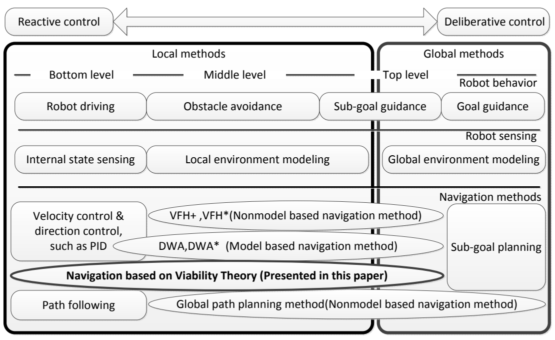

Reactive control
Deliberative control
Local methods
Global methods
Bottom level
Middle level
Top level
Robot behavior
Robot driving
Obstacle avoidance
Sub-goal guidance
Goal guidance
Robot sensing
Internal state sensing
Local environment modeling
Global environment modeling
Navigation methods
Velocity control & direction control, such as PID
VFH+ ,VFH*(Nonmodel based navigation method)
DWA,DWA* (Model based navigation method)
Sub-goal planning
Navigation based on Viability Theory (Presented in this paper)
Path following
Global path planning method(Nonmodel based navigation method)

Fig. 1. Navigation control structure and comparison of navigation methods

The structure of classical reactive methods is shown in Fig. 1. Navigation control can be divided into three levels: bottom level for robot driving, middle level for obstacle avoidance and top level for goal guidance [23]. Studies confirm that different level behaviors need different information. Goal guidance need global environmental information. Obstacle avoidance need local environmental details. Robot driving need internal states, such as velocity, acceleration,etc. As we mentioned above, the robot driving control faces the problem of dynamic constraint, such as acceleration limitation which may conflict with the velocity set values from the obstacle avoidance controller. Therefore, velocity set value of goal guidance control should be chosen under the control range constrained by robot dynamic and the environment. The classical reactive navigation methods are less accurate in general. These methods lack a theoretical support to precisely describe the relationship between environment and robot dynamics, this leads to gaps between each control levels, and the overall behavior of navigation tends to be slow.

Compared with classical reactive methods, global path planning methods which need complete global environmental details for a non-collision path to the goal must combine a robot driving control method to follow the planned path. But this path may be unable to be followed due to the constrains, such as acceleration limitation. Nonmodel methods like VFH series also have this disadvantage. For model based methods like DWA, both acceleration limitation and obstacle constraints are considered. But the assumption of a constant acceleration/deccelearation limitation dose not conform to the actual robot characters. Both VFH and DWA do not consider the real robot motor characters, so a robot control method such as PID must be combined for motor control.

The existing classical methods describe the relationship between environment and robot dynamics inaccurately. So the speeds of existing classical methods tend to be slow. Therefore, we propose a new control framework based on a new theory for robot navigation. This theory can precisely describe robot characters and the relationship between the robot dynamics and the environment. The framework based on this theory can include all three levels of navigation behavior, and integrates robot, goal and environment information all together.

Viability which is initially introduced by Aubin [2] concerns the dynamic adaptation of evolutionary systems to environment defined by constraints. Viability can be effectively used for system control, stability analysis, security design and other control areas. The outline of viability is: Considering a control system and a set, if for any initial point in this set, there exists a solution of the control system such that it stays in the set for ever. The set is said to be viable under this control system, see for instance [13].

At the moment, the application of viability is in its infancy. Because viability are based on set-valued analysis and nonsmooth analysis, it is elusive for engineering application. In [18], V stability theory which is similar to viability is used for the navigation of a sailing robot. The performance is excellent. But the interval approach for this application needs one minute to give a control signal. This is not suitable for indoor wheeled robot. In [32], a viability kernel is provided by Support Vector Machines(SVM) techniques to make the safety analysis of a high speed vehicle. But this application just uses some notions of viability, and only provide the dangerous warning to the driver. In [26], viability is applied for the aiming control of the submarine camera. This method can provide loose control signal for the underactuated systems, but a nonholonomic control is needed for bottom robot driving cascaded with viability method. The viability problem of linear system have been solved in [9],. The navigation method presented in [24] use Lyapunov function to construct a searching map for planar robot navigation. The performance of navigation is fast. This method is partly same as viability. The safety of robot is ensured by the Control Lyapunov Function theory [27], but this method would take much computing resource to build and search on a predefined map. Recently, we initially confirmed the feasibility of high speed obstacle avoidance of wheeled robots based on viability in [22] which is a preliminary work with a predefined simple environment and an approximate robot model.

We propose a method for robot navigation based on viability. If the robot is considered as a control system, obstacle as the boundary of the region (set). Thus, the process of robot's navigation with obstacle avoidance is just a viability design in the sense of control theory. We use LADAR to explore the unknown environment. The proposed framework is shown in Fig. 2. The raw LADAR data is processed to acquire security line segments which represent the geometric feature of the local environment. This process can help to build the position and velocity boundaries for viability. What's more, this process can avoid the local minima in the sensor's range. When the robot finds its internal states (position, velocity etc.) are on the viability boundaries, we introduce these information into the viability condition constrains module shown in Fig. 2. The module can give the precise admissible ranges of control constrained by self dynamics and environment based on viability. Then, we provides an efficient radical navigation control limited by predefined ranges. The objective function of the optimizer includes the goal's information.

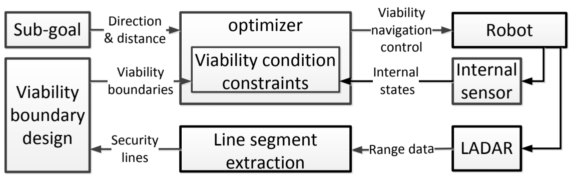


Fig. 2. The navigation control framework for wheeled robots based on the viability

According to the structure of the proposed framework in Fig.2, the paper is organized as follows. In the section 2, viability theory related to the robot is briefly introduced. Then, we design the viability boundary motivated by viability in the section 3. In the section 4, robot LADAR data is processed by line segment extraction method to build viability boundary. In section 5, the navigation optimizer based on viability condition constraints are explained in detail. At last, a simulation is made to demonstrate the feasibility of our method.

2 SOME NOTIONS OF VIABILITY

In this section, we introduce some notions on viability. For more information of viability, see [2, 13, 14, 16]. Consider a control system

$$\dot{x}(t) = f(x, u), u \in U, \tag{1}$$

where $x \in \mathbb{R}^n$ is state, $u \in U$ is control, $f : \mathbb{R}^n \to \mathbb{R}^n$, $g : \mathbb{R}^n \to \mathbb{R}^{m+n}$ are Lipschitzian.

Definition 1 [2] *Let $K \subset \mathbb{R}^n$. If for any initial point $x_0 \in K$ in this set, there exists a solution $x(t)$ of the control system (1), such that it stays in the set forever, $x(t) \in K, \forall t \geq 0$, then the set K is said to be viable under this control system.*

In order to get the viable condition, we first introduce the notion of tangent cone.

Definition 2 [2] *Let $K \subset \mathbb{R}^n$ be nonempty. The tangent cone of K at $x \in K$ is defined by:*

$$T_K(x) = \left\{ v \in \mathbb{R}^n | \lim_{t\to 0_+} \inf \frac{d_K(x+tv)}{t} = 0 \right\}, \tag{2}$$

where $d_K(y)$ is the distance of a point $y \in \mathbb{R}^n$ to the set K. In other words, $T_K(x)$ is the set of $v \in \mathbb{R}^n$ such that there exists a sequence of real numbers $t_k > 0, k = 1, 2, \ldots$, converging to 0 and a sequence of $v_k \in \mathbb{R}^n, k = 1, 2, \ldots$, converging to v satisfying

$$x + t_k v_k \in K, \forall k \geq 0. \tag{3}$$

Tangent cone is the generalization of tangent plane from smooth case to nonsmooth case. With this concept, the following theorem is on the viability condition:

Theorem 1 [2] *The closed set $K \subset \mathbb{R}^n$ is viable for the system (1) if and only if*

$$\Big(\bigcup_{u\in U} f(x,u)\Big) \bigcap T_k(x) \neq \varnothing, \forall x \in K. \tag{4}$$

where $\varnothing$ is empty.

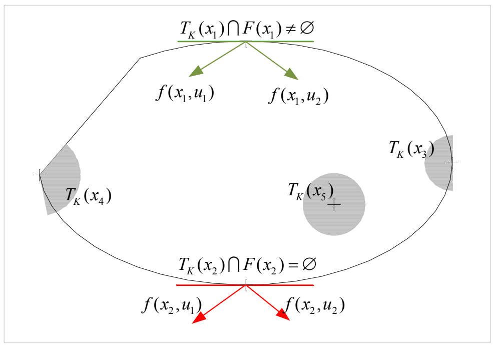


Fig. 3. Viability condition and tangent cone

In Fig.3, the set K is a nonsmooth ellipse. Tangent cones of states are marked as grey area in Fig. 3. Evidently, if state x is in the interior of K, $T_K(x) = \mathbb{R}^n$. Hence, (4) always holds for interior of K. In order to verify the viability by the Theorem 1, only states on the boundary need be considered. For instance, starting from boundary state x_1, at least two control can pull the system back to K. But unfortunately, when the system is on the boundary at state x_2 ,the evolution of the system can't go back to the environment with any available control. In order to use this theory on the navigation of robot, we face two challenges: The first one is to find the boundary of the environment; The second one is to evaluate the relationship between the tangent cone of boundary and the evolution of the system by robot computer in numerical way. It is difficult for us to determine the viability by Theorem 1 under a nonlinear system, see [13], especially to use viability in the engineering field, see [33]. For instance, Support Vector Machine(SVM) method is used for the viability kernel training in [32]. This method consumes lots of time and computing resources for viability kernel training, and the result boundary is rough. So the literature [32] only presents some qualitative safety analysis. But in [15], viability condition for an affine nonlinear control system is given in a region defined by inequality constraints based on the nonsmooth analysis. Fortunately, our wheeled robot is a kind of affine nonlinear system, and the free regions of robot can be represented by inequality constraints.

Consider an affine nonlinear system:

$$\dot{x}(t) = f(x) + g(x)u, u \in U, \tag{5}$$

where$x \in \mathbb{R}^n$ is state, $u \in U$ is system control, $f : \mathbb{R}^n \to \mathbb{R}^n$, $g : \mathbb{R}^n \to \mathbb{R}^{m+n}$are Lipschitzian. Generally there exists limitations on control value in real engineering problem, these limitations can be expressed by:

$$U = \big\{ u \subset \mathbb{R}^m | h_i(u) \leq 0, i = 1, \cdots, p) \big\}, \tag{6}$$

where $h_i : \mathbb{R}^m \to \mathbb{R}, i = 1, \cdots, p$ are convex (not necessary smooth). We suppose the robot environment is expressed by the following inequalities:

$$K = \big\{ x \in \mathbb{R}^n | \varphi_j(x) \leq 0, j = 1, \cdots, q) \big\}, \tag{7}$$

where $\varphi_j : \mathbb{R}^n \to \mathbb{R}, j = 1, \cdots, q$ are smooth function. φ_j represent the boundaries of obstacle. From the above analysis, we confirm that viability condition only cares the tangent cone at the boundary. Due to this reason, we must check whether the state x is on the boundary or not. Given a $x \in K$, define $J(x) = \big\{ 1 \leq j \leq q | \varphi_j(x) = 0, j = 1, \cdots, q) \big\}$. If $J(x)$ is empty set, the state x is interior of the K, otherwise it is a point on the boundary. Thus, the tangent cone of K at boundary state $x \in K$ is formulated by

$$T_K(x) = \{ y \in \mathbb{R}^n | \nabla \varphi_j(x)^{\mathrm{T}} y \leq 0, j \in J(x) \}. \tag{8}$$

Given a $x \in \mathbb{R}^n$, consider the following inequalities:

$$h_i(u) \leq 0, i = 1, \cdots, p,$$
$$\nabla\varphi_j(x)^{\mathrm{T}}(f(x) + g(x)u) \leq 0, j \in J(x). \tag{9}$$

where $u \in \mathbb{R}^m$ is variable.

Theorem 2 [15] *The set K is viable with respect to the system (5) if and only if the inequalities (9) has solutions for any $x \in K$.*

Theorem 2 transforms the viability condition Theorem 1 from geometric type to numerical type which is suitable for computing by the embedded computer. The first inequality of (9) gives the constraints of the control. The seond inequality shows the high order relationship of viability between the system (5) and boundaries (7).

3 WHEELED ROBOT DYNAMICS ANALYSIS AND VIABILITY CONTROL

We choose differential driving wheeled robot as the system, because it is wildly used.

3.1 Wheeled Robot Modeling

The existing classical navigation methods generally give the linear velocity and angular velocity set values of the robot, and use other methods to indirectly drive the robot motors. In order to control the motors directly, we introduce the motor dynamics into the model of the robot.The position and orientation of the robot are (x, y) and ϕ, see Fig. 4. The dynamic equations of the robot is

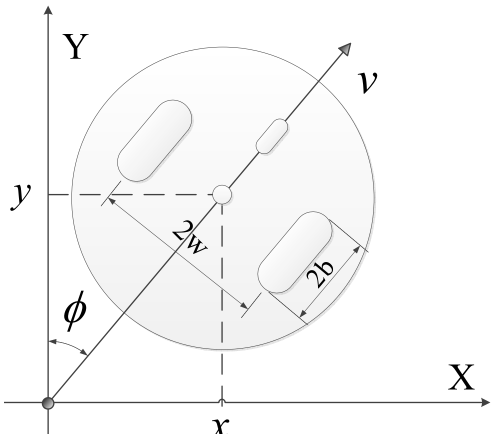


Fig. 4. Wheeled mobile robot model

presented as following:

$$\begin{cases} \dot{x} = v\sin(\phi), & \tau_{r1} = g(u_1 - kg\omega_{r1}), \\ \dot{y} = v\cos(\phi), & \tau_{r2} = g(u_2 - kg\omega_{r2}), \\ \dot{\phi} = \omega, & \omega_{r1} = (v + \omega w)/b, \\ \dot{v} = \beta_1(\tau_{r1} + \tau_{r2}), & \omega_{r2} = (v - \omega w)/b, \\ \dot{\omega} = \beta_2(\tau_{r1} - \tau_{r2}), & \end{cases} \tag{10}$$

where (x, y, ϕ) is robot position and orientation (pose) vector; (v, ω) are the linear velocity and the angular velocity; τ_{r1}, τ_{r2} are the left wheel, right wheel gearbox torques which are proportional to the motor coil current. We use k to represent the induced electromotive force coefficient; g is the ratio of motor gear box; (u_1, u_2) are control voltages; $(\omega_{r1}, \omega_{r2})$ are angular velocity of left and right wheels. We represent the motor torque by the difference between control voltage (u_1, u_2) and induced electromotive force $kg\omega_r$. This difference is proportional to the current of motor coil, because high order self-inductance is very small and can be ignored. Here the angular velocity of the motor is $g\omega_r$. In our model, we put resistance of the coil into k. $\beta_1 = 1/(bm)$, $\beta_2 = w/(bI)$, where m and I are the mass and the moment of the robot; $2w$ is the distance of two wheels; b is the radius of the wheel. System (10) can be rewritten as the following:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \\ \dot{v} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} v\sin(\phi) \\ v\cos(\phi) \\ \omega \\ -\frac{2kg^2}{b^2m}v \\ -\frac{2kg^2w^2}{b^2I}\omega \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{g}{bm}(u_1 + u_2) \\ \frac{gw}{bI}(u_1 - u_2) \end{bmatrix}. \tag{11}$$

Evidently, the constraints of the motor control voltage are

$$|u_j(t)| \leq u_{\max}, (j = 1, 2), \tag{12}$$

where $u_{\max}$ is the maximum control voltage of the motor. This value is limited by the voltage of motor controller. Traditional navigation methods frequently limit the linear and angular velocities or accelerations. We only limited the motor control signal. Because according to the robot model (10), the control (u_1, u_2) limitation can also limit the maximal velocities $(\dot{x}, \dot{y})$ and accelerations $(\dot{v}, \dot{\omega})$ indirectly. The environmental obstacles are denoted by the following formula:

$$G(x, y) \leq 0. \tag{13}$$

3.2 Viability Boundary Design

Since the control (u_1, u_2) only affects the robot speed (v, ω), and has no direct influence on the robot position (x, y). But the environmental boundary (13) only constrains the position (x, y). That means $\frac{\partial G}{\partial v} = 0$, $\frac{\partial G}{\partial \omega} = 0$. If (11) and the gradient of (13) are directly introduced into viability condition (9) as $f(x) + g(x)u$ and $\nabla\varphi_j(x)^{\mathrm{T}}$, the coefficients of (u_1, u_2) would become zero, the inequalities can not give the solution for (u_1, u_2). So the original boundary (13) should have more states of robot model (11). In other words, we should raise the dimension of (13) for viability control. i.e., we should limit both position and velocity of the robot by a new boundary with higher dimensions, which includes more robot model states motivated by (9). We design a new boundary as following:

$$\varphi_n = v_n t_{\text{set}} - l_n \leq 0, \tag{14}$$

where v_n is the robot normal velocity towards the boundary G; l_n is the distance from the robot to the boundary G; t_{set} is a setting parameter for the collision time, v_n is depended on the robot velocity, pose states and geometric feature of G, l_n is measured by the robot LADAR in the nearby area. It is not necessary to know the overall information of G to obtain boundary (14). Only the information of G in the detecting range of LADAR is enough. From the information viewpoint, the viability control belongs to classical reactive navigation.

The parameter t_{set} is used to set the elasticity of the viability boundary. Smaller the value is, closer the robot is to the obstacle when it begins the obstacle avoidance, and a larger control is needed. Since the control of robot (u_1, u_2) has constraint (12), the parameter t_{set} has a lower bound. On the contrary, when this parameter is set too large, the robot would get viability control far away from the obstacle, long distance LADAR information is needed. It is not necessary to avoid the remote obstacle during navigation. On the other hand, the long distance LADAR information contains many noises. From the viewpoint of designed boundary(14), the normal direction of the obstacle and normal distance between robot and the obstacle should be measured by LADAR. So we have to process original sensor data into a geometric feature to get the information of (14).

4 LADAR DATA PROCESSING FOR VIABILITY CONTROL

LADAR can obtain accurate range information of the environment near by the robot. This information is given in the form of polar coordinate points. The original data should be processed in order to suit for viability. Extracting geometric feature from the original data can save the computing resources. On the other hand, geometric feature contains information needed by the viability boundary (14). Here we use line segment feature extraction method to process the LADAR data instead of traditional methods which generally change the LADAR data into a histogram. The existing mature line feature extraction methods include Line Tracking (LT) algorithm , Iterative End Point Fit (IEPF) algorithm and Prototype based Fussy Clustering(PFC) algorithm, etc., see [6]. Our method is based on IEPF, because LADAR data can be processed at a high speed by IEPF. In Ref. [17], IEPF algorithm is used to identify road features on a high-speed unmanned vehicle which won the championship thereby. We process LADAR data in three steps: The first step is breakpoint detection; the second step is line extraction; the third step is security transformation.

4.1 Breakpoints Detection

The discontinuity of LADAR data can be adaptively found by the breakpoints detection. This discontinuity would be marked as a breakpoint. Let the scan depth value of two consecutive points i,$(i-1)$ be ρ_i, ρ_{i-1}. If the difference between the two values is greater than an adaptive threshold, these two consecutive scan points are a pair of breakpoints, see [6]. This processing is shown in algorithm 1.

Algorithm 1 Breakpoints detection for LADAR.

```
Input:
    The offset for sensor noise $\varepsilon$;
    The angular resolution of the sensor $\triangle\omega$;
    The threshold angle to extrapolate the maximum range
    value between $\rho_i$ and $\rho_{i-1}$,$\lambda$;
    The adaptive threshold between two adjacent scan
    point depths $D_{max}$.
Output:
    The breakpoints index array $\mathbb{BR}$.
1. FOR all scan points
2.     SET $D_{max}$ to $\min(\rho_i, \rho_{i-1})(\frac{\sin(\lambda)}{\sin(\lambda-\triangle\omega)} - 1) + \varepsilon$
3.     IF $|\rho_i - \rho_{i-1}| > D_{max}$ THEN
4.         The $i$ and $(i-1)$th points become breakpoints;
5.         SEND $i$ and $(i-1)$ to breakpoints array $\mathbb{BR}$;
6.     ENDIF
7. ENDFOR
```

4.2 Line Segment Extraction

In the step of line segment extraction, we convert all points of scan data from polar coordinate to cartesian coordinate. The starting and the ending points in a continuous sequence of points are connected as a line segment. Then the other scan points between these two endpoints are examined by our improved IEPF method for viability control. This method processes the convex and concave scan points differently, see Fig. 5. Assuming that LADAR is mounted at the center of the robot, the coordinate (0,0) of Fig. 5 represents the location of the robot. The raw LADAR data with noise is displayed by black square dots. Red cross points represent detected breakpoints. Pink dash line segments are the line features which are extracted from raw LADAR data using the improved IEPF method. Blue lines are obtained by rotating extracted

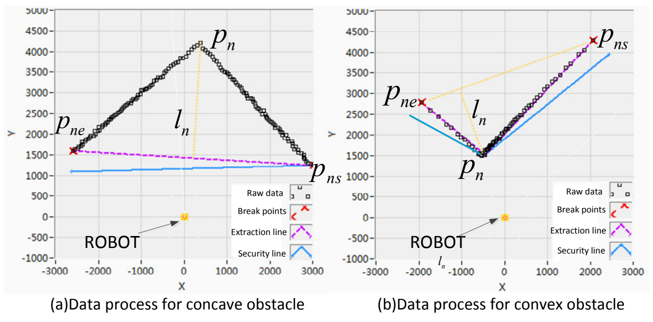


Fig. 5. LADAR Data processing

line segments, and are used for boundaries (13) in our navigation method. Fig. 5(a) shows data processing for concave obstacle. p_{ns}, p_{ne} are breakpoints of a continuous sequence of scan points. The scan points between p_{ns}, p_{ne} are behind the line segment $p_{ns}p_{ne}$ from the robot's view. When there is no goal in the corner, $p_{ns}p_{ne}$ is the extracted line segment. Fig. 5(b) shows data processing for convex obstacle. The data processing is similar with the traditional IEPF algorithm, see [17]. The algorithm splits scan points $P = \{p_{ns}, \cdots, p_n, \cdots, p_{ne}\}$ between two continuous breakpoints p_{ns}, p_{ne} into two subsets $P' = \{p_{ns}, \cdots, p_n\}$ and $P" = \{p_n, \cdots, p_{ns}\}$, when the maximum distance l_{max} from scan points p_n in the set P to the line $p_{ns}p_{ne}$ is greater than the threshold l^*_{Th}. Denote the line $p_{ns}p_{ne}$ as $Ax + By + C = 0$, $C \geqslant 0$, the distance l_n of point p_n can be calculated by the following:

$$l_n = \frac{Ax_n + By_n + C}{\sqrt{A^2 + B^2}}, \tag{15}$$

where x_n, y_n are the coordinates of scan point p_n. If the obstacle is convex, $l_n > 0$,and if the obstacle is concave, $l_n < 0$. Our improved IEPF method uses this difference to build line segments at different environmental cases. This method is shown in Algorithm 2.

4.3 Security Rotation

The real obstacle may extend beyond the extracted lines, because the raw LADAR data has noise and using line segments to represent real obstacle is inaccurate. In order to keep the robot safe, we need to move the extracted lines toward to the robot. Because viability boundaries (14) is sensitive to the normal direction of obstacle, we rotate the line segment with an angle around the end point nearer to the robot. A new line segment is achieved, see blue lines in Fig. 5. Such that not only the position of obstacle is changed for safety, but also a much more secure normal direction for the viability boundary is provided. We should manually adjust this angle, because this angle is decided by LADAR noise which may vary in different environment.

Algorithm 2 Improved IEPF method for viability control.

```
Input:
    The breakpoints index array, $\mathbb{BR}$; The minimum number of elements in a line segment $N^*_{\min}$;
    The threshold distance of scan points to endpoints line $l^*_{\mathsf{Th}}$.
Output:
    The line segments array $\mathbb{LS}$;
1.  SET $n_e$ to 1
2.  WHILE $n_e \leqslant NumberOfScanPoints$
3.    SET $n_s$ to $n_e$
4.    $n_e++$
5.    WHILE $n_e$ in not a breakpoint in $\mathbb{BR}$
6.      $n_e++$
7.        IF $n_e == NumberOfScanPoints$ THEN BREAK
8.      ENDIF
9.    ENDWHILE
10.   WHILE $n_e - n_s + 1 > N^*_{\min}$
11.       FOR all range values among start $n_s$ and last points $n_e$
12.         CALCULATE $l_n$ by (15)
13.           IF $l_n < 0$ && the goal is in the concave corner THEN $l_n = -l_n$
14.         ENDIF
15.     ENDFOR
16.     SET $l_{\max}$ to $\max\{l_1, \cdots, l_n, \cdots, l_{ne-ns}\}$
17.     IF $l_{\max} > l^*_{\mathsf{Th}}$ THEN
18.       SET $n_e$ to index of the $l_{\max}$ scan point
19.     ELSEIF
20.         $n_e$ and $n_s$ are sent to array $\mathbb{LS}$ as end points of a line segment
21.         BREAK
22.     ENDIF
23.   ENDWHILE
24. ENDWHILE
```

Wheeled robot should especially pay attention to the security line segment in front of it. This line can be used to build security boundary for obstacle control. On the other hand, doing this will reduce the details of the local environmental information greatly.

5 CONTROL OF ROBOT WITH LADAR

T_{sample} is denoted as scanning period of LADAR. At the scanning time k, a temporary trajectory frame is created. The initial state of the robot is $[x_{\text{t}}, y_{\text{t}}, \phi_{\text{t}}, v_{\text{t}}, \omega_{\text{t}}]^{\text{T}} = [0, 0, 0, v_{(k)}, \omega_{(k)}]^{\text{T}}$, where $v_{(k)}$, $\omega_{(k)}$ are the linear and angular velocities which are acquired by the robot sensor at time k. Then the LADAR data is processed in the trajectory frame. As mentioned in section IV, the security line segment in front of robot is important. We define this line in the trajectory frame with the equation $y_{\rm st} = ax_{\rm st} + c, c > 0$, where a is slope, c is the intercept value. This line segment can be designed for viability boundary according to (14), where

$$v_n = v_{\rm t}\frac{1}{\sqrt{a^2+1}}\cos(\phi_{\rm t}), \tag{16}$$

$$l_n = \frac{ax_{\rm t} - y_{\rm t} + c}{\sqrt{a^2+1}}. \tag{17}$$

The viability boundary can be expressed as following:

$$\varphi_n = v_{\rm t}\frac{1}{\sqrt{a^2+1}}\cos(\phi_{\rm t})t_{\rm set} - \frac{ax_{\rm t} - y_{\rm t} + c}{\sqrt{a^2+1}} \le 0, \tag{18}$$

At time k, when a proper parameter $t_{\rm set}$ is given, the robot finds its own state $[0, 0, 0, v_{(k)}, \omega_{(k)}]^{\rm T}$ is on or beyond the viability boundary (18). At this time, the robot dose not collide with the obstacle $y_{\rm st} = ax_{\rm st} + c, c > 0$. But the state of the robot has collision with the viability boundary (18) in the high dimension. So we need a piecewise constant control (u_1, u_2) to make the robot states back into the boundary (18). The admissible control range can be obtained by viability condition (9).

The viability boundary (18) only relates to the sates $x_{\rm t}$, $y_{\rm t}$, $\phi_{\rm t}$, $v_{\rm t}$ of the robot and does not include the state $\omega_{\rm t}$. In order to use viability condition (9), we should change the robot model to have the boundary (18) states $[x_{\rm t}, y_{\rm t}, \phi_{\rm t}, v_{\rm t}]^{\rm T}$. The gradient of viability boundary (18) related to these states is

$$\nabla\varphi_n = \begin{bmatrix} -\frac{a}{\sqrt{a^2+1}} \\ \frac{1}{\sqrt{a^2+1}} \\ -v_{\rm t}\frac{t_{\rm set}}{\sqrt{a^2+1}}\sin(\phi_{\rm t}) \\ \frac{t_{\rm set}}{\sqrt{a^2+1}}\cos(\phi_{\rm t}) \end{bmatrix}, \tag{19}$$

If $[\dot{x}_{\rm t}, \dot{y}_{\rm t}, \dot{\phi}_{\rm t}, \dot{v}_{\rm t}]^{\rm T}$, which includes control (u_1, u_2), is obtained, inequalities (9) would give the admissible control range. Then, an optimizer is designed to select an optimal control in the admissible control range for goal guidance.

5.1 Robot Modeling in Temporary Navigation Coordinate

For model (11), the differential equation for the state ω is

$$\dot{\omega} + \frac{2kg^2w^2}{b^2I}\omega = \frac{gw}{bI}(u_1 - u_2). \tag{20}$$

The solution of the equation (20)is

$$\omega(t) = \frac{b}{2kgw}(u_1 - u_2) + Ce^{-\frac{2kg^2w^2}{b^2I}t}, \tag{21}$$

where C is a constant. Since viability control is piecewise constant at each scan time. The constant C can be solved by initial state $\omega_{(k)}$. Substitute $\omega_{(k)}$ into (21).

$$\omega_{(k)} = \frac{b}{2kgw}(u_1 - u_2) + \lim_{t\to 0} C_k e^{-\frac{2kg^2w^2}{b^2I}t}, \tag{22}$$

where C_k is the constant at time k and admits

$$C_k = \omega_{(k)} - \frac{b}{2kgw}(u_1 - u_2). \tag{23}$$

Therefor the robot model in temporary navigation coordinate frame at time k is

$$\begin{bmatrix} \dot{x}_{\rm t} \\ \dot{y}_{\rm t} \\ \dot{\phi}_{\rm t} \\ \dot{v}_{\rm t} \end{bmatrix} = \begin{bmatrix} v_{\rm t}\sin(\phi_{\rm t}) \\ v_{\rm t}\cos(\phi_{\rm t}) \\ \omega_{\rm t} \\ -\frac{2kg^2}{b^2m}v_{\rm t} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{g}{bm}(u_{1(k)} + u_{2(k)}) \end{bmatrix} \tag{24}$$

where $\omega_{\rm t}$ can be calculated by the following equation:

$$\dot{\phi}_{\rm t} = \omega_{\rm t} = \beta(u_{1(k)} - u_{2(k)})(1 - e^{-\alpha t}) + \omega_{(k)}e^{-\alpha t}, \tag{25}$$

where

$$\alpha = \frac{2kg^2w^2}{b^2I}, \beta = \frac{b}{2kgw}. \tag{26}$$

$(u_{1(k)}, u_{2(k)})$ is the control at time k. Now we change the robot model to (24) which are suitable for (9) with (19).

5.2 Viability Constraints

We observe that (19) is the normal vector of viability boundary in state space $[x_{\rm t}, y_{\rm t}, \phi_{\rm t}, v_{\rm t}]^{\rm T}$, $[\dot{x}_{\rm t}, \dot{y}_{\rm t}, \dot{\phi}_{\rm t}, \dot{v}_{\rm t}]^{\rm T}$ is the derivative of model (24) states. According to (9), the dot product of (24) and (19) should be less than 0. Theorem 2 means when robot states are on the boundary (18), there should exist a control which can make robot states' velocities be included in the boundary tangent cone for viability. This is the relationship between robot and environment described by viability.

Considering robot model (25), the steering control $(u_1 - u_2)$ needs time to affect the angular velocity $\dot{\phi}_{\rm t}$. In order to introduce the steering control into inequalities (9), a short operation time $\triangle t < T_{\text{sample}}$ is introduced. According to (25), the angular velocity of time $(k + \triangle t)$ is

$$\dot{\phi}_{t(k+\triangle t)} = \beta(u_{1(k)} - u_{2(k)})(1 - e^{-\alpha\triangle t}) + \omega_{(k)}e^{-\alpha\triangle t}. \quad (27)$$

Replacing $\dot{\phi}_{\text{t}(k)}$ by $\dot{\phi}_{\text{t}(k+\triangle t)}$ in viability condition (9), we have

$$\nabla\varphi_{\text{n}(k)}[\dot{x}_{\text{t}(k)} \;\; \dot{y}_{\text{t}(k)} \;\; \dot{\phi}_{\text{t}(k+\triangle t)} \;\; \dot{v}_{\text{t}(k)}] < 0. \quad (28)$$

The steering control $(u_1 - u_2)$ is introduced to inequalities (28) by $\dot{\phi}_{\text{t}(k+\triangle t)}$. When the robot state is on or beyond the viability boundary (18), the solution control of the inequalities (28) can make the robot in the viability boundary (18) in a short time $\triangle t$. Introducing $\dot{\phi}_{\text{t}(k+\triangle t)}$ instead of $\dot{\phi}_{\text{t}(k)}$ into (28) has a error with upper bound. Shorter the operation time $\triangle t$ is, smaller this error is. Since $\triangle t$ is short, the orientation angle at time $k + \triangle t$ is $\phi_{\text{t}(k+\triangle t)} \approx \phi_{\text{t}(k)} + \dot{\phi}_{\text{t}(k+\triangle t)}\triangle t \approx 0$, and $\sin(\phi_{\text{t}(k+\triangle t)}) \approx \phi_{\text{t}(k+\triangle t)} \approx \dot{\phi}_{\text{t}(k+\triangle t)} \triangle t$, $\cos(\phi_{\text{t}(k+\triangle t)}) \approx 1$. The derivative of robot states for viability control becomes

$$\begin{aligned}&[v_{\text{t}(k)}\dot{\phi}_{\text{t}(k+\triangle t)},\; v_{\text{t}(k)},\; \dot{\phi}_{\text{t}(k+\triangle t)},\\ &(\frac{g}{bm}(u_{1(k)} + u_{2(k)}) - \frac{2kg^2}{b^2m}v_{\text{t}(k)})]^{\text{T}}\end{aligned} \quad (29)$$

$\frac{\partial\varphi_{\text{n}(k)}}{\partial\phi_{\text{t}(k)}} = -v_{\text{t}(k)}\frac{t_{\text{set}}}{\sqrt{a^2+1}}\sin(\phi_{\text{t}(k)})$ in $\nabla\varphi_{\text{n}(k)}$ of (28) also need to use the gradient at time $k + \triangle t$. Thus, the gradient of φ_{n} at time k is reformulated as

$$\nabla\varphi_{\text{n}(k)} = \begin{bmatrix} -\frac{a}{\sqrt{a^2+1}} \\ \frac{1}{\sqrt{a^2+1}} \\ -v_{\text{t}(k)}\frac{t_{\text{set}}}{\sqrt{a^2+1}}\dot{\phi}_{\text{t}(k+\triangle t)} \triangle t \\ \frac{t_{\text{set}}}{\sqrt{a^2+1}} \end{bmatrix}. \quad (30)$$

Substituting both (29) and (30) into (28), the viability condition is reformulated as

$$\begin{aligned}&-\frac{av_{\text{t}(k)}\dot{\phi}_{\text{t}(k+\triangle t)}}{\sqrt{a^2+1}} + \frac{v_{\text{t}(k)}}{\sqrt{a^2+1}}\\ &-\frac{v_{\text{t}(k)}t_{\text{set}}[\beta(u_{1(k)}-u_{2(k)})(1-e^{-\alpha\triangle t})+\omega_{(k)}e^{-\alpha\triangle t}]^2\triangle t}{\sqrt{a^2+1}}\\ &-\frac{t_{\text{set}}[\frac{2kg^2}{b^2m}v_{\text{t}(k)}-\frac{g}{bm}(u_{1(k)}+u_{2(k)})]}{\sqrt{a^2+1}} < 0,\end{aligned} \quad (31)$$

The first term in (31) represents the tendency of the robot running out of the viability boundary (18) in X direction. The second term represents the Y direction tendency. The third term supplies negative value for the inequalities by using steering control $(u_{1(k)} - u_{2(k)})$. The fourth term supplies negative value by decreasing $(u_{1(k)} + u_{2(k)})$ for deceleration. Except variables $(u_{1(k)}, u_{2(k)})$, other parameters are all robot states which can be obtained by the sensor.

We expect the robot to keep a high speed while avoiding obstacles. robot need to achieve the goal at the end, so we should build an objective function for the goal. Building a new objective function for goal guidance and speed optimization is a challenge. Thus, we assign the derivative $\dot{v}_{\text{t}(k)}$ to 0. This means that the deceleration method for the viability control is omitted. The derivative of robot state for viability control at time k becomes

$$\begin{aligned}&[\dot{x}_{\text{t}(k)} \; \dot{y}_{\text{t}(k)} \; \dot{\phi}_{\text{t}(k+\triangle t)} \; \dot{v}_{\text{t}(k)}]^{\text{T}} =\\ &[v_{\text{t}(k)}\dot{\phi}_{\text{t}(k+\triangle t)},\; v_{\text{t}(k)},\; \dot{\phi}_{\text{t}(k+\triangle t)},\; 0].\end{aligned} \quad (32)$$

The viability condition of (31) becomes:

$$\begin{aligned}&-\frac{av_{\text{t}(k)}\dot{\phi}_{\text{t}(k+\triangle t)}}{\sqrt{a^2+1}} + \frac{v_{\text{t}(k)}}{\sqrt{a^2+1}} -\\ &\frac{v_{\text{t}(k)}t_{\text{set}}[\beta(u_{1(k)}-u_{2(k)})(1-e^{-\alpha\triangle t})+\omega_{(k)}e^{-\alpha\triangle t}]^2\triangle t}{\sqrt{a^2+1}} < 0,\end{aligned} \quad (33)$$

To satisfy the inequalities (33), we have to use steering control. In (31), a negative fourth term which means deceleration can help the system viable. When this term is set to be 0, the steering control $(u_{1(k)} - u_{2(k)})$ need to be increased to enlarge the negative value of the third term in (31) for viability. When the robot encounters the viability boundary at a high speed, the solution of (33) would decelerate one motor to increase $(u_{1(k)} - u_{2(k)})$ for turning. This will lead to decreasing the linear velocity. Therefor, the control $(u_{1(k)}, u_{2(k)})$ for (33) will ensure the fourth term of (31) negative. This leads the robot to be viable.

When the parameter t_{set} is smaller, $(u_{1(k)} - u_{2(k)})$ need to be larger enough to make the robot to go back to the viable region in $\triangle t < T_{\text{sample}}$. Noticing that there have constraints of control in (12), the viability boundary parameter t_{set} must be manually adjusted to suit for the constraints of control. Since the LADAR scanning period T_{sample} is short, $\triangle t$ would be small. $\triangle t$ is important, because it gives steering control $(u_{1(k)} - u_{2(k)})$ a chance to affect $\phi_{\text{t}(k+\triangle t)} \approx \phi_{\text{t}(k)} + \dot{\phi}_{\text{t}(k+\triangle t)} \triangle t$ in one control iteration. For nonholonomic constraint, the robot is not able to instantaneously move in the same direction with the line that passes through the axis of the driven wheels. As we mentioned above in trajectory frame at time k, $\phi_{\text{t}(k)} = 0$. That means the robot is not able to instantaneously move along X axis. But with the parameter $\triangle t$, steering control can make the robot moving along the X axis by $\phi_{\text{t}(k+\triangle t)}$ with limited error. Of course, the parameter $\triangle t$ have a lower bound. A smaller $\triangle t$ means the robot need a larger steering control to go back into the variable area in $\triangle t$. Since the real time controller would keep the control output unchanged in each control period which is equal to T_{sample}, small $\triangle t$ may lead to the robot vibration near the viability boundary. So the parameter $\triangle t$ should be chosen properly by the performance of the control.

5.3 Optimization Model of Navigation Control

The navigation control of wheeled robots includes two types of behavior: obstacle avoidance and goal guidance. Viability condition (9) gives an admissible control range by inequalities for obstacle avoidance. We can design an optimization model for goal guidance. This change a navigation problem to an optimization problem. The optimization model for the robot navigation is designed as the following:

$$(\mathbf{P}) \quad \text{minimize } (\theta_e - \dot{\phi}_s T_{\text{turning}})^2 \tag{34}$$

$$\text{subject to} \begin{cases} \text{motor control constraint (12)}, \\ \text{constraint (33) when state is out of (18)}, \end{cases}$$

where $\dot{\phi}_s = \beta(u_1 - u_2)$ is the stable orientation angle velocity of the robot, the value of β is presented in (26). T_{turning} denotes the turning time parameter for the robot turning toward the goal. Here we use $\dot{\phi}_s T_{\text{turning}}$ to represent the expected direction of the robot orientation angle. Let θ be the target guiding angle and ϕ be the robot orientation angle presented in (10), the guidance error $\theta_e \in [-\pi, \pi]$ would be $(\phi - \theta)$. The objective function can be written as:

$$\text{minimum } ((\phi - \theta) - \beta(u_1 - u_2)T_{\text{turning}})^2 .$$

We use T_{turning} to adjust the intensity of goal guidance. Smaller turing is, faster the robot turns to the goal. But the overall speed of the robot navigation slows down. This value can be adjusted according to the distance between the robot and the goal. When the distance is far, a bigger value would be used in order to ensure a high speed of the robot. When the goal is near, a smaller value would be used to ensure the robot guiding towards the goal. We also monitor the value of polynomial (31) using the optimized control (u_1, u_2) obtained from (34). A positive value of polynomial (31) means only steering control can't ensure the safety of robot, wheel breaking system should also be used.

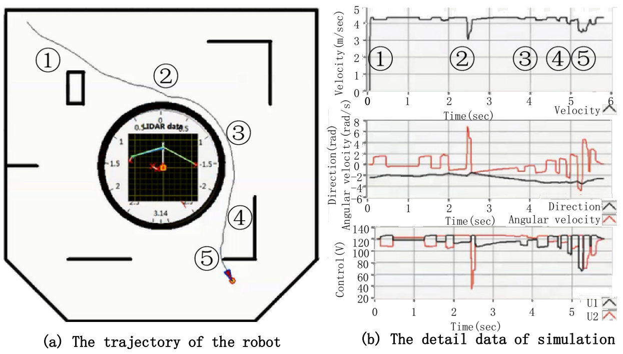
The motivation of this work is to increase the speed of the robot navigation, because it is important for improving the efficiency of the robot transportion. That is why optimization model (34) only optimize the steering control. When the robot approach the goal, it can use other mature navigation method, such as VFH+, for deceleration and stopping at the goal.

If the robot states encounter a viability boundary, we set the farther end point of the front security line as the subgoal which information is used for (34). When the robot is obstacle free, we directly use the original goal information for (34). This can effectively avoid local minima.

5.4 Simulation of Robot Navigation with LADAR

Based on the data process method of LADAR presented in section IV, the useful information of raw LADAR data is not only the position, but also the tendency of obstacles. So we give a more complex environment to test the proposed method. The environment for simulation is shown in Fig. 6(a). Viability control avoids the obstacle at ①, ②, ③, ④, ⑤, and finally achieve the goal drawn as the yellow dot. We use our IEPF method to process the LADAR data. The optimization problem (34) is solved by Sequential Quadratic Programming(SQP).

The simulation environment is built by National Instrument company softwares: Labview, NI robotic module, NI control and simulation module and other related toolkits of Labview. The length and width of the simulation space is 20 meters, the diameter of the centered circular obstacle is 8 meters. Robot simulation parameters are listed as follows: mass of the robot m is 43.5 kg; inertia moment of rotation I is 1.16 kg·m 2 ; radius of wheels b is 0.0768 m; distance between two wheels $2w$ is 0.448 m; ratio of the gear box g is 10.6; induced electromotive force constant k is 0.2; the maximum peak control of the motor u_{max} is 126. The LADAR start angle is $-120°$; end angle is $120°$. The simulated LADAR contains 240 scan lines with 30mm noise, LADAR sample period T_{sample} is 0.1sec, this period equals to the period of viability control. We adjust $\triangle t$ to 0.05 sec. The simulated controller can run SQP in real time.

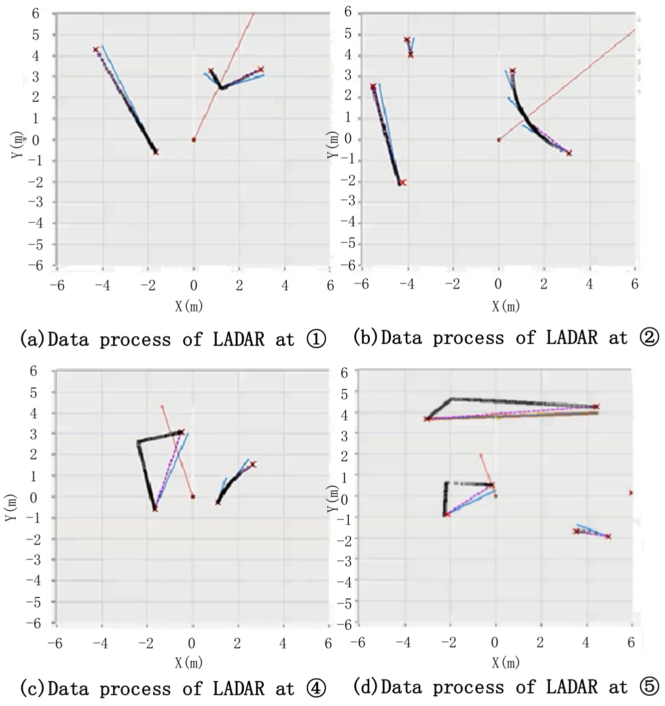


(a) The trajectory of the robot (b) The detail data of simulation

Fig. 6. The simulation of wheeled mobile robot viability navigation based on LADAR

Fig. 6(b) shows the velocity(m/sec) vs. time(sec) graph on the top, the direction(rad) and angular velocity(rad/s) vs. time(sec) graph in the middle and the motor control(V) vs. time(sec) graph at the bottom. In the middle graph, the black line is the direction, the red line is the angular velocity. In the bottom graph, the black line is the control voltage of the right wheel, the red line is the control voltage of the left wheel.

The robot navigates in the simulation environment with a high speed which is at least 4 meters per second as shown in the Velocity vs. Time graph of Fig. 6(b). The black line of the Direction vs. Time graph shows the turning control of robot is smooth. For the Control vs. Time graph, at least one motor is controlled by a voltage that is slightly lower than or equal to u_{max} in the navigation process. This means that there is always a motor keep running at the full speed during obstacle avoidance. The other motor is controlled by SQP method based on viability control and realizes the corresponding steering behavior for obstacle avoidance. The curves in Control vs. Time graph sometimes are sharp, because the LADAR simulation has noise and security lines often change with robot moving. These facts and discretization error disturb the smooth output of the controller.



(a)Data process of LADAR at ① (b)Data process of LADAR at ② (c)Data process of LADAR at ④ (d)Data process of LADAR at ⑤

Fig. 7. LADAR data processing at the places of 1, 2, 4, 5 in simulation.

In Fig. 6, robot turns left to avoid non-smooth convex obstacle at ① and turns left to avoid non-linear convex circular obstacle at ②. Robot detours around the circular obstacle at ③. The concave non-smooth object is placed at ④. Finally the robot avoids the thin barrier and achieves the goal at ⑤. The robot LADAR data process at places of ①, ②, ④, ⑤ is shown in Fig. 7. The elements meaning of Fig. 7 is similar with that of Fig. 5. Only a red thin line indicating the goal direction is added. Data process at ① is shown in Fig. 7(a). The square obstacle is transformed to blue line segments to represent non-smooth convex obstacles; The data process of the circular obstacle is shown in Fig. 7(b). Here we use three outward blue line segments to represent the original circular obstacle. The non-smooth concave obstacles data process at ④ is shown in Fig. 7(c). We use line segment connecting two breakpoints instead of the raw data. Using this method,useless and trivial information from LADAR can be filtered. The circumstance near the end of navigation is shown in Fig. 7(d).

6 CONCLUSION

We propose a new navigation method for wheeled robot with LADAR in an unknown environment based on viability. Viability provides a precise description of the relationship between a dynamic system and the environment. When guaranteing viability conditions, the robot can barely reached the maximum speed with an optimized control. Viability gives us a guidance on how to deal with the sensor information for a dynamic system. Our LADAR data process presented in section IV conforms with the viability condition requirements. Based on viability, the boundary of dynamic system should have the same dimensions as state space of the robot. So we raise the dimension of original position boundary of security line segments by defining a time parameter t_{set} which introduces velocity into the designed boundary for viability control. By using parameter $\triangle t$, we solve the control problem of nonholonomic constraint of the robot. At last, we design an optimization model to find a more efficient control for the goal by omitting the deceleration item of the viability condition. All these works constitute a framework shown as Fig. 2.

We find that there have some shortcoming with our method. When we discretize our controller, a discretization error exists disturbing the smooth outcome of the controller. In our work, we use small iteration time t_{sample} to minimize this error. For the future work, we would develop a discretization method to eliminate this error. Other improvement would be on the LADAR data process. We only process LADAR data for necessary information required by viability condition, so breakpoints often abruptly emerge when robot moves, and a new security line segment suddenly appears. This leads the controller gives out a sharp control signal. We would find a more smooth data process method for this problem in our future work.

Compared with classical navigation methods, our method needs smaller computation resources and is more efficient in data processing. Viability control serves as a basic theory for wheeled robot navigation. Theoretically, other kinds of robots may also enjoy the benefit of viability in improving the performance of navigation.

REFERENCES

[1] Amoozgar M H, Alipour K, Sadati S H, A fuzzy logic-based formation controller for wheeled mobile robots, Industrial Robot-An International Journal, 2012, 38(3):269-281.

[2] Aubin J P, Viability Theory, Boston: Birkhauser, 1991.

[3] Aurenhammer F, Voronoi diagrams—a survey of a fundamental geometric data structure, ACM Computing Surveys (CSUR), 1991, 23(3): 345-405.

[4] Belkhouche F, Bendjilali B. Reactive path planning for 3-D autonomous vehicles, IEEE Transactions on Control Systems Technology, 2012, 20(1):249-256.

[5] Borenstein J, Koren Y, The vector field histogram: Fast obstacle avoidance for mobile robots, IEEE Transactions on Robotics and Automation, 1991, 7(3): 278-288.

[6] Borges G A, Aldon M J, Line extraction in 2D range images for mobile robotics, Journal of Intelligent Robot System, 2004, 40(3): 267-297.

[7] Brandao A S, Sasaki A S, Castelano C R, et al, Autonomous Navigation with Obstacle Avoidance for a Car-Like Robot, Robotics Symposium and Latin American Robotics Symposium (SBR-LARS), IEEE, 2012: 156-161.

[8] Bretl T. Minimum-time optimal control of many robots that move in the same direction at different Speeds. IEEE Transactions on Robotics, 2012, 28(2):351-363.

[9] Chen Z, Gao Y. Determining the viable unbounded polyhedron under linear control systems. Asian Journal of Control, 2014, 16 (6):1-7.

[10] Chou C C, Lian F L, Wang C C, Characterizing indoor environment for robot navigation using velocity space approach with region analysis and lookaheadverification, IEEE Transactions on Instrumentaion and Measurement, 2011, 60(2): 442-451.

[11] Croon G C, Weerdt E, Wagter C. The appearance variation cue for obstacle avoidance. IEEE Transitions on Robotics, 2012, 28(2):529-534.

[12] Fox D, Burgard W, Thrun S, The dynamic window approach to collision avoidance, IEEE Robotics and Automation Magazine, 1997, 4(1): 23-33.

[13] Gao Y, Viability criteria for differential inclusions, Journal of Systems Science and Complexity, 2011, 24: 825-834.

[14] Gao Y, Determining the viability for a class of nonlinear control systems on a region with nonsmooth boundary(in Chinese), Control and Decision, 2006, 21(8):12-30.

[15] Gao Y, Determining the viability for a affine nonlinear control system(in Chinese), Journal of Control Theory and Applications, 2009, 26(6):654-656.

[16] Gao Y, Lygeros J, Quincampoix M. On the reachability problem of uncertain hybrid systems, IEEE Transactions on Automatic Control, 2007, 52(9): 1572-1586.

[17] Han J, Kim D, Lee M, Enhanced road boundary and obstacle detection using a downward-looking LIDAR sensor, IEEE Transactions on Vehicular Techology, 2012, 61(3); 971-985.

[18] Jaulin L, Bars F L. An interval approach for stability analysis: application to sailboat robotics. IEEE Transactions on Robotics, 2013, 29(1):282-287.

[19] Khatib O, Real-time obstacle avoidance for robot manipulator and mobile robots, The International Journal of Robotics Research, 1986, 5(1): 90-98.

[20] Likhachev M, Ferguson D I, Gordon G J, et al, Anytime Dynamic A*: An anytime, replanning algorithm, Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS), 2005: 262-271.

[21] Liu L, Xu X M, The design of simulation software for autonomous navigation and obstacle avoidance mobile robot(in Chinese), Journal of Huazhong University of Sci and Tech, 2011, 39(Sup.II):196-199.

[22] Liu L, Gao Y, Wu Y P, High speed obstacle avoidance control of wheeled mobile robots with nonhomonymic constraint based on viability theory(in Chinese), Control and Decision, 2014, 29(9):1623-1627.

[23] Lwin Y Y, Yamamoto Y, Obstacle-responsive navigation scheme of a wheeled mobile robot based on lookahead control. Industrial Robot-An International Journal, 2012,39(3):282-293.

[24] Ögren P, Leonard N E, A convergent dynamic window approach to obstacle avoidance, IEEE Transaction on Robotics, 2005, 21(2): 188-195.

[25] Pal L, Kristion Y, Oyvind S. Snake robot locomotion in environments with obstacles. IEEE/ASME Transactions on Mechatronics, 2012, 17(6):1158-1169.

[26] Panagou D, Kyriakopoulos, Viability control for a class of underactuated systems, Automatica, 2013, 49(1):17-29.

[27] Primbs J A, Nevistic V, Doyle J C, Nonlinear optimal control: A control Lyapunov function and receding horizon perspective, Asian Journal of Control, 1999, 1(1):14-24.

[28] Rebai K, Azouaoui O, Benmami M, et al, Car-like robot navigation at high speed, Proceddings of IEEE International Conference on Robotics and Biomimetics(ROBIO), 2007: 2053-2057.

[29] Simmons R, The curvature-velocity method for local obstacle avoidance, Proceedings of IEEE International Conference on Robotics and Automation(ICRA'96), 1996, 4: 3375-3382.

[30] Ulrich I, Borenstein J, VFH+: Reliable obstacle avoidance for fast mobile robots, Proceedings of IEEE International Conference on Robotics and Automation(ICRA'98), 1998, 2: 1572-1577.

[31] Ulrich I, Borenstein J, VFH*: Local obstacle avoidance with look-ahead verification, Proceedings of IEEE International Conference on Robotics ans Automation (ICRA'00), 2000, 3: 2505-2511.

[32] Vandanjon P O, Coiret A, Lorino T, Viability theory and road safety, Vehicle System Dynamics, 2013.

[33] Wieber P B, Viability and predictive control for safe locomotion, Proceedings of IEEE/RSJ International Conference on Intelligent Robots and System(IROS), 2008: 1103-1108.

[34] Yoo J K, Kim J H. Fuzzy integral-based gaze control architecture incorporated with modified-univector field-based navigation for humanoid robots. IEEE Transactions on Systems, Man, and Cybernetics PART B: Cybernetics, 2012, 42(1):125-139.

ACKNOWLEDGMENT

This research is supported by the National Natural Science Foundation of China under grant No. 11171221, Doctoral Program Foundation of Institutions of Higher Education of China under grant No. 20123120110004, Shanghai First Class Discipline Construction Project under grant No. XTKX2012 and Doctoral Fund of University of Shanghai for Science and Technology under grant No. 1D-10-303-002.

Lei Liu received his Ph.D. from the University of Shanghai for Science and Technology in 2013. Now he is a lecturer of School of Management, University of Shanghai for Science and Technology. His research interests include control, simulation and modelling of various robots such as wheeled robots, industrial robots and vehicles, etc.

Yan Gao received his Ph.D. degree from Dalian University of Technology in 1996. Now he is a full professor of School of Management, University of Shanghai for Science and Technology. He has published 250 journal papers, and authored 2 books. His research interests include nonsmooth optimization, systems engineering and hybrid system control.

Yuepeng Wu received the M.S. degree from University of Shanghai for Science and Technology in 2009 and he is a doctoral candidate in University of Shanghai for Science and Technology. His research interests include control theory and robotic system and robotic system.

AUTHORS' ADDRESSES

Lei Liu, Ph.D.,
Prof. Yan Gao, Ph.D.,
School of Management,
University of Shanghai for Science and Technology,
516 Jungong Road, Shanghai, 200093 China
e-mail: liulei@usst.edu.cn, Gaoyan@usst.edu.cn
Yuepeng Wu, M.S.
School of Optical-Electrical,
University of Shanghai for Science and Technology,
516 Jungong Road, Shanghai, 200093 China
e-mail: some_liu@hotmail.com

Received: 2015-04-07
Accepted: 2015-10-30