中央研究院
資訊科學研究所
Institute of Information Science, Academia Sinica • Taipei, Taiwan, ROC

TR-IIS-18-004

Real-Time Hydrodynamic-based Obstacle Avoidance for Nonholonomic Mobile Robots with Curvature Constraints*

Pei-Li Kuo, Chung-Hsun Wang, Han-Jung Chou and Jing-Sin Liu

Sep. 12, 2018 || Technical Report No. TR-IIS-18-004

http://www.iis.sinica.edu.tw/page/library/TechReport/tr2018/tr18.html

Real-Time Hydrodynamic-based Obstacle Avoidance for Nonholonomic Mobile Robots with Curvature Constraints*

Pei-Li Kuo, Chung-Hsun Wang, Han-Jung Chou and

Jing-Sin Liu

Institute of Information Science, Academia Sinica, Nangang, Taipei, Taiwan 115, ROC
r01525004@ntu.edu.tw, r02221012@gmail.com, frankwang50302@gmail.com,liu@iis.sinica.edu.tw

Abstract—The harmonic potential field of an incompressible fluid governed by Laplace equation is potential for mobile robots to generate smooth, natural-looking paths for obstacle avoidance. The streamlines generated by boundary value problem of Laplace equation have explicit or analytic vector field as the path tangent or robot heading specification without the waypoints. This paper presents an implementation of on-line obstacle avoidance system for nonholonomic curvature-constrained mobile robot regarded as a particle moving along smooth path primitives composed by streamlines. In combination with streamline path approaches, the proposed approach to generate an obstacle avoidance path satisfies nonholonomic constraint by pure pursuit algorithm. First, we use the potential flow field around a circle to derive three primitive curvature- constrained paths to avoid single obstacle. Furthermore, pure pursuit controller is implemented to achieve a smooth transition between the streamline paths in the environment with multiple obstacles. In addition to simulations, a proof of concept experiment implemented on a two-wheel driving mobile robot with range sensors validates that the proposed hydrodynamic path planning algorithm is able to on-line generate a path with a lower maximum curvature not violating curvature constraint to navigate smoothly and safely among multiple cylinder obstacles in partially unknown cluttered environments.

Keywords: harmonic potential field, curvature constraint, obstacle avoidance, nonholonomic mobile robot

*Part of this work was presented in *6th International Symposium on Advanced Control of Industrial Processes, Taipei, Taiwan, June 2017*, and *2017 International Conference on Advanced Mechatronic Systems, Xiamen, China ,December 2017*.

1. Introduction

Due to advances in sensing and computing technology, deployment and application of autonomous mobile robots are prevalent as platforms for missions such as search and rescue, autonomous driving, manufacturing in cluttered environments [1]-[4]. Along with increasingly heavy interaction between human and robots, many real-time obstacle avoidance algorithms are designed to avoid static and dynamic obstacles in open space or in narrow passages, which is difficult for obstacle avoidance.

Artificial potential field (APF) approach is one of the most well-known reactive on-line obstacle avoidance methods applicable in known or unknown environments [5]. The goal position of the robot is assigned as an artificial

attractive potential and obstacles are applied as artificial repulsive forces. Then, the collision avoidance path is derived by using the gradient of linear superposition of each potential. However, by following the gradient path of APF, navigation routes generated by APFs suffer from the local minima, i.e. positions where the gradient of APF vanishes so that the robot gets stuck there, thus preventing the robot from reaching the target or lower the navigation efficiency. In general, it is desired that no other local minima except the goal exists in APF so that the navigation could reach the goal. For this purpose, hydrodynamic or harmonic potential functions (HPFs) [7]-[9], [11], [17] are special types of APFs derived from the velocity potential of solution to boundary value problem of Laplace equation with appropriate boundary conditions in a computational domain. Properties of HPF such as min-max principle and superposition were proved in [7]. To repel from the obstacles, two types of boundary conditions are imposed on the solution to Laplace equation in a computational domain: Dirichlet type and Neumann type. HPF has the property of min-max principle, so that no local minima other than the goal in the interior of cluttered or bounded environments with Dirichlet boundary conditions (the potential on the obstacle is assigned a constant high value, i.e. the motion on the obstacle boundary is along the normal direction of the obstacle boundary/wall) or Neumann boundary conditions (i.e. the motion on the obstacle boundary/wall is parallel to the tangential direction of the obstacle boundary since the normal component is null , or the flow can not pass through the boundary). The gradient of HPF, or streamline, defining the vector field of the path at every point can be computed efficiently analytically for simple obstacle shape such as circle or numerically. Laplace equation in a computational domain could be discretized using finite difference method, resulting in a system of linear equations. Numerical methods such as the Jacobi iteration, Gauss-Seidel iteration and SOR (successive over relaxation) iteration methods in grid environment, or finite element [25] can be employed to solve the linear equations. The gradient of the obtained potential values gives the streamline, or the direction of velocity at each grid [7], offering an explicit specification of heading of smooth, natural- looking path for navigation. A log-space algorithm with GPU acceleration was proposed to fix the numerical precision problem of numerical solution of discretized Laplace equation (a linear system of linear equations) via the finite difference methods [10] at the grid points that has nearly vanishing gradients. For path planning on an unstructured terrain consisting of meshes of different size and geometry, [25] proposed to use a graph search to generate an initial path from start to goal, then used streamlines to smooth the initial path. Wang et al. [16] introduced a reactivity parameter to adjust the amplitude of the path's deflection around an obstacle and an optimal 3D path is obtained by genetic algorithm.

Nonholonomic motion planning in multiple obstacles environment is challenging since the motion planners have to plan the heading and the speed by dealing simultaneously with collision avoidance and nonholonomic constraint [17], in which the curvature constraint significantly restricts severely the allowable paths to be followed by a nonholonomic mobile robot. HPF-based kinodynamic motion planning [22], [26] in particular curvature-constrained nonholonomic motion planning, takes into account the robot's dynamic capabilities and allows mobile robots to navigate safely and comfortably even in high speed along a streamline-based trajectory compatible with the kinodynamic constraints of the motion. [26] used the gradient of HPF as an additional input to guide the motion of a two-wheeled drive mobile robot based on the controller design using an invariant manifold to avoid the obstacle, thus extending the guidance method of [22] based on HPF. With the advantage of smooth trajectories generated by HPF approaches, Lau et al. [18] provides a streamline-based kinodynamic motion planning approach to avoid elliptical obstacles, guaranteeing both velocity and curvature are within limits by adjusting the strength of a source and a sink if a portion of trajectory violates kinematic constraints.

Motion planning based on hydrodynamics applies different fundamental elements such as a point sink (representing the goal), a point source

(representing the robot location), or a uniform flow (defined as a flow with constant speed) plus a doublet (representing the obstacle), and their superposition, to create a new HPF. An advantage is streamlines could be computed off-line based on prior obstacle information (distribution, i.e. shape, size, location and number) for simple shape such as circle in this paper. The path tangent or vector field at each point corresponds to the streamline of the flow with velocity defined by weighted superposition of velocity which is induced by an individual obstacle. Notably, HPF-based path planner is a complete [7] and anytime algorithm [10], in which the streamlines generated cover the free regions of the workspace. There are a few simulations showing that a robot modeled as a point (fluid) particle could smoothly navigate without collision with the (circular) obstacles [6]-[9], [11]-[15], [27], [30] by following streamlines from a variety of start points. Along this line of work, this paper elaborates on demonstrating the potential of hydrodynamics-based motion planning approach for constant speed circular nonholonomic robots to navigate smoothly in real-world partially unknown environments cluttered with cylinder-shaped obstacles. An obstacle avoidance system using three primitive streamline-based paths and a path selection strategy is proposed and implemented on Dr. Robot X80 robot, which equipped with sonar and infrared sensors to detect the obstacles during motion. In practice, localization error requires the motion planner to on-line update primitive path according to the lateral distance-based path selection strategy. This strategy makes the avoidance of small obstacle easier. For avoiding multiple obstacles, pure pursuit algorithm [15],[16],[17] is implemented in this paper for the purpose of enabling a smooth transition without violating the curvature constraint from the current robot position in an initially planned streamline to a selected goal in a new streamline between obstacles in real time.

The paper is organized as follows. Section II gives a brief introduction of a mobile robot with two independently driven wheels for the experiment. Section III mentions the harmonic potential field approach for avoiding cylindrical obstacles. Then, we propose three primitive paths based on streamlines and a distance-based path selection strategy of a primitive path for nonholonomic mobile robots. In Section IV, we propose a new real-time obstacle avoidance framework using primitive paths and pure pursuit algorithm. Comparisons and proof of concept experimental result in a simple environment are presented in Sec. V. Sec. VI ends with conclusion of the paper.

2. The Mobile Robot

2.1. Wheeled mobile robot system

We implemented our real-time hydrodynamics-based obstacle avoidance algorithm on Dr. robot X80, a wireless two-wheeled drive mobile robot platform. Fig. 1 shows the configuration and the front view of the mobile robot. The X80 mobile robot is an integrated electronic and software robotic system. It can be designed through a set of ActiveX control components (SDK) developed for C/C++. DUR5200 Ultrasonic Sensor and GP2Y0A21YK Sharp Infrared Sensor are equipped on the mobile robot. The robot platform is further modified to equip with Laser scanner, Kinect and laptop. The navigation algorithm runs directly on the remote PC through wireless communication.

2.2. Kinematic model

The mobile robot is controlled by low level velocity control of two wheels driven by DC motors independently. Fig. 2 illustrates the kinematic model, where (x, y) denotes the coordinates, and θ denotes the orienation of the central point of the differential-drive circular mobile robot with radius r_{Robot}. The kinematics describing the rolling without slipping of wheels can be expressed as the nonholonomic unicycle (1).

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} r/2 & r/2 \\ -r/d & r/d \end{bmatrix} \begin{bmatrix} w_L \\ w_R \end{bmatrix}, \quad \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \quad (1)$$

where v and ω denote velocity and angular velocity, respectively; w_L and w_R denote left and right wheel velocity, respectively; r is wheel radius; d is distance between two wheels. The detailed parameters are listed in Table Ⅰ.

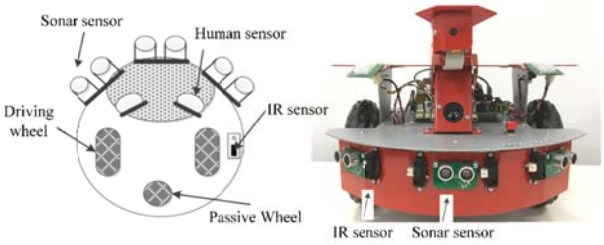

Fig. 1 Outline of the mobile robot Dr. Robot X80

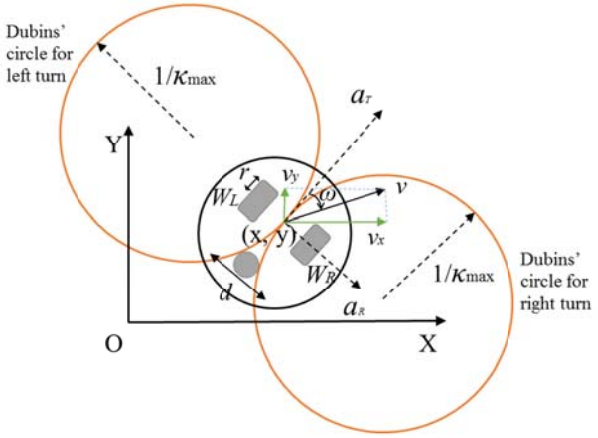

Fig. 2 The mobile robot's kinematic model with curvature constraint, where a positive curvature denotes a right (clockwise) turn. The robot is assumed as a circle. The robot velocity v and its x-component v_x, y-component v_y, the radial acceleration a_R and tangential acceleration a_T are shown.The O-XY coordinate system is the global coordinate frame, and a local frame is attached to a representative point of the mobile robot.

TABLE Ⅰ Parameters of the mobile robot Dr. Robot X80

Wheel's radius r	12.5 (cm)
Distance between two wheels d	25 (cm)
Robot radius r_{Robot}	38 (cm)
Height	25.5(cm)
Weight	3.5 (kg)
Max. Angular velocity	0.75 (rad/s)
Operating / Max.speed	0.5(m/sec) / 1(m/sec)
Cycle time	200 (ms)
Safety distance r_{Safe}	0.1 (m)
Curvature constraint $\kappa_{\max}$	1.5 (1/m)

2.3. Obstacle detector

The sensor configuration is shown in Fig. 3.DUR5200 Ultrasonic Sensor and GP2Y0A21YK Sharp Infrared Sensor are equipped on the mobile robot. There are three sonars (Sonar 1, Sonar 2, and Sonar 3) and four infrared sensors (IR 1, IR 2, IR 3, and IR 4) on the mobile robot, where θ_1 equals to 12°, θ_2 equals to 18° and θ_3 equals to 15°.The detecting range of an ultrasonic

sensor is from 4 to 255cm, while the detecting distance range of IR sensor is between 10 and 80 cm. The sensors' update rate are both 10Hz.We assume the obstacle is located in the direction of the sensor if only a single sensor detects an obstacle. Otherwise, when two sensors detect an obstacle at the same time, we assume that the obstacle lies on the bisector of these two sensors' directions. Fig. 4 shows the detection of obstacle. For instance, if Sonar 2 detects an obstacle, the obstacle is located in front of the robot with azimuth angle 0°. Likewise, Sonar 1, Sonar 3, IR 1, IR 2, IR 3 and IR 4 detect the obstacle with azimuth of 45°, –45°, –30°, –12°, 12°, and 30° respectively. If both Sonar 2 and IR 3 detect an obstalce, then the obstacle's direction will be 6°, which is between Sonar 2 and IR 3. The estimated obstacle location is transformed into the global frame for the motion planner. The estimation error is accomodated by a safety distance r_{Safe} in practical implementation of navigation system.

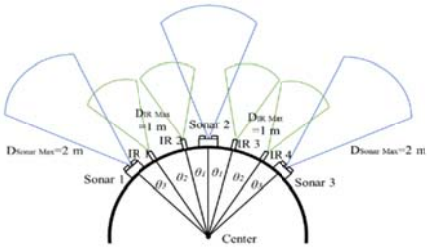

Fig. 3 Configuration of sonar and infrared sensors and their sensing ranges of Dr. Robot X80.

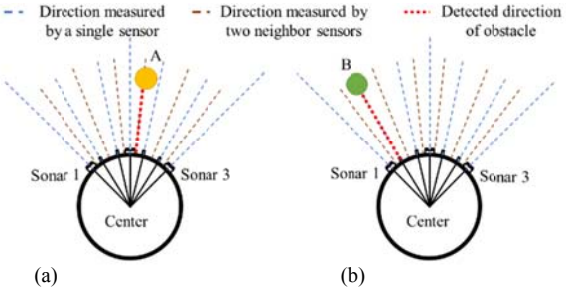

Fig. 4. Scenarios of obstacle detection. The ray of each sensoris presented in dashed line. As the rays intersect with an obstacle, thenearest intersection pointis retained. (a) An obstacle is in front of the robot. Sonar 2 detects an obstacle with distance less than 150 cm, the robot knows the obstacle is located in front of the robot with azimuth angle 0°. (b) Both Sonar 2 and IR 3 detect the obstacle, and the obstacle's direction will be 6 °, which is between Sonar 2 and IR 3. The distance of the obstacle is the average of the data received from two sensors.

3. Obstacle Avoidance Model by Harmonic Potential Field

In this section, we present our hydrodynamic path planning framework. The path to avoid an obstacle is similar to the streamline of the fluid flow around a cylinder [8]. Thus, the path for robots to follow is generated by the gradient of an artificial potential field. In potential flow theory, an uniform flowaround a cylindrical obstacle can be modeled by superpositionof uniform flow and doublet [18]. We briefly summarize the C^2 smooth path produced by streamlines of harmonic potential field and then introduce three primitive paths and pure pursuit algorithm for real-time obstacle avoidance. We assume the obstacles are far apart.

3.1. Mathematical model of the artificial potential field

Harmonic potential functions are solutions to Laplace's equation, so functions generated by Laplace's equation do not exhibit local minima [7]. In two-dimensional computational domain of Euclidean space, the velocity potential ϕ is a solution of Laplace's equation

$$\nabla^2 \phi = 0 \quad \text{with given boundary conditions}$$

that governs the flow of the non-viscous, incompressible, irrotational fluid particle motion at every point of the domain. A streamline indicates local flow direction: its tangent at every point (vector field) is in the direction of local fluid velocity associated with the flow defined in (2).

$$\mathbf{u} = \nabla\phi \tag{2}$$

We assume that the robot and the obstacle are modeled as a circle defined by its radius r_{Robot} , $r_{Obstacle}$, respectively. The distance between the robot's center and the obstacle's center is $D = D_{Sensor} + r_{Obstacle} + r_{Robot}$, where D_{Sensor} is the range value measured by the sensor, $r_{Obstacle}$ is the radius of obstacle, and r_{Robot} is the radius of the robot.Given a safety distance r_{Safe} between a robot and an obstacle, the radius of obstacle is enlarged by taking account of the robot radius as $r_{Obs} = r_{Obstacle} + r_{Robot} + r_{Safe}$. In the case of a circular robot and a circular obstacle, the collision free criterion for safe navigation is that the distance between the point robot (robot center) and the obstacle center is larger than r_{Obs} .

Consider a mobile robot at $\mathbf{x} = [x \ y]^T$ moves in the +x-axis direction with a forward/ longitudinal speed U to avoid a circular obstacle of radius $r_{Obstacle}$ located at origin, the velocity potential field $\phi(x, y)$ can be represented as the superposition of an uniform rectilinear flow and a doublet [9], [14], [19]as

$$\phi(x, y) = Ux + \frac{A}{\mathrm{x}^2 + \mathrm{y}^2} x \tag{3}$$

where constant $A = Ur^2$. According to (2), the robot's velocity in Cartesian coordinates is

$$u = \frac{\delta\phi}{\delta x} = u + \frac{2y^2 A}{(\mathrm{x}^2 + \mathrm{y}^2)^2} - \frac{A}{\mathrm{x}^2 + \mathrm{y}^2}, \quad v = \frac{\delta\phi}{\delta y} = -\frac{2Axy}{(\mathrm{x}^2 + \mathrm{y}^2)^2} \tag{4}$$

In practice (and in our experiments in Sec. V), we assume a constant forward speed U, so the velocity in (4) is normalized to unity while its direction of motion is preserved. Then, normalized velocity and acceleration of each point on the streamline in the uniform flow are given by (5), (6)

$$u_N = \frac{u}{\sqrt{u^2 + v^2}} U, \quad v_N = \frac{v}{\sqrt{u^2 + v^2}} U \tag{5}$$

$$a_x = \frac{\delta u_N}{\delta x} u_N + \frac{\delta u_N}{\delta y} v_N, \quad a_y = \frac{\delta v_N}{\delta x} u_N + \frac{\delta v_N}{\delta y} v_N \tag{6}$$

Furthermore, curvature and deviation of curvature can be derived by velocity and acceleration as

$$\begin{aligned} \kappa &= \frac{u_N a_y - v_N a_x}{(u_N{}^2 + v_N{}^2)^{3/2}} \\ &= -2Ay\sqrt{A^2 + 2AU\left(-x^2 + y^2\right) + U^2\left(x^2 + y^2\right)^2} \\ &\times \frac{\left[A^2 + 2AU\left(x^2 + y^2\right) + U^2\left(-3x^4 - 2x^2y^2 + y^4\right)\right]}{\left(x^2 + y^2\right)\left[A^2 + 2AU\left(-x^2 + y^2\right) + U^2\left(x^2 + y^2\right)^2\right]^2} \end{aligned} \tag{7}$$

$$\dot{\kappa} = \frac{\delta\kappa}{\delta x}\frac{dx}{dt} + \frac{\delta\kappa}{\delta y}\frac{dy}{dt} = \frac{\delta\kappa}{\delta x} u_N + \frac{\delta\kappa}{\delta y} v_N$$

$$= \frac{24AUxy\left[\begin{array}{l} A^4\left(x^2 - y^2\right)^4 - U^4\left(x^2 - y^2\right)\left(x^2 + y^2\right) \\ -2A^3U\left(x^2 + y^2\right)^2 + 2AU^3\left(x^2 + y^2\right)^4 \end{array}\right]}{\left(x^2 + y^2\right)^2\left[A^2 + 2AU\left(-x^2 + y^2\right) + U^2\left(x^2 + y^2\right)^2\right]^3}$$

From (5), given a start position, a streamline can be derived by numerical integration of the velocity vector field that specifies the tangent of the path or the robot heading at each point. In addition, the curvature of a streamline could be determined by the initial position of the streamline. That is, the velocity vector field serves as a vector field for guidance of robot motion. Therefore, in the path planning applications, the streamlines provide a pool of systematic paths that have explicit or analytic vector field for the tangent to the path as the path specification. However, there are several drawbacks for robots to purely follow streamline paths. First of all, if a robot's initial moving direction is close to the center of obstacles, there will be larger curvature of streamline with the distance getting closer. For example, in Fig. 5(b), the initial position of robot at (-5, 0.2) have the maximum curvature. Second, the paths with smaller curvature are longer and keep unnecessary distance with obstacle. Moreover, for paths which initial position is further than radius of obstacle from x-axis, it's not necessary to follow streamline path because a robot can pass an obstacle straightly, such the paths with start position further than the radius of obstacle with the x-axis in Fig. 5(a). Therefore, we provide improved approach in the following section.

In general , assume that a robot locates at $\mathbf{x}_{robot}$ with initial heading angle θ with respect to +x-axis, and an obstacle is at $\mathbf{x}_{obs}$. Robot position $\mathbf{x}_{robot}$ is related to $\mathbf{x}$ via the transformation matrix

$$\mathbf{x}_{robot} = R(\theta)\cdot\mathbf{x} + \mathbf{x}_{obs}$$
$$R(\theta) = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \qquad (8)$$

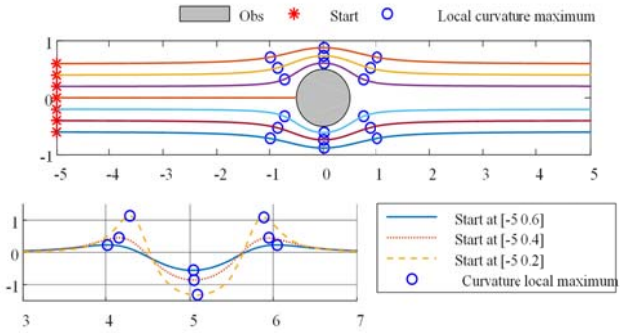

Fig. 5. Streamline paths with different start positions and their curvatures for a uniform flow around a circular obstacle.

3.2. Three primitive paths with curvature constraint

We first identify curvature bounded paths for robots to avoid a cylinder obstacle, and multi-obstacle situation will be discussed in the following section. There are two strategies for local obstacle avoidance. First, the robots could pass an obstacle with minimum curvature changes. Second, it could also pass an obstacle with maximum allowed curvature for sharp turn. Different timing for these two strategies depends on the upcoming obstacle's position after pass first obstacle. In Fig. 6 (a), after passing the first obstacle, the second obstacle is coming immediately. Therefore, the robot has to turn sharply to prevent collision. On the other hand, pass the first obstacle with minimum curvature allowed the robot to pass these two obstacle in the same side while there is no enough space to pass through between two obstacles with curvature constraints in Fig. 6(b). The third way is that robots can also pass the obstacle from the farther side by sharp turn. The following are the details and procedures of sharp turn and minimum curvature turn by streamline path.

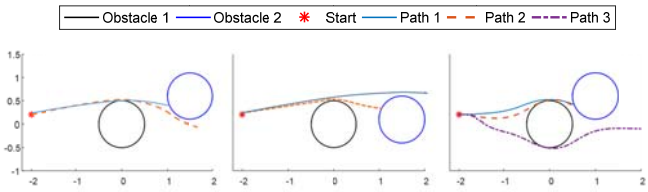

Fig. 6. Paths of sharp turn and minimum-curvature turn for local obstacle avoidance. Path 1: minimum- curvature turn. Path 2, Path3: sharp turn.

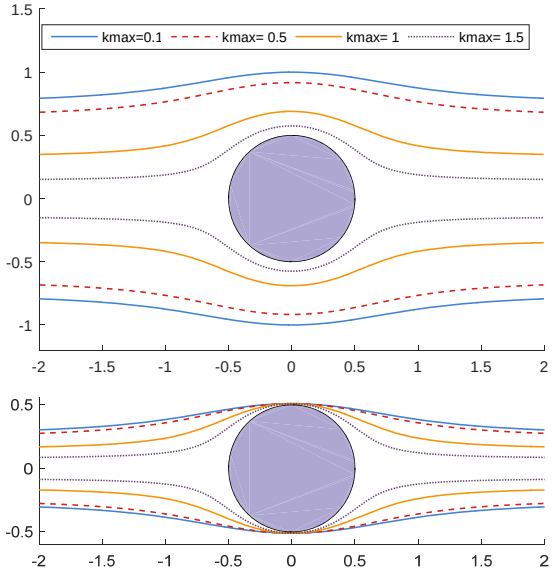

Fig. 7 Concept of sharp turn streamline paths to circumvent a circular obstacle of radius 0.5 m with the center placed at the origin in x-y plane.. (a) Hydrodynamic streamline paths with different curvature constraints. (b)Feasible paths with different curvature constraints. The curvature of streamline is larger as it is closer to the circle.

For local obstacle avoidance, three collision-free curvature-constrained primitive paths that most aligned to the current robot's heading are proposed based on the richness of streamlines. First, a point robot could circumvent an obstacle with maximum curvature from its left or its right side. Second, it could also pass an obstacle with maximum curvature viaasharp turn. An initial streamline is chosen based on initial robot configuration and a priori known obstacle distribution, taking into consideration of curvature constraint. This initial path may collide with obstacles. To ensure safe navigation, one of our obstacle avoidance strategy is to make the robot change from one streamline to another streamline at a lookahead distance via a local, on-line pure pursuit algorithm without violating curvature constraint. Details are as follows.For simplicity of illustration, it is assumed assume that the robot is moving in the +x-direction and a circular obstacle of radius r_{obs} is located at origin, so that the maximum curvature of streamline occurs at the y-axis.

(A) Sharp left or right turns

According to the curvature profile of the potential flow field in Fig. 7, we identify that local maximum of paths' curvature is located at the y-axis while the robot is moving in the x-direction. Furthermore, the maximum curvature of a path is smaller when the path is further from the obstacle. In order to find the curvature maximum allowablestreamline path, the curvature maximum points lying on the y axis is identified first by Procedure 1. In Procedure1, the point *pLow* with curvature smaller than maximum is identified by increasing distance from (0, *a*). Then, binary search is used along the y-axis between (0, *pLow+a*) and *(0, pLow)*, to locate the point with curvature κ which equals to the maximum allowed curvature κ_{max} .

$$|\kappa| \le \kappa_{\max} \qquad (9)$$

Then, generate the path by velocity in (5) by numerical integration for the discrete-time system forwardly and backwardly. According to the curvature profile in Fig. 7 (b), there are extra two points with local maximum curvature in a single path, so the global maximum curvature of a path has to be checked whether curvature of a pathis within constraint after the path is generated. Fig. 7(a) presents hydrodynamic streamline path with different curvature constraints.

One characteristic of streamline path is the clearance with an obstacle. Besides, for paths which initial position is further than radius of obstacle from x-axis, it's no need to deform the path because it can avoid obstacle straightly. In order to get closer to the obstacle, we translate the path by shifting the curvature maximum point to the (0, a). For a 2D circular obstacle, it's able to pass it by opposite sides symmetrically. Therefore, we define two possible obstacle avoidance paths based on the obstacle's radius and robot's maximum allowed curvature, and modified paths are shown in Fig. 7 (b). Due to the fact that these two paths are not connected with robot's initial position, we will use path pursuit strategies to pursuit these two curvature maximum paths.

Consider the avoidance of the nearest obstacle within the sensing range in front of the robot. Two curvature-constrained streamline-based turn paths symmetrical with respect to the line connecting robot center and obstacle center could be used as two primitive paths to ensure safe navigation from left or right side of the obstacle based on the obstacle's radius. We search the two streamlines corresponding to left turn and right turn with curvature maximum equal to maximum curvature $\kappa_{\max}$. The desired streamline is obtained by shifting the selected streamline parallelly until its curvature maximum point grazes the obstacle boundary.

Procedure 1:

Input:Maximum allowed curvature $\kappa_{\max}$, radius of the obstacle a
Output: Maximum curvature in Y-axis y

//First find the point, *yLow*, which curvature lower than
// *Kmax* by searching with a distance a along +Y-axis from
// (0, a), and then *yHigh*is decided by $yLow + a$ yLow-a.
$yLow = 2a$;
While $\kappa(0, yLow) > \kappa_{\max}$ // κ<- curvature(x, y) from (7)
  $yLow = yLow + a$
endwhile
  $yHigh = yLow - a$
// Curvature maximum point is found by binary search
// between (0, *yHigh*) and (0, *yLow*)
While $curvHigh - curvLow < \varepsilon$ // ε :an tolerate error
  $y = (yLow + yHigh) / 2$
  If $|\kappa(0, y)| > \kappa_{\max}$ **then** $yLow = y$
  Else $yHigh = y$ **end if**
end while
return y

(B) Minimum curvature turn

We exploit the property that the deflection and curvature of a streamline become smaller as it is farther from the obstacle. First, search the streamlines whose maximum curvature is not larger than $\kappa_{\max}$ farther from the obstacle than the current streamline the robot stays to find the first streamline when shifted to the robot current location is collision-free. This streamline is used as the minimum curvature turn path.

We find a path starting from current robot position that can pass an obstacle with minimum curvature. If we purely follow streamline path, there will be an unnecessary distance with obstacles. Moreover, we realize that when the initial moving direction far from the center of obstacle, it results in smaller curvature and deviation while passing the obstacle. Fig. 8 shows the concept of minimum curvature path. Instead of following the streamline path, we find the minimum- curvature path by using the streamlines farther from the obstacle. Firstly, we search streamline paths further from obstacle than the streamline the robot lies on. Then, move the streamline along with the lateral direction of the robot back to the robot's start position iteratively until the path collides with the obstacle. After we find the minimum curvature collision free path, pull the path back to the robot initial position and searching is finished.

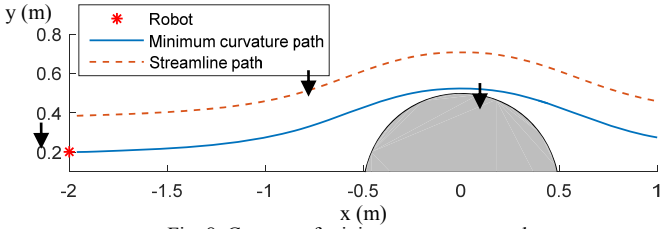

Fig. 8 Concept of minimum curvature path

Given an obstacle's radius and robot's maximum allowed curvature, we can derive three primitive paths which satisfy curvature constraints as shown in Fig. 9. Fig. 9 demonstrates primitive paths for various robot's initial position (or relative distance to the obstacle) and curvature constraints. For all three primitive paths, a robot needs to keep enough distance with the obstacle to achieve pursuit primitive paths with stricter curvature constraint, i.e. lower maximum curvature. In addition, while maximum curvature constraint deceases, it's more difficult to achieve primitive paths. Excessively restrictive curvature constraint causes no feasible solution path is found. Moreover, the lateral distance to the obstacle will influence the ability to find a feasible path.

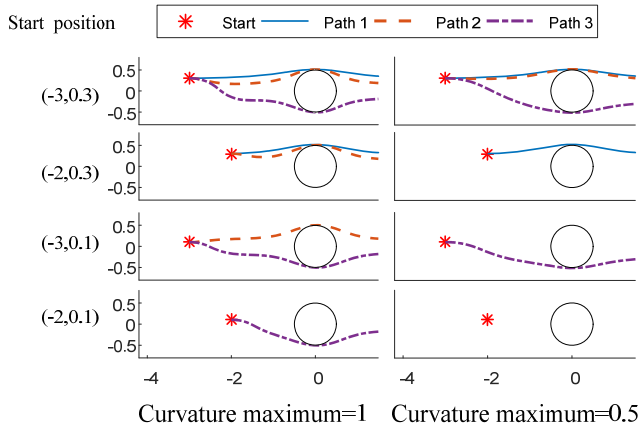

Fig. 9 Single obstacle avoidance path for different start position and curvature maximum. Radius of the obstacle is 0.5 meter. Robot forward moving direction is positive x-axis (toward right).

3.3. Distance-based obstacle-avoiding path selecting strategy

The path selecting strategy depends on the reaction distance to upcoming obstacle position or lateral displacement relative to the size of obstacle. The strategy is illustrated in the scenario of Fig. 10.The robot is initially located at x=-2 with different lateral distance y related to a cylindrical obstacle at the origin. Let $b+$ and $b-$be the points which two sharp turn paths intersect with the line $x=-2$.The interval [-r, r], denoted by [$d-$, $d+$] in Fig. 10 at the vertical line $x=-2$is partitioned into intervals B$+$ ~ B$-$ by the labeled points

$d-$ to $d+$ according to the start points of the primitive paths, symmetrically with respect to the robot current position. We define L_{sharp} as the distance between points $c+$ (the start point with right sharp turn path) and $c-$ (the start point with minimum curvature path).

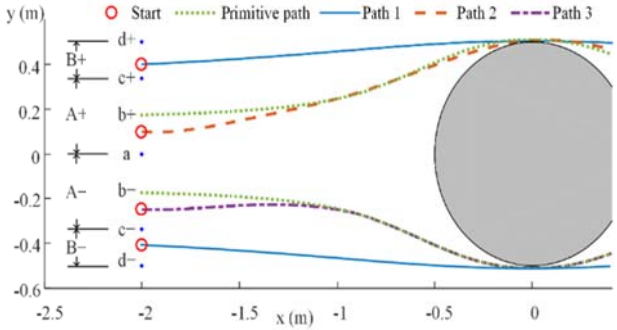

Fig. 10 Illustration of primitive path selecting strategy. The range $[-r_{obs}, r_{obs}]$ according to the obstacle size is partitioned manually into three intervals from far to near to reflect the reaction distance.Robot with different lateral displacements related to an obstacle will pursue different primitive streamline paths aligned to current robot heading. Path 1 and Path2 are tangentially traverse the enlarged circular obstacle boundary.

Specifically, given a current robot position and an obstacle of known size and position, the obstacle-avoiding path selecting strategy is proposed. The strategy is according to the relative lateral distance $d_{lat} \leq r_{obs}$ measured from the center of obstacle (x_{obs}, y_{obs}) in front of the robot, where current robot location is at a fixed longitudinal distance. The range $[-r_{obs}, r_{obs}]$ according to the obstacle size is partitioned manually into three intervals from far to near to reflect the reaction distance as Fig. 5shows.as the following rules

$$\begin{cases} d_{lat} \in A+ \text{ A}- \rightarrow \text{Path 1} \\ d_{lat} \in B+ \text{ B}- \rightarrow \text{Path 2} \\ d_{lat} \in \text{special cases} \rightarrow \text{Path 3} \end{cases} \quad (8)$$

Once an obstacle is sensed by the obstacle detector, the motion planner determines the proper primitive path for the mobile robot to follow to circumvent the obstacle. This strategy is designed to use a minimum curvature turn in Intervals $A+$ and $A-$, while it pursues sharp left turn in Interval $B+$ and a sharp right turn in Interval $B-$ as the obstacle is closer. In addition, this strategy makes the avoidance of a small obstacle easier. Note that for obstacles with the same radius, the paths generated by streamlines of Laplace equation according to the same position are identical. Hence, streamline-based paths can be computed in advance for circular obstacles with different radii and stored and maintained in the dataset. The path obtained by transforming a sample path computed for an obstacle located at origin to the estimated or true obstacle position could then be reused to on-line plan the collision-free movement with reduced time-complexity.

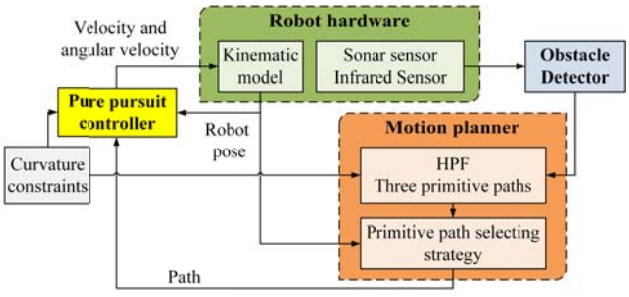

Fig. 11 Flowchart of on-line obstacle avoidance system for a curvature constrained nonholonomic mobile robot based on primitive streamline paths, a path selection strategy and a pure pursuit algorithm for streamline changing .

4. Real-Time Streamline-based Obstacle Avoidance Strategy

4.1 Overview of the obstacle avoidance system

Fig. 11 depicts the building blocks of the obstacle avoidance system. The real-time obstacle avoidance system is built by three subsystems, which are the obstacle detector, the motion planner that incorporates curvature constraint, and the pure pursuit controller used to control the robot to follow the specific primitive path with allowable angular velocity satisfying the curvature constraint. For the part of robot hardware, we used robot's kinematic model to estimate robot's own kinematics and sensors to detect surrounding environments. The obstacle's global location is estimated by the range data receiving from sonar and infrared sensors. Our motion planner initially selects a streamline starting from the robot's start position, which is generated based on a priori known obstacle distribution. The obstacle's location is updated based on new sensor data during robot's forward motion, and the motion planner will decide whether to enable local re-planning based on a path selection strategy or to retain original path for the robot to follow. Local re-planning is performed by generating and updating a local subgoal that is on a new primitive collision-free path and smooth transition between streamlines is enabled via pure pursuit.

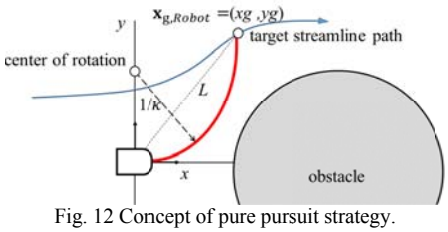

Fig. 12 Concept of pure pursuit strategy.

4.2 Pure pursuit controller for mobile robots

Pure pursuit is a path tracking method by calculating the curvature of a new circular path for vehicle to pursuit a goal position ahead of the vehicle by leaving the initially planned path from its current position [15]. Due to the fact that nonholonomic robots cannot directly move in the lateral direction, a robot pursues a subgoal position ahead of the robot to redirect from its current position along an arc of curvature κ via pure pursuit [20], [21]. Since the obstacle avoidance path can be computed analytically according to the relative position of robot and obstacle and shape and size of the obstacle, this method has an implementation advantage. Firstly, the equations of the pure-pursuit curvature control law are derived. The curvature κ of the vehicle is defined as the inverse of the distance r_c, also called as the radius of curvature, between the vehicle's frame origin and its instantaneous center of rotation. Second, curvature also represents the instantaneous change of the vehicle heading angle $d\theta$ with respect to the traveled distance ds. Hence, curvature is formally defined as follows:

$$\kappa = \frac{1}{r_c} = \frac{d\theta}{ds} \quad (10)$$

Fig. 12 illustrates the concept of pure pursuit controller. An algorithm is shown in Algorithm 1. Once a primitive path is generated, the next step is to find a goal point $\mathbf{x}_g$ which is located after the closest point. Let a local coordinate system be attached to the robot with its origin set as rotation center of the robot and +x-axis of the local frame aligned with the forward motion direction. Then transform a goal point $\mathbf{X}_g$ in the global coordinate to $\mathbf{x}_{g,Robot}$= $[x_{g,Robot} y_{g,Robot}]^T$ in local frame with $x_{g,Robot}$ and $y_{g,Robot}$ denoting the longitudinal and the lateral displacements, respectively,

$$\mathbf{x}_{g,Robot}=R(\theta)(\mathbf{X}_g-\mathbf{X}_{Robot}) \quad (9)$$

where $\mathbf{X}_{Robot}$ is current robot position in global frame, and θ is the heading angle of robot in global frame.

The goal point $\mathbf{x}_{g,robot} = (x_g, y_g)$ in vehicle coordinates can be represented with κ and α by geometry:

$$x_g = r_c\left(\cos(\alpha)-1\right) = \frac{\cos(\alpha)-1}{\kappa}$$
$$y_g = r_c \sin(\alpha) = \frac{\sin(\alpha)}{\kappa}$$

, where α is the angle of the arc between the vehicle and the goal.

The subgoal point to pursuit keeps a specific lookahead distance $L = \sqrt{x_{g,Robot}{}^2 + y_{g,Robot}{}^2}$, since nonholonomic robots cannot correct errors directly with respect to the nearest point on the path. The curvature κ of the robot is defined as the inverse of radius of curvature r_c, i.e. the distance between the origin of the robot frame and its instantaneous center of rotation. In addition, curvature can also be defined as the instantaneous change of the heading angle $\Delta\theta$ with respect to the travel distance $U{\cdot}\Delta t$ in sampling time Δt. Curvature then could be further related with robot's velocity and angular velocity. Therefore, the angular velocity of a robot moves along a path at a constant speed U could be computed from the path curvature via the relation

$$\kappa = \frac{1}{r_c} = \frac{\omega}{U} = \frac{\Delta\theta}{U{\cdot}\Delta t}, \quad (10)$$

where $\kappa = y_{g,Robot} / L^2$ [20]. To satisfy the maximum allowable curvature, we regularize the signed curvature as

$$\kappa_{constraints} = Sign(\kappa)\cdot\kappa_{\max} \quad if \quad \kappa > \kappa_{\max} \quad (11)$$

Thus, the motion in the local frame of the robot can be applied to command motion controller via the inverse kinematics of (1). The displacement $\Delta\mathbf{X}_g$ in the global frame can be derived from the displacement $\Delta\mathbf{x}_{Robot}$ in the local frame

$$\Delta\mathbf{x}_{Robot} = \begin{bmatrix}-\sin(\Delta\theta)\\ \cos(\Delta\theta)\end{bmatrix} U{\cdot}\Delta t, \qquad \Delta\mathbf{x}_g = \mathrm{R}(\theta+\Delta\theta)\,\Delta\mathbf{x}_{Robot} \quad (12)$$

4.3 Modification of lookahead distance based on the distance between robot and closet point on the path

Lookahead distance is the only parameter in pure pursuit algorithm, and the reason for the lookahead distance is that nonholonomic robots cannot correct errors directly with respect to the nearest point on the path [19]. The pure pursuit procedure is summarized in Algorithm 1. Fig. illustrates the pure pursuit path is sensitive to the setting of lookahead distance. For the case presented in Fig. 13, the path with lookahead 0.5m has effective and efficient tracking ability. However, the path with too large lookahead distance, 1m, is smoother but unable to track path accuracy. On the contrary, the path with a shorter lookahead 0.1m responds to tracking errors quickly, but the robot's motion is unstable and overdamping because of the curvature constraint. Both paths obtained with lookahead 1 m and 0.1 m lead to collide with the obstacle. In practice, we create a set of lookahead distances that range from 0.1 to 2 with equal interval distribution according to radius of obstacle. After calculating the pure pursuit path for each lookahead distance value, the shortest path is selected. The common practice for setting lookahead distance takes into consideration its effect on path geometry and tracking performance. A longer lookahead distance results in smoother paths but worse tracking accuracy. In contrast, a shorter lookahead can reduce tracking errors more quickly. Yet,

due to curvature constraint, the pure pursuit controller may not be able to follow steering commands, and the robot motion becomes unstable. Therefore, a suitable lookahead is needed for both stability and tracking performance.

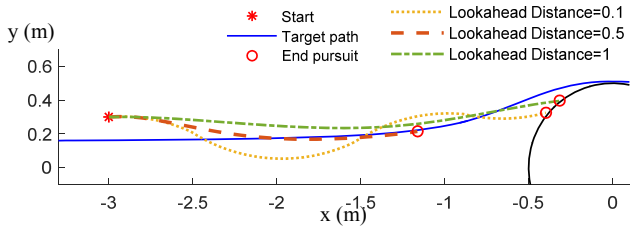

Fig.13.Replanning via pure pursuit paths starting from a fixed position with a set of subgoals determined by different lookahead distances. The pursuit paths with lookahead distance 0.1 and 1intersect with the obstacle, while the pure pursuit path with lookahead distance 0.5 is collision-free.

Algorithm 1: Pure pursuit streamline path

Input:robot initial pose, target streamline path, lookahead distance, maximum allowable curvature
Output: status, pursuit path

While (**not** *timeout***or***status*) **do**
  let R(θ)represent the transformation to robot coordinate
  pClosest← {(x, y) | min{|(x, y) – poseRobot|} **and** (x, y) in pursuit path}
  $\mathbf{X}_g$← {(x, y) | min{|(x, y) – pClosest|} **and** (x, y) in pursuit path after pClosest}
  $\mathbf{x}_{g,Robot}$←R(θ)($\mathbf{X}_g$-$\mathbf{X}_{Robot}$) (9)
  Calculate the curvature $\kappa \leftarrow y_{g,Robot} / L^2$
  Regularize the curvature constraints(11)
  Set the steering angle of the robot (12)
  Update robot's heading direction, poseRobot
    If poseRobot and curvature == streamline path **do**
      status ←**TRUE**
      store path into pathArray
    if collide with obstacle **do**
      status ←**FALSE**
end while

4.4 Multiple obstacles avoidance strategies

In an environment composed by multiple obstacles, previous researchers provided several different methods to create a guidance vector field. [6]-[8] used the weighted superposition of single obstacle. Although the sum of HPFs is also HPF, hence free of local minimum, the superposition has no guarantee to satisfy the curvature constraints. Hence, we propose a new avoidance strategy for multiple obstacles.

(1)Superposition of multiple obstacles

[6]-[8] proposes a weighted velocity field which not only guarantees no local equilibria in the workspace, but also satisfies zero Neumann boundary condition on every boundary of obstacle. Following the streamline-based path planner, each obstacle is treated individually. The total influence of all obstacles in an environment with N obstacles on velocity field $\mathbf{V_{total}}$ can be expressed as the weighted sum of N velocity fields of $\mathbf{V_1}, \dots, \mathbf{V_N}$ for each obstacle

$$\mathbf{V_{total}} = \sum_{i=1}^{N} w_i \mathbf{V_i} \tag{13}$$

where w_i is the position-dependent weighting function for obstacle i. One can design $w_i = \prod_{j \neq i}^{N} \frac{d_i}{d_i + d_j}$ with d_i denoting the shortest distance between the robot and the obstacle i. This design makes the closest obstacle have the largest weight. In real-time applications, we have to calculate (130 for all of

the obstacles within the sensing range or a user-defined zone even its weighting is slight.

(2)Multiple obstacles path by pure pursuit

As the above mentioned, we provide a new strategy for robots to move smoothly in a multi-obstacles environment. According to primitive paths discussed in the last section, we can identify three primitive paths once robot's initial pose, maximum allowable curvature, obstacle's position and radius are given. Fig. summarized the discussions so far as a flowchart of hydrodynamic path planning in combination with pure pursuit in multiple obstacles situation. In multiple obstacles situation, we initialize a queue called *poseRobotArray* to store robot's initial pose. While the queue is not empty, we assign the first element of *poseRobotArray*to *poseRobot*, and pop the first element of the queue. Then, we move the robot forward to the target to check whether the path is collision-free. If no obstacles are detected on the path, we store the path into *pathArray*. Otherwise, generate three primitive path to avoid the detected obstacle. For each generated path, we check if it collides with any other obstacle. If it is collision-free, we store robot's final pose into *poseRobotArray*. On the other hand, check if the collision happened before or after passing the first obstacle. If the collision happened before the first obstacle, we generate three primitive paths to avoid new obstacle from the robot's initial pose. In contrast, we set the point on the path closest to the original obstacle as *poseRobot*and then avoid new obstacle. Finally, select the optimized path from *pathArray*.

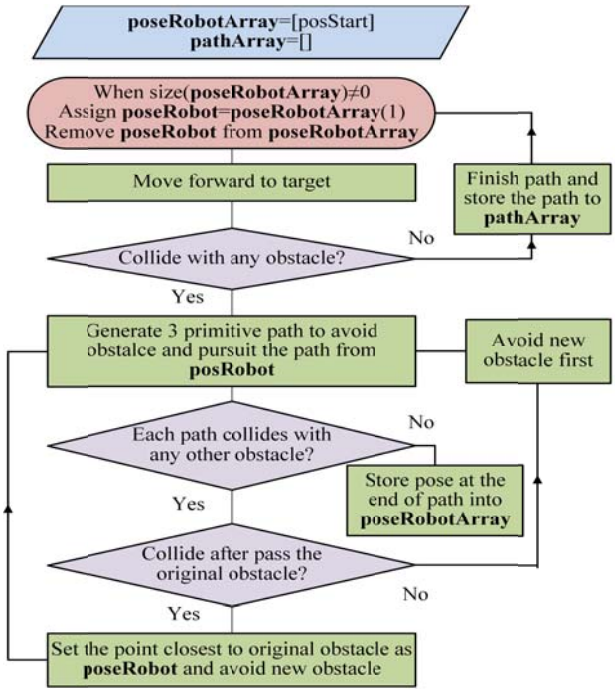

Fig. 14 Flowchart of hydrodynamic path planning by pure pursuit in multiple obstacles situation.

4.5 Safety reaction distance

The following discusses the safety reaction distance to an obstacle, given a specific curvature constraint.

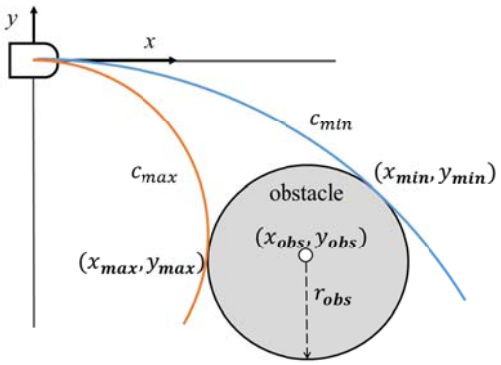

Fig.15 There are two turning directions around the obstacle. Obstacle-avoiding path from either side of an obstacle, their curvatures and tangential points

For a given obstacle to be avoided, there are two turning directions around the obstacle in the plane. In order to take into account the curvature constraint, we need to compute the range of curvature of circular arcs that touches the obstacle from different sides at any speed. As shown in Fig. 15, we can calculate the the radius of each right turning circular arc that makes tangential contact on the boundary of obstacle and their respective tangential contact point as follows

$$c_{min} = \frac{2(x_{obs} - y_{obs})}{\sqrt{x_{obs}^2 + y_{obs}^2 + r_{obs}^2}},\ c_{max} = \frac{2(x_{obs} + y_{obs})}{\sqrt{x_{obs}^2 + y_{obs}^2 + r_{obs}^2}} \qquad (14)$$

$$x_{min} = \frac{(x_{obs} - y_{obs})}{1 - (c_{min} - r_{obs})},\ x_{max} = \frac{(x_{obs} + y_{obs})}{1 + (c_{min} - r_{obs})} \qquad (15)$$

$$y_{min} = \frac{y_{obs}}{1 - (c_{min} - r_{obs})}, y_{max} = \frac{y_{obs}}{1 + (c_{min} - r_{obs})} \qquad (16)$$

where c_{min}, c_{max} are radius of minimum and maximum turning arcs that contact the obstacle tangentially and (x_{min}, y_{min}), (x_{max}, y_{max}) are their respective tangential contact points.

In [20] and [21], as shown in Fig. 16, the hitting distance function d_v was defined for a unicycle (1) with curvature defined by the ratio of angular velocity ω and linear velocity v as

$$d_v = \begin{cases} d_c\left(\frac{\omega}{v}, obs\right), \text{if } v \neq 0 \\ \infty, \text{otherwise} \end{cases} \qquad (17)$$

where $d_c(c, obs), , c \in [1/c_{max}, 1/c_{min}]$ is the arc length that the point robot would travel following a circular arc of radius $1/c$ before hitting the obstacle obs at a point (x_i, y_i).

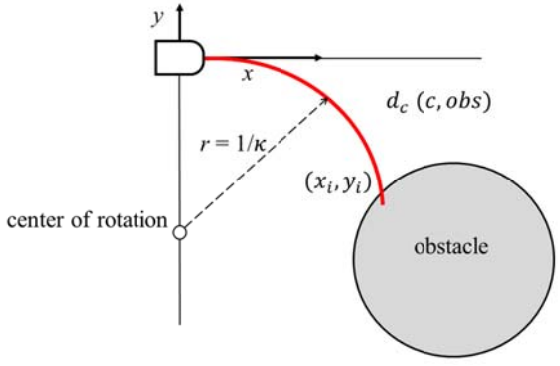

Fig.16.Hitting distance to an bstacle, where robot forward moving direction is positive x.

In our scenarios, given maximum allowed curvature ($1/Rmax$), for a unicycle moving forward along the +x direction, the safety reaction distance to avoid an obstacle is given by $Xobs$.

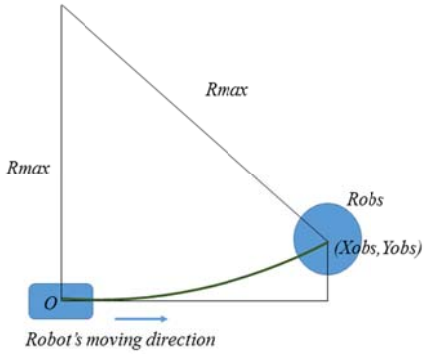

Fig.17 geometry relation of mobile robot and obstacle.

According to the geometry relation in Fig. 17, it is easily seen that

$$\left(R\max+Robs\right)^2 = Xobs^2 + \left(R\max-Yobs\right)^2 \qquad (18)$$

Thus, the safety reaction distance in the +x forward motion direction before the robot hits the obstacle is

$$Xobs = \sqrt{\left(R\max+Robs\right)^2 - \left(R\max-Yobs\right)^2} \quad (19)$$

5. Comparisons and experiment

In this section, we demonstrate the proposed algorithm for navigation within multiple circular obstacles to show the planner's performance in a cluttered environment. Comparisons of our approaches with other methods are also shown in the following. The speed of robots was set as 1 m/s for all scenarios, and the initial heading direction is aligned in positive x-axis defined as the forward direction. Two different cases are discussed.

-Pure pursuit method vs lane hopping method
-Multiple obstacles environment

5.1 Comparison of pure pursuit method and lane hopping method

In order to leave an initially planned streamline and change to another one, lane hopping (streamline changing) [8], [9] is enabled in case the robot is too close to obstacle (imminent collision) or the current streamline the robot follows violates the curvature constraint. Lane hopping requires that the x coordinate in the two streamline paths before and after hopping is almost the same. In lane-hopping method, a 2×2 filter matrix K_{filter} is used to generate lane-hopping paths. In contrast to filter matrix, pure pursuit strategy is easier in application, for only a single value, i.e. lookahead distance, is need to be tuned. Thus, it's easier to find feasible paths by pure pursuit. Furthermore, pure pursuit method is also designed to satisfy the curvature constraint for the part of streamline-changing path, in addition to smoothness. Fig. 18 presents that pursuit of target path both by pure pursuit and lane hopping. Both the filter matrix $K_{filter} = 0.1I$ with I 2×2 identity matrix in lane-hopping and lookahead distance L=0.5m in pure pursuit are selected manually so that the paths can achieve their respective goal on the selected streamline, then follow the new streamline. The maximum curvature of lane hopping (about 10 (1/m)) is remarkably larger than that of pure pursuit (1(1/m)) in the beginning of streamline changing, and the lane-hopping path can achieve the target streamline earlier.

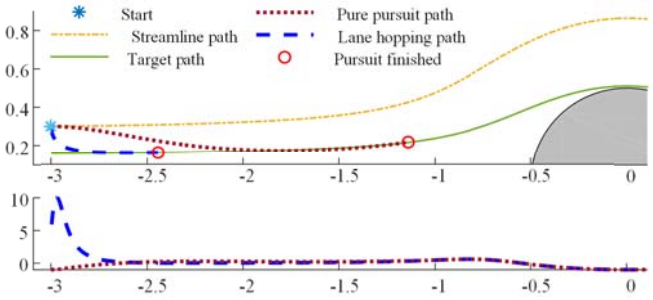

Fig.18 Comparison of pure pursuit and lane hopping: path (upper plot) and curvature (lower plot)

5.2 Multi-obstacles environment

(A) Comparison with streamline path by weighting velocity of each single obstacle

In Fig. , the proposed method is compared with the streamline path obtained via the weighting method. The maximum curvature of pure pursuit path and streamline path are 0.50 (1/m) and 1.44 (1/m), respectively. The maximum curvature of pure pursuit path is smaller than the streamline path.

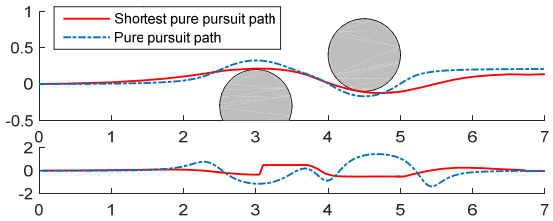

Fig. 19 Two obstacle environment. Comparison of pure pursuit path and weighting streamline path.

(B) Clutter case: comparison with Lau's approach

In this example, we compare our methods with fluid motion planner provided by Lau et al [7], [14] in Fig. 20. In the clutter case, there are two feasible paths with curvature constraints 0.3 (1/m) and 0.8 (1/m) respectively by our proposed method. On the other hand, streamline path with curvature constraints 1 (1/m) collides with an obstacle and fails due to late response. In sum, our proposed method performed well and can have a higher probability to get feasible paths with small maximum curvature.

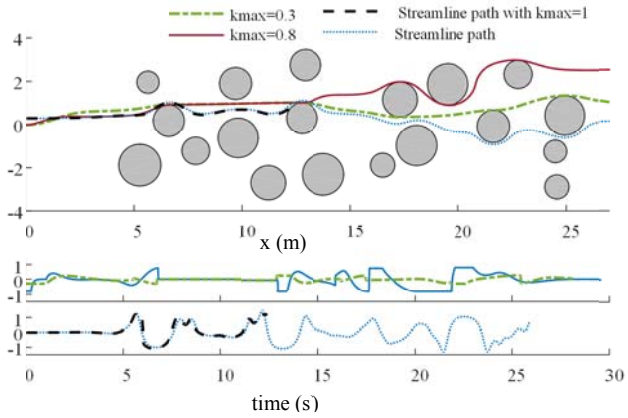

Fig. 20. A cluttered environment: upper plot -the paths, lower plots-the curvature. Black dashed path denoting the path generated by [7], [14] collides with an obstacle.

5.3 Proof of Concept Experiment and Discussion

In the indoor experiment, three aspects related to the feasibility of trajectory are presented, which are (1) curvature constraint, (2) arrangement of obstacles, and (3) sensor detection error. First, while maximum curvature constraint decreases, it's more difficult for robots to achieve primitive paths. Second, a robot needs to keep enough clearance from the obstacle to enable pursuing all three primitive paths. Third, we rely on low-cost sonar and infrared sensors to estimate obstacle location. Furthermore, in order to guarantee obstacle avoidance, the obstacles are arranged so that only one obstacle is detected within a pre-specified lookahead distance of the mobile robot's current location at a time.

5.3.1 Experimental setting

The mobile robot is initially located at the origin of the global coordinate system and its forward moving direction is + x axis with constant linear velocity U=0.5 m/s. Several location-unknown obstacles are all assumed identical cylinders with radius r_{Obs} = 0.1 m, which are distributed around the forward motion route. We arrange the clearance between any two adjacent obstacles smaller than the sensing range of the sensors but large enough to allow the pure pursuit algorithm to generate a local collision-free path. Hence, the enlarged radius of an obstacle $r_{obs} = r_{Robot} + r_{Obs} + r_{Safe}$ is 0.4m.In addition, the minimum distance between two obstacle's center is $2r_{Obs}$ which is wide enough for the robot to pass between two obstacles. The maximum allowable curvature for the mobile robot is set as 1.5 (1/m), as shown in Table 1. The pure pursuit command rate is 10 Hz, and the safety distance is 0.1 m, which is the robot displacement in a period. In our experiment, the lookahead distance equals to the robot radius 0.2 m.

5.3.2 On-line static cylinder obstacles avoidance

The experimental setup and the trajectory are depicted in Fig. 22 with Fig. 21 showing the velocity field generated by the gradient of the solution to Laplace equation in this map. The map of the environment is created by Hector SLAM, an open source SLAM algorithms available in ROS [23]. Similar to [11], there are four cylinder obstacles placed at (1, 0), (1.8, -0.6), (2.6, 0) and (2.6, -1.2) in meter. We assume the projection of obstacle onto the ground plane is an identical circle, but the number of static obstacles and their locations are unknown. The primitive paths can be computed within 0.2 milliseconds in our implementation. The robot can autonomously localize itself once new sensor readings are available (within 1millisecond).In the experiments, two feasible smooth paths shown in Fig.9are found for on-line safe navigation. Different paths are obtained due to the slight variation in initial position and heading of the mobile robot.The location of detected obstacle are oscillating because of sensor noise and detection error. Fig.23 shows the velocities and the path curvature profiles of two navigation paths generated during experiments depicted in Fig. 22.

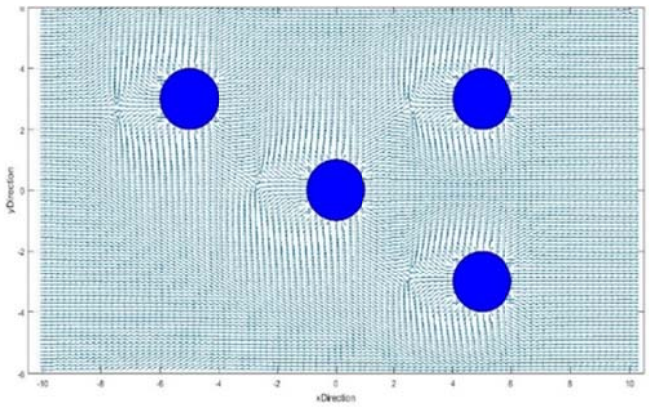

Fig. 21 The smooth velocity field generated by Laplace equation with Dirichlet boundary conditions. in the experiment scenario depicted in Fig. 22, assuming bounded rectangular domain with circular obstacles sufficiently apart.

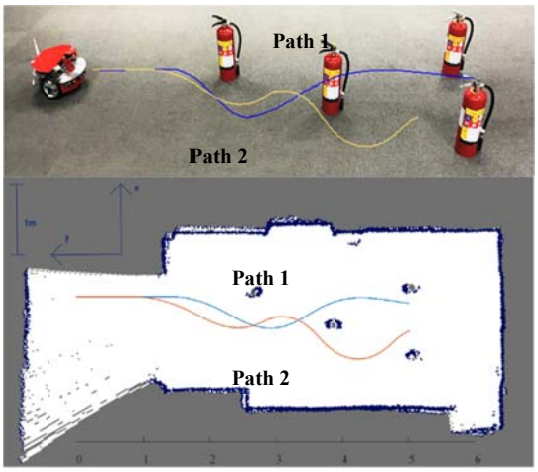

Fig. 22 Experimental setup (upper: the photo, lower: the mapping) and two resulting obstacle-avoiding paths generated online by the streamline-based approach.

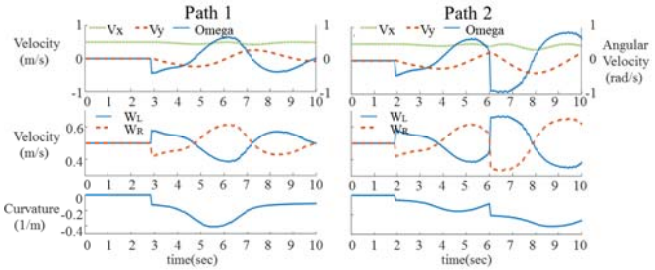

Fig. 23 Robot velocities, speeds of left and right wheels and curvature profiles of the two paths depicted in Fig. 9, where v_x is the longitudinal velocity and v_y is the lateral velocity of the robot.

6. Conclusion

A hydrodynamic motion planning method is implemented to generate smooth collision-free paths based on streamlines. The streamlines provide a pool of systematic smooth paths that have explicit and easily computable vector fields useful for specification of path tangent at each point for navigation guidance of autonomous vehicles. We demonstrate their potential for nonholonomic robots to avoid obstacles with curvature constraint in this paper. First, streamlines extracted from harmonic potential field are used to design three primitive smooth paths for a single obstacle avoidance along with their application situations according to a lateral distance relative to obstacle size. Second, for local multiple- obstacles avoidance situation, pure pursuit algorithm is implemented to pursuit streamline changing paths satisfying the curvature constraint among multiple obstacles. Simulation results show that pure pursuit paths in combination with initially planned streamlines can find feasible paths with smaller curvature constraint compared with previous fluid motion approaches. Furthermore, proof of concept experiment was conducted to validate the feasibility of safe, smooth navigation of two-wheel driving mobile robots via the proposed strategy in an environment composed by obstacles with large separation distance enabling the feasibility of pure pursuit path. Future work is planned to focus on goal reaching, the extension to 3D space, and toward the safe navigation in more complex environments with the anytime, complete real-time HPF- based path planner.

References

[1] J. F. Dong, S. E. Sabastian, T. M. Lim, and Y. P. Li, "Autonomous In-door Vehicles," in *Handbook of Manufacturing Engineering and Technology*, Springer, 2015.

[2] L. Adouane, *Autonomous Vehicle Navigation: From Behavioral to Hybrid Multi-Controller Architectures*, CRC Press, 2016.

[3] V. Kunchev, L. Jain, V. Ivancevic, and A. Finn, "Path planning and obstacle avoidance for autonomous mobile robots: A review," *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, pp. 537-544, 2006.
[4] J. Minguez, F. Lamiraux, and J.-P. Laumond, "Motion planning and obstacle avoidance," *Springer Handbook of Robotics*, pp. 1177-1202, 2016.
[5] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *IEEE International Conference on Robotics and Automation*, pp. 500-505, 1985.
[6] F. Fahimi, "Obstacle avoidance using harmonic potential functions," in *Autonomous Robots*, pp. 1-49, Springer, 2009.
[7] C. I.Connolly and R. A.Grupen,On the applications of harmonicfunctionsto robotics," Journal of Robotic Systems, vol.10, no.7, pp.931-946, 1993.
[8] M. Kaľavský and Ž. Ferková, "Harmonic potential field method for path planning of mobile robot," ICTIC, vol.1, issue.1, pp.41-46, 2012.
[9] S. Akishita, S. Kawamura, and K. Hayashi, "New navigation function utilizing hydrodynamic potential for mobile robot," *IEEE International Workshop on Intelligent Motion Control*, pp. 413-417, 1990.
[10] K. H. Wray, D. Ruiken, R. A. Grupen, and S. Zilberstein, "Log-space harmonic function path planning," *IEEE/RSJ International Conference onIntelligent Robots and Systems,* pp. 1511-1516, 2016.
[11] S. Waydo and R. M. Murray, "Vehicle motion planning using stream functions," *IEEE International Conference on Robotics and Automation*, pp. 2484-2491,2003.
[12] R. Daily and D. M. Bevly, "Harmonic potential field path planning for high speed vehicles," *American Control Conference*, pp. 4609-4614, 2008.
[13] T. Owen, R. Hillier, and D. Lau, "Smooth path planning around elliptical obstacles using potential flow for non-holonomic robots," *Robot Soccer World Cup*, pp. 329-340, 2011.
[14] R. Palm and D. Driankov, "Fluid mechanics for path planning and obstacle avoidance of mobile robots," *11th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, pp. 231-238, 2014.
[15] J. O. Kim and P. Khosla, "Real-time obstacle avoidance using harmonic potential functions," *IEEE International Conference on Robotics and Automation*, pp. 790-796, 1991.
[16] H. Wang, W. Lyu, P. Yao, X. Liang, and C. Liu, "Three-dimensional path planning for unmanned aerial vehicle based on interfered fluid dynamical system," *Chinese Journal of Aeronautics,* vol. 28, pp. 229-239, 2015.
[17] C. I. Connolly and R. A. Grupen, "Nonholonomic path planning using harmonic functions," Citeseer, 1994.
[18] D. Lau, J. Eden, and D. Oetomo, "Fluid Motion Planner for Nonholonomic 3-D Mobile Robots With Kinematic Constraints," *IEEE Transactions on Robotics,* vol. 31, pp. 1537-1547, 2015.
[19] B. R. Munson, D. F. Young, and T. H. Okiishi, Fundamentals of Fluid Mechanics, 1990.
[20] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," DTIC Document, 1992.
[21]J. Morales, J. L. Martínez, M. A. Martínez and A. Mandow, "Pure-pursuit reactive path tracking for nonholonomic mobile robots with a 2D laser scanner," *EURASIP Journal on Advances in Signal Processing,* vol. 2009, Article ID 935237, 10 pages, 2009.
[22] A. A. Masoud, "Kinodynamic motion planning,"*IEEE Robotics& Automation Magazine*, vol.17, no. 1, pp.85-99, 2010.
[23] S.Kohlbrecher, O. Von Stryk, J. Meyer and U.Klingauf, "A flexible and scalable slam system with full 3d motion estimation," *IEEE International Symposium on Safety, Security, and Rescue Robotics*, 2011.
[24] M. A.Belabbas and S.Liu, "New method for motion planning for non-holonomic systems using partial differential equations,"*American Control Conference,* pp. 4189-4194, 2017.
[25] D. Gingras, E. Dupuis, G. Payre & J. de Lafontaine,. Path planning based on fluid mechanics for mobile robots using unstructured terrain models. *IEEE International Conference on Robotics and Automation,* pp. 1978-1984, *2010* .
[26] K. Motonaka. Kinodynamic Motion Planning for a Two-Wheel-Drive Mobile Robot. In *Handbook of Research on Biomimetics and Biomedical Robotics* ,pp. 332-346, IGI Global, 2018.
[27] C. A. Liu, Z. Wei & C.. Liu. A new algorithm for mobile robot obstacle avoidance based on hydrodynamics. *IEEE International Conference on Automation and Logistics,* pp. 2310-2313, 2007.
[28] Y. Golan, S. Edelman, A. Shapiro & E. Rimon. Online Robot Navigation Using Continuously Updated Artificial Temperature Gradients. *IEEE Robotics and Automation Letters*, *2*(3), 1280-1287, 2017.
[29]C. Louste & A. Liégeois. Path planning for non-holonomic vehicles: a potential viscous fluid field method. *Robotica*, *20*(3), 291-298, 2002.
[30] M. D. Pedersen & T. I. Fossen. Marine vessel path planning & guidance using potential flow. *IFAC Proceedings Volumes*, *45*(27), pp.188-193, 2012.