COMPUTER-AIDED DESIGN & APPLICATIONS, 2016
VOL. 13, NO. 3, 379–388
http://dx.doi.org/10.1080/16864360.2015.1114395

A Spline-based Flexible Method of Virtual Force Design for Dynamic Motion Planning of Robots

S.H. Choi and W.K. Zhu

The University of Hong Kong, Hong Kong

ABSTRACT
Virtual force approach is preferable for motion planning of mobile robots in dynamic environments. It composes virtual attractive forces to drive robots to destinations and virtual repulsive forces to steer robots away from neighbouring robots. However, most traditional methods use functions with limited controllable parameters, compromising design flexibility essential for smooth yet responsive robot motions.

This paper proposes a spline-based method to enhance design flexibility of virtual forces that streamline robot motions. We take advantage of local controllability of interpolating cubic splines to generate desirable smooth virtual forces for robots. This merit facilitates responsive robot motions in dynamic situations. A case study of autonomous military robots is presented to validate the approach, in terms of enhanced motion safety and shortened operational time.

KEYWORDS
Splines; force design; robot motion planning; dynamic environments

1. Introduction

1.1. Background

Motion planning for mobile robots is a problem on how a team of robots can achieve collective motion objectives in a shared working place while avoiding interference with one another [16]. Research in this field dates back to the late 1980 s and remains one of the most difficult and important problems in multi-robot control, particularly in dynamic environments. Various applications cover this problem, such as logistics [8], military operations [10], and computer graphics [23].

The environments of motion planning can be classified into two categories, namely static and dynamic [14]. In a static environment, the situation is completely known and remains constant, in terms of layout, obstacles, tasks, traffic conditions, etc. In a dynamic environment, on the other hand, routing decisions for motion planning are based on frequently changing conditions. The advantage is that route validity can be maintained during operations even in uncertain circumstances, such as stochastic task requests and dynamic traffic conditions. However, optimality of the routing cannot be ensured. This paper addresses motion planning problems in dynamic environments.

Performance of a robot team can be evaluated from a number of perspectives, for instance, time spent in a mission, energy consumed, distance travelled, motion safety, or a combination of these criteria [16]. This paper is aimed to improve robot team performance by enhancing motion safety and shortening operational time. Safety is paramount in making sure missions are accomplished without suspension and failure. With safety ensured, operational time is particularly crucial in fast-paced and tightly-scheduled missions.

Alongside motion planning, the issue of task allocation, which distributes jobs to the robots, also plays a pivotal role in improvement of operational time. To address this issue, we have proposed an auction-based method which features a closed-loop bid adjustment mechanism [25]. To make fair comparisons between different motion planning methods, this paper uses the same task allocation scheme with our auction-based method.

There are some popular techniques for multi-robot motion planning, such as protocol-based methods [15], sampling-based techniques [11], genetic algorithms [21], spatiotemporal planning [2], potential field methods [12], virtual force approaches [7], graph-based techniques [13], etc.

Each of these techniques has its own pros and cons. Virtual force approach is generally preferable for robots to perform team work in a dynamic environment. This approach composes virtual attractive forces that drive the robots towards their targets and, at the same time, exert repulsion to steer the robots away from the obstructing robots [9][18]. Similar approaches are potential field

CONTACT S.H. Choi shchoi@hku.hk

methods, which direct the robots as if they were particles moving in a force field. These methods require full information of static environments beforehand and suffer local minimum [12].

Helbing et al. [7] proposed an empirical force model to simulate dynamics of multiple mobile agents. This model kept the agents away from one another by a repulsive interaction force and adopted a body force to counteract body compression between any agents. Although this model was derived from real data, its disadvantage was that the agents were modelled as particles which tended to shake in response to numerous forces in high-density crowds.

Diller et al. [4] presented a method to control multiple micro robots in three dimensions using magnetic gradient. Micro robots assumed different magnetic directions in a rotating or oscillating magnetic field. The proposed method could be used for 3D control of a team of micro robots inside micro fluidic channel or human body for localized therapy or diagnostics. Nevertheless, it was only effective in controlling two robots.

The design of virtual forces plays a pivotal role in improving robot motions to avoid robot collisions and enhance overall team performance. However, most traditional methods lack design flexibility to locally adjust the virtual forces according to the changing situations, hampering responsiveness of robot motions in dynamic environments and leading to possible robot collisions.

Concerning the attractive force, a traditional method is to formulate a constant force as $\vec{f}_{goal} = K_1\vec{u}$, where K_1 is a constant and $\vec{u}$ is a unit vector pointing from the robot to its destination. The dashed line in Fig. 1 shows the corresponding force design and its drawback. Another traditional method is to adopt the force magnitude linearly proportional to the distance from the goal, in the form of $\vec{f}_{goal} = K_2\vec{x}_{goal}$, where $\vec{x}_{goal}$ is a distance vector pointing from the robot to its destination. Solid line in Fig. 1 shows this force design and its deficiencies. Although the constant K_2 can be tuned to adjust the force design, the flexibility of controlling the curve shape locally is still much limited.

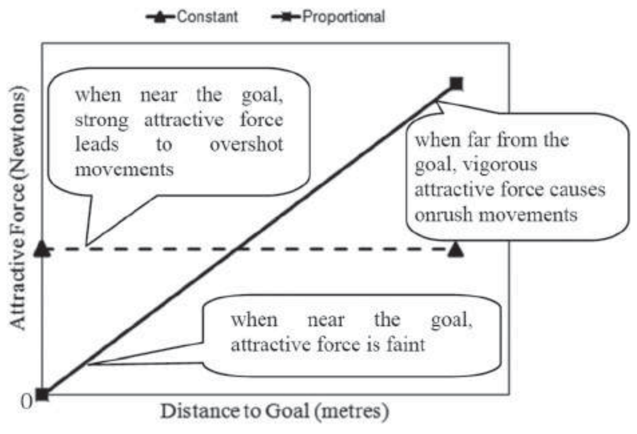

Figure 1. Traditional designs for attractive force and their drawbacks.

Turning to repulsive force, most traditional methods use an inversely proportional function to calculate the force: $\vec{f}_i = \frac{A}{\vec{x}_i}$, where $\vec{x}_i$ is a distance vector pointing from neighbour i to the robot concerned; and A is a scaling factor. The corresponding force design and it drawbacks are shown in Fig. 2. The flexibility of force design is poor as well, for there is only one controllable parameter, A. For example, if the magnitudes of repulsive forces in the outer sensing range are strengthened to steer the robots, it will inevitably jeopardize the force magnitudes in the inner sensing range.

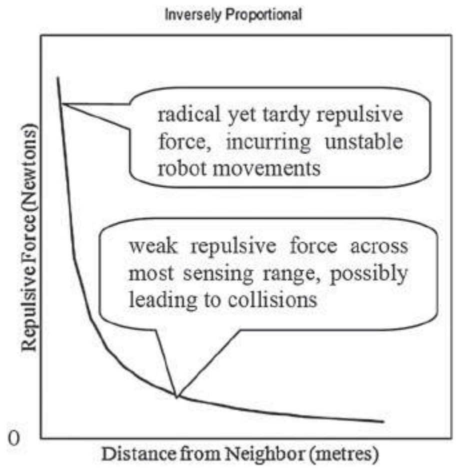

Figure 2. Traditional design of repulsive force and its drawbacks.

Overall, most traditional methods use simple functions to design virtual forces, for the sake of reducing computational burden of online force calculations. Although such methods are computationally effective, they generally lack design flexibility, which is crucial to generate adaptive and responsive virtual forces to suit dynamic situations.

1.2. Research objective

It is therefore desirable to develop a flexible method of enhancing the design freedom of virtual forces, while maintaining computational efficiency of the force functions. For this purpose, we propose a spline-based design method for both attractive and repulsive virtual forces. Splines are traditionally used in geometric applications, like curve fitting and surface design. This paper features interpolating cubic splines for force design. Interpolating user-assigned control points facilitates flexible force design to generate desirable force magnitudes at specified locations of each robot. The shape of a spline can be locally adjusted to modify the force magnitudes, without jeopardizing the interpolation properties of other control points. The virtual forces, generated by flexibly designed spline functions, can drive the robots responsively in accordance with real-time situations during motions. As such, operations of robot team become safer and overall operational time is shortened considerably.

2. Spline Functions and Curves

Splines are a common type of polynomial curves. A spline consists of piecewise polynomial curves that are differentiable up to a prescribed degree [5]. The term "spline" originally refers to a thin strip of wood or metal mounted through some desired points for ship hull design. Other traditional applications of splines include automotive and aircraft industries, curve and surface design in computer graphics, and data fitting in statistics [22]. Some examples of splines are interpolating spline [22], approximating spline [5], Bézier curve [20], B-spline [3], and Non-Uniform Rational B-Spline (NURBS) [17], which are summarised in Table 1.

This paper chooses interpolating cubic spline because of the following four main merits.

Firstly, an interpolating cubic spline facilitates flexible design for virtual forces. The spline passes through user-specified control points, generating desired force magnitudes at specified positions of each robot. Meanwhile, change of a control point will not jeopardize properties of other segments.

Secondly, in comparison with linear or quadratic splines, a cubic spline not only provides continuous forces with smooth changing rates, but also features second derivatives of forces. This feature appeals a lot to some advanced actuators of robots that can output more agile forces.

Thirdly, the degree of an interpolating cubic spline is independent of the number of control points, and remains persistently cubic and numerically stable over all the curve segments. This merit allows more control points for flexible force design, while keeping numerical stability with minimum interpolation errors possible.

Fourthly, this method is computationally efficient. An interpolating cubic spline can be described by a set of cubic polynomial functions, whose coefficients can be easily pre-calculated offline by standard methods.

An interpolating cubic spline with (n-1) segments meets the following conditions:

(1) Each curve segment is described by a cubic polynomial function:

$$S(x) = s_i(x) = \sum_{k=0}^{3} \alpha_{i,k}(x - x_i)^k,\ i = 1,\ 2,\ \ldots,\ n-1$$

(2) The spline function is C^2 smooth over the domain $[x_1,\ x_n]$;

(3) The spline interpolates all the given points $\{(x_i,\ y_i)|\ i = 1\ 2, \ldots, n\}$: $S(x_i) = y_i$.

It is efficient to derive the coefficients and to uniquely define the spline functions. For each curve segment, there are four coefficients: $\alpha_{i,k},\ k = 0,\ 1,\ 2,\ 3$. For an entire spline consisting of n-1 curve segments, 4(n-1) coefficients are needed. A n interpolating cubic spline $S(x)$ is C^2 smooth at every inner point and gives three equations:

$$s_{i-1}(x_i) = s_i(x_i), s'_{i-1}(x_i) = s'_i(x_i), s''_{i-1}(x_i) = s''_i(x_i)$$

where i = 2, 3, . . . , n-1. For n-2 inner points, we have 3(n-2) equations.

To meet the interpolation condition (3), for all the n control points, we have other n equations:

$$S(x_i) = y_i,\ i = 1,\ 2,\ \ldots,\ n$$

The number of additional equations is: 4(n-1) - 3(n-2) - n = 2. These two equations are given by the boundary

Table 1. Summary of the properties of different splines of degree *m*.

	Interpolation and local control	Smoothness	Stability	Derivation
Interpolating spline	Satisfactory	C^{m-1}	*m* order stable, independent of number of control points	Standard, efficient
Approximating spline	No interpolation with acceptable local control	C^{m-1}	*m* order stable, independent of number of control points	Standard, efficient
Bézier curve	No interpolation with poor local control	C^{m-1}	Degree is *m*, given *m*+1 control points. Higher degree may lack stability	Standard, efficient
B-spline	Satisfactory local control, but no interpolation unless additional parametric points are supplemented	C^{m-1}	Stable if degree *m* is properly small	Standard, efficient
NURBS	Excellent local control, but no interpolation unless additional parametric points are supplemented	C^{m-1}	Stable if degree *m* is properly small	Standard, efficient

conditions at two end points. For efficient computation and minimum curvature at the ends, natural cubic spline boundary conditions are used in this paper, giving:

$$s_1''(x_1) = 0 \quad , \text{ and } \quad s_{n-1}''(x_n) = 0$$

Now given these 4(n-1) equations, we can derive the 4(n-1) coefficients $\alpha_{i,k}$, $k = 0,\ 1,\ 2,\ 3$, where i = 1 2, . . . , n-1, and hence uniquely define an interpolating cubic spline.

3. Procedure and Guideline of the Spline-based Flexible Method

This section presents the details of the proposed flexible spline-based method of virtual force design for dynamic motion planning of robots. As shown in Fig. 3, the implementation of the spline-based virtual force design consists of four steps. The first three steps are carried out offline while the last step is realised online.

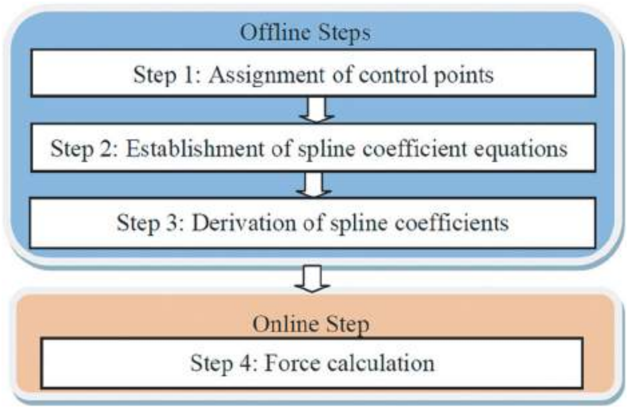
Offline Steps
Step 1: Assignment of control points
Step 2: Establishment of spline coefficient equations
Step 3: Derivation of spline coefficients
Online Step
Step 4: Force calculation

Figure 3. Implementation of the spline-based virtual force design.

In step 1, the number of control points and their desired positions are specified, based on the following empirical guideline. A spline is uniform in that the control points are evenly distributed across the spline. For the attractive force function, it is monotonically increasing. Two end points of the spline are specified first, according to the application requirements, such as the longest distance to the goal and the desired driving force near the destination. One inner point is used to tune the shape of the spline to be convex or concave, so that the anticipated force magnitude and its changing rate can be achieved. For the repulsive force function, it is monotonically decreasing. The two end points are positioned first, subject to the desired force value at the brim of the sensing range and the strongest repelling force when two robots almost collide. Some inner points are then specified to divide the sensing range into some sections that correspond to different levels of collision danger. The inner points are also used to tune the shape of the spline locally to achieve the anticipated force values and their changing rates.

In step 2, some cubic polynomial functions are used to describe the curve segments of a spline, with a number of coefficients to be derived. A set of linear coefficient equations are established according to the spline properties, like smoothness, interpolation, and end point boundary conditions.

In step 3, the unknown coefficients are derived by a linear algebra algorithm, and hence the spline functions for both the attractive force and the repulsive force are uniquely defined.

After these three offline steps, the spline functions in step 4 can be applied to each robot for online calculations of virtual forces, according to the real-time dynamic situations in motion.

4. Case Study: Military Robots at Battle Fields

A simulator is implemented to validate the proposed spline-based method of virtual force design. The simulator is developed using the C++ programming language on the Player/Stage open-source platform, which is widely used for multi-robot control and simulation [19]. The Player/Stage runs in a Linux-based operating system called Fedora.

The simulator has been used for simulation of various operations, including AGVs at container terminals, military robots in battle fields, and rescue robots in disaster areas. This section presents simulation of a team of autonomous military robots to destroy enemies.

Military robots have been receiving increasing popularity in modern battle fields, such as unmanned aerial vehicles for reconnaissance, ground robots for combats, and underwater robots for scouting [6], [24]. Fig. 4 shows a simulated battle field, which involves sixteen military robots fighting together in a mission to destroy five types of enemies. The battle field covers an area of 3800 m $\times$ 2500 m. Each robot, modelled as a small black triangle, measures 10m$\times$4 m and weighs 30 tonnes. The maximum speed is 35m·s^{-1}. The maximum acceleration and deceleration are 3m·s^{-2} and -3m·s^{-2}, respectively. The sensing range of a sonar is 400 m.

It is assumed that a military satellite spies above the battle field and commands the robots to destroy the enemies detected. The stealth robots prowl to the enemies and destroy them, without being discovered and fought back. This assumption is seen in a number of realistic military operations, such as unmanned aerial vehicles (UAVs) in stealth attack missions and submarines to secretly assault the adversarial warships [1]. The five

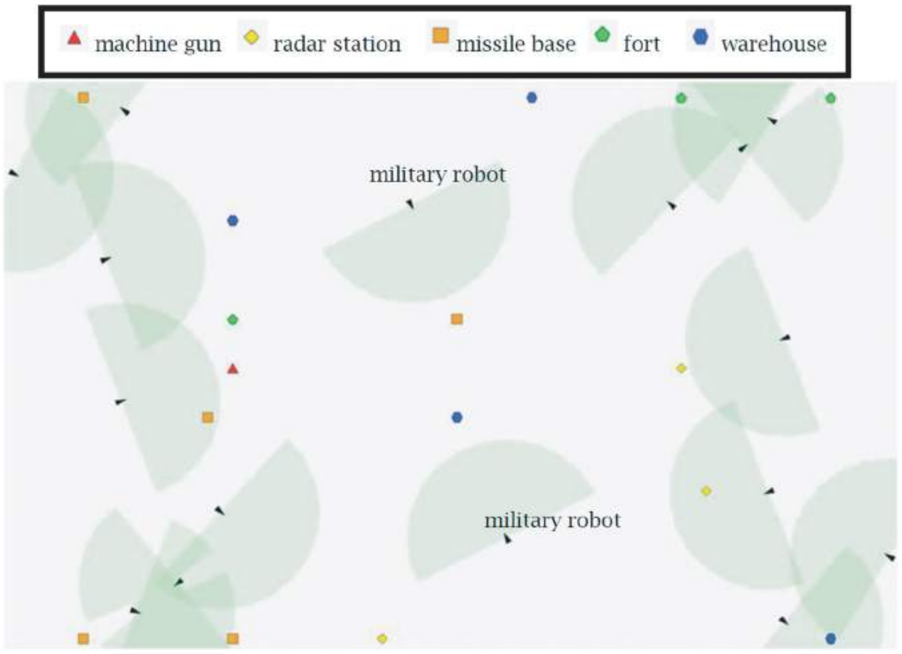

Figure 4. A simulated battle field with sixteen robots to destroy five types of enemies.

Table 2. Mission description of sixteen robots to destroy five types of enemies.

Types of Enemies	Time to Destroy (minute)	Amount	Characteristics
Machine gun	0.25	100	• appear stochastically and continuously
Radar station	1	10	
Missile base	1	10	• stationary
Fort	0.5	50	• attacking targets rather than the stealth robots
Warehouse	1	20	

types of enemies appear stochastically and continuously at different locations. It is assumed that enemy targets are stationary, such as radar stations, missile bases, and forts.

A mission for the sixteen stealth robots to destroy five types of enemies is described in Table 2. The robots are assumed to be capable of destroying all the five types of enemies. There are two major operational uncertainties, namely unpredictable appearance of enemies and dynamic interferences between the comrade robots.

The following section presents how the interpolating cubic splines are applied in both the attractive force and repulsive force, respectively.

To describe the dynamics of the virtual forces on the motion of each robot, a navigation function is formulated in the form of Newton's law of motion: $\vec{f}_{net} = \frac{d(m\vec{v})}{dt} = \vec{f}_{goal} + \sum \vec{f}_i$, where $\vec{f}_{net}$ is the net force, $\vec{f}_{goal}$ is the attractive force by the destination, while $\sum \vec{f}_i$ is the composition of the repulsive forces from the imminent neighbours of a robot.

Design of the attractive force and the repulsive force for this case study is illustrated as follows.

4.1. Attractive force

- *Step 1: Assignment of control points*

The assignment of control points in attractive force design follows such a guideline. The attractive force spline is monotonically increasing. The two end points of the spline are specified first, according to the longest distance to the target and the desired driving force near the goal, respectively. One inner point is used to tune the shape of the spline to be convex, so that the attractive force is more vigorous than the average and the changing rate near the goal is sharper to achieve swifter motion adjustment.

When a robot is approaching its enemy, the attractive force should be neither too strong to make the robot overshoot its target, nor too weak to delay the attack. A moderate value of 12000N is designed at the goal position, giving a control point (0, 12000), as the red dot on the left in Fig. 5.

With respect to the other end, since the battle field is 3800 m $\times$ 2500 m, the longest distance between two locations is $\sqrt{3800^2 + 2500^2} \approx 4600$m. Given the weight of a robot 30 tonnes and the maximum acceleration of 3m·s^{-2}, the maximum attractive force is 30×10^3 kg $\times$ 3m·s^{-2} $=$ 90000N. The other end point is (4600, 90000).

A middle control point between the two end points is placed to adjust the shape of the spline and its magnitude

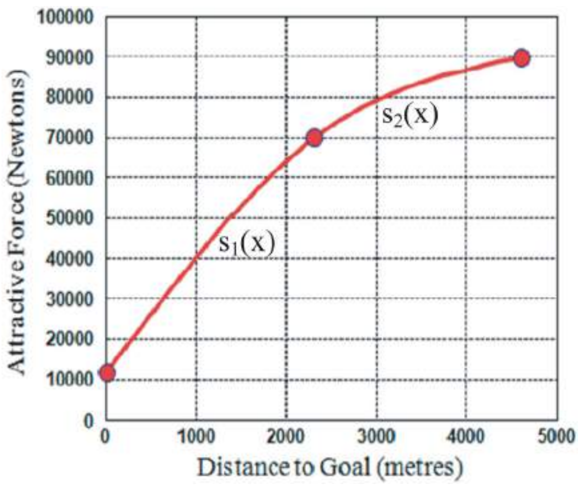

Figure 5. Design of attractive force.

should preferably be stronger than the average, to maintain a vigorous attractive force during motion. This gives another control point (2300, 70000).

- *Step 2: Establishment of spline coefficient equations*

Given the three control points: (0, 12000), (2300, 70000), and (4600, 90000), an interpolating cubic spline, consisting two curve segments, is uniquely defined. Each curve segment is described by four coefficients and there are eight coefficients to be derived.

- *Step 3: Derivation of spline coefficients*

The coefficient vector is derived as:

$$\alpha = [a_1 \ b_1 \ c_1 \ d_1 \ a_2 \ b_2 \ c_2 \ d_2]^T$$

$$= \begin{matrix} [12000 & 29.3 & 0 & -0.00000078 \\ 70000 & 17.0 & -0.00539 & 0.00000078]^T \end{matrix}$$

Hence, the spline is uniquely defined over the domain [0, 4600] by Eqn. (4.1):

$$S(x) = \begin{cases} s_1(x) = 12000 + 29.3x - 0.00000078x^3, \\ \qquad 0 \le x < 2300 \\ s_2(x) = 70000 + 17.0(x - 2300) \\ -0.00539(x-2300)^2 \\ \quad +0.00000078(x-2300)^3, \\ \qquad 2300 \le x \le 4600 \end{cases} \tag{4.1}$$

The derived interpolating cubic spline for attractive force control is shown in Fig. 5.

4.2. Repulsive force

- *Step 1: Assignment of control points*

The assignment of control points in repulsive force design follows such a guideline. The repulsive force spline is monotonically decreasing. The two end points are positioned first, subject to the desired force value at the brim of the sensing range and the strongest repelling force when two robots almost collide, respectively.

Some inner points are then specified to divide the sensing range into some sections that correspond to different levels of collision dangers. The inner points are also used to tune the shape of the spline locally to achieve the anticipated force values and their changing rates.

Regarding an end point, when a neighbour is detected at brim of the sensing range of 400 m, the repulsive force begins to take effect. It gives the first control point (400, 0), as shown in Fig. 6. Since the mass of a robot is 30 tonnes and the maximum deceleration is -3m·s^{-2}, the maximum magnitude of the repulsive force is: $30 \times 10^3\text{kg} \times 3\text{m}\cdot\text{s}^{-2} = 90000\text{N}$, giving the other end point (0, 90000).

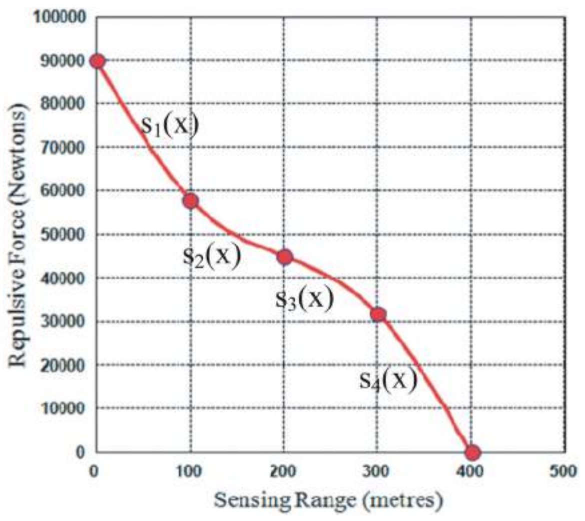

Figure 6. Design of repulsive force.

At the switching boundary between the outer half and the inner half of the sensing range: $\frac{R}{2} = 200m$, the force magnitude is designed to be half of the maximum repulsive force: $\frac{90000}{2} = 45000\text{N}$, giving a control point (200, 45000). This middle point also serves to ensure smoothness of the force curve at the switching boundary.

Concerning the outer half of the sensing range, it is preferable to generate relatively stronger force to steer the robot in an earlier stage. A sub-middle point (300, 32000) is specified to generate vigorous repulsive force.

Turning to the inner half of the sensing range, the force spline should have increasingly sharp derivative to provide swifter responsiveness. A sub-middle point (100, 58000) is assigned to make two slightly concave curve segments in the inner half of sensing range.

- *Step 2: Establishment of spline coefficient equations*

Given the five control points: (0, 90000), (100, 58000), (200, 45000), (300, 32000), and (400, 0), an interpolating cubic spline, consisting four curve segments, is uniquely defined. Each curve segment is described by four coefficients and there are sixteen coefficients to be derived.

- *Step 3: Derivation of spline coefficients*

The coefficient vector is derived as:

$$\alpha = [a_1\ b_1\ c_1\ d_1\ a_2\ b_2\ c_2\ d_2\ a_3\ b_3\ c_3\ d_3\ a_4\ b_4\ c_4\ \ d_4]^T$$
$$= \begin{matrix} [90000 & -367.5 & 0 & 0.0047 \\ 58000 & -225.0 & 1.42499 & -0.00475 \\ 45000 & -82.5 & 0 & -0.00475 \\ 32000 & -225.0 & -1.42501 & 0.00475]^T \end{matrix}$$

Thi s spline is uniquely defined over domain [0, 400] by Eqn. (4.2):

$$S(x) = \begin{cases} s_1(x) = 90000 - 367.5x + 0.00475x^3, \\ \qquad 0 \le x < 100 \\ s_2(x) = 58000 - 225.0(x-100) \\ \quad +1.42499(x-100)^2 - 0.00475(x-100)^3, \\ \qquad 100 \le x < 200 \\ s_3(x) = 45000 - 82.5(x-200) \\ \quad -0.00475(x-200)^3,\ 200 \le x < 300 \\ s_4(x) = 32000 - 225.0(x-300) \\ \quad -1.42501(x-300)^2 + 0.00475(x-300)^3, \\ \qquad 300 \le x \le 400 \end{cases} \tag{4.2}$$

The corresponding interpolating cubic spline is shown in Fig. 6.

- *Step 4: Online force calculation*

After the offline design of interpolating cubic splines for both the attractive force and the repulsive force, the spline functions are ready for online dynamic motion planning for the team of military robots. Simulation and comparison results are presented in the following section.

4.3. Comparison with traditional force design

To validate the proposed approach in terms of motion safety, two methods of force design were compared, as illustrated in Fig. 7(a) and Fig. 7(b). These include: (I) traditional method with proportional attractive force and inversely proportional repulsive force and (II) the spline-based method for both the attractive force and the repulsive force. Two methods accomplished the same mission as described in Table 2.

In Fig. 7(a), two robots collided near the left bottom corner, as highlighted in the dashed circle. The weak repulsive forces to the two robots by only a single inversely proportional function led to this collision.

In Fig. 7(b), the spline-based method streamlined the motions of the two robots, allowing them to operate safely with a comfortable distance between each other.

This validated that the spline-based method could responsively tune the force magnitudes, according to the real-time dynamic situations. As such, the motions of robots are streamlined and the likelihood of robot collisions reduced.

To further evaluate the shortened operational time, each method was tested with five simulation runs. Average values and standard deviations are shown in Fig. 8. Compared with traditional method, the spline-based method could shorten the mission time by 27.8%.

This confirmed that the flexibly designed splines can formulate responsive virtual forces for the robots in a dynamic battle field. The operational efficiency of the military mission was secured accordingly.

5. Discussion on Implementation to Physical Robots

5.1. Migration to physical robots

The Player/Stage robotic simulation system has been widely used in multi-robot control and simulation. It consists of two sub-packages, namely Player and Stage. Player provides a network interface to a variety of physical robots and sensors. Stage is a Player plug-in simulation package which simulates a population of mobile robots moving and sensing in a 2D bit-mapped environment. Virtual devices of Stage present a standard Player interface, and hence no or few significant changes are required to move between simulation and hardware. Controllers designed in Stage have been demonstrated to work on various physical robots. Hence, the proposed motion planning module can be readily adopted for operations of physical robot teams to benefit various real-world applications, such as logistics, military, and disaster rescue.

On the other hand, for most robotic simulation systems, there are some inevitable discrepancies between the simulated and physical robot operations. Concerning our simulator, two implementation issues are worthy of discussion.

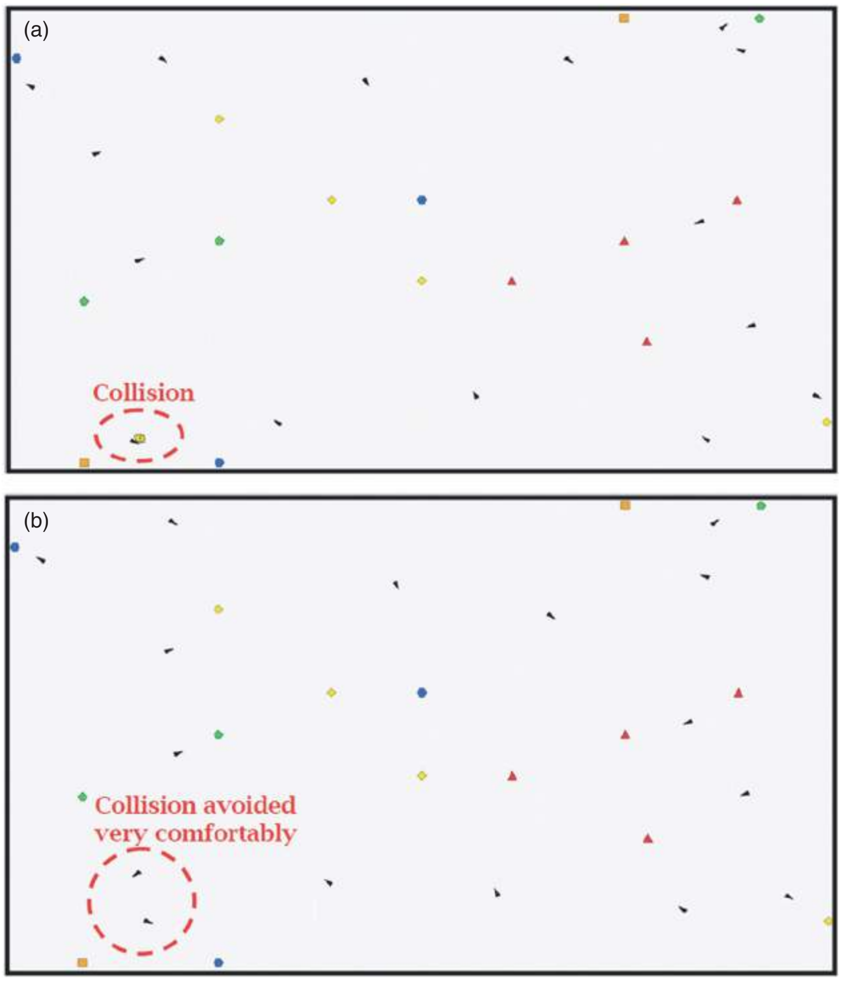

Figure 7. (a) Snapshot with proportional attractive force and inversely proportional repulsive force. (b) Snapshot with spline-based attractive force and repulsive force.

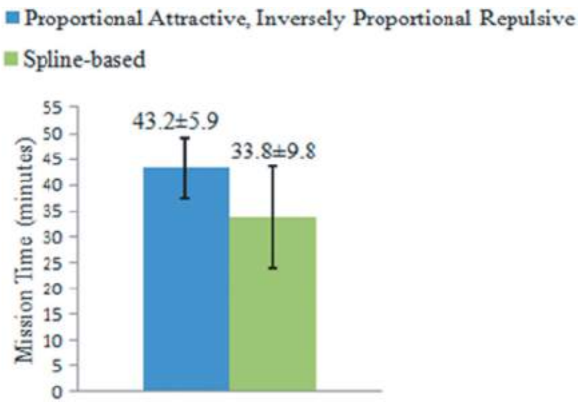

Figure 8. Mission times by two virtual force designs.

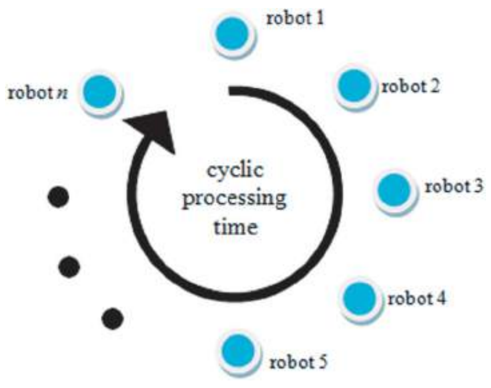

Figure 9. Cyclic processing of a robot team in simulation.

5.2. *Cyclic processing of a robot team*

The first issue is the cyclic processing mechanism of the robot team. Our method is applicable to physical robots, which are decentralised with independent and simultaneous motion planning. However, in computer simulations above, robots are inevitably processed sequentially and periodically by the computer program, as shown in Fig. 9. As such, the more robots involved in the team, the longer it takes to process and update all the robots during simulation. This would lead to a potentially dangerous situation where some robots that have not yet been updated with current sensing information may collide with other robots. Indeed, to simulate a large number of robots, the computer program design for the behaviours of each robot and the program control logic should be optimised to render the execution as efficiently as possible. Fortunately, implementation to physical robots does not suffer this limitation.

5.3. Reliability of sonars

Another issue is the reliability of the sonars. In the simulator, sonars work in an ideal way. In physical robot operations, however, possible power interruption will influence stability of power supply, causing some sensing noises. Then, dispersion and absorption of signals are attributed to the surface characteristics of the objects detected and the weather conditions. Finally, some possible signal noises caused by interference between different sources of signals need to be filtered appropriately. Hence, for physical robots, adequate tolerances and proper signal processing should be incorporated accordingly to ensure the measurements of the sonars to be reliable.

6. Conclusion and Future Work

This paper proposes a spline-based method to improve the design flexibility of virtual forces for dynamic motion planning of mobile robots. Interpolating user-assigned control points facilitates flexible force design that generates more desirable force magnitudes for each robot at specified locations. The shape of a spline can be flexibly designed and locally tuned to modify the force values, without jeopardizing the interpolation properties of other control points. The resulting smooth virtual forces provide agility needed for the robots to response to real-time and dynamic situations during operations. A military case study shows that the likelihood of robot collisions can be reduced and the operations become safer. Moreover, the overall operational time of the robot team is shortened substantially.

Nevertheless, it should be noted that, similar to most traditional force design methods, the virtual force design by our method is in accordance with the specific applications and on a trial-and-error basis. For example, the number of control points and their desired positions are designed with the proposed empirical guideline. As future work, it would indeed be more fruitful if a preliminary effective analytic guideline could be derived to assist the design of force splines. It is expected that, with such a guideline, the design process would be more effective and automatic.

ORCID

S.H. Choi http://orcid.org/0000-0003-1925-5962
W.K. Zhu http://orcid.org/0000-0001-8510-1000

References

[1] Bertuccelli, L. F.; Choi, H.-L.; Cho, P.; How, J. P.: Real-time multi-UAV task assignment in dynamic and uncertain environments. In: Proceedings of The American Institute of Aeronautics and Astronautics Conference on Guidance, Navigation, and Control. 2–4, August, 2009. Chicago, USA. pp. 389–395. (2009).

[2] Cortés, J.; Martínez, S.; Karatas, T.; Bullo, F.: Coverage Control for Mobile Sensing Networks, IEEE Transactions on Robotics and Automation, 20(2), 2004, 243–255. http://dx.doi.org/10.1109/TRA.2004.824698

[3] de Boor, C.: A Practical Guide to Splines. Springer-Verlag. pp. 113–114. 1978. http://dx.doi.org/10.1007/978-1-4612-6333-3

[4] Diller, E.; Giltinan, J.; Sitti, M.: Independent Control of Multiple Magnetic Microrobots in Three Dimensions, The International Journal of Robotics Research, 32(5), 2013, 614–631. http://dx.doi.org/10.1177/0278364913483183

[5] Egerstedt, M.; Martin, C.: Control Theoretic Splines: Optimal Control, Statistics, and Path Planning. Princeton University Press, New Jersey, USA. 2010.

[6] Faied, M.; Mostafa, A.; Girard, A.: Vehicle Routing Problem Instances: application to Multi-UAV Mission Planning. In: Proceedings of AIAA Guidance, Navigation, and Control Conference. 16–18 September, 2010. Toronto, Canada. pp. 124–131. (2010).

[7] Helbing, D.; Buzna, L.; Johansson, A.; Werner, T.: Self-Organized Pedestrian Crowd Dynamics: Experiments, Simulations, and Design Solutions, Transportation Science, 39(1), 2005, 1–24. http://dx.doi.org/10.1287/trsc.1040.0108

[8] Kala, R.; Warwick, K.: Multi-Level Planning for Semi-autonomous Vehicles in Traffic Scenarios Based on Separation Maximization, Journal of Intelligent and Robotic Systems. 72(3–4), 2013, 559–590. http://dx.doi.org/10.1007/s10846-013-9817-7

[9] Lakoba, T.I.; Kaup, D.J.; Finkelstein, N.M.: Modifications of the Helbing-Molnár- Farkas-Vicsek Social Force Model for Pedestrian Evolution, Simulation, 81(5), 2005, 339–352. http://dx.doi.org/10.1177/0037549705052772

[10] Lau, G.; Liu, H.: Real-Time Path Planning Algorithm for Autonomous Border Patrol: Design, Simulation, and Experimentation, Journal of Intelligent and Robotic Systems. 75(3–4), 2014, 517–539. http://dx.doi.org/10.1007/s10846-013-9841-7

[11] Li, Y.; Li D.; Maple, C.; Yue, Y.; Oyekan, J.: K-Order Surrounding Roadmaps Path Planner for Robot Path Planning, Journal of Intelligent and Robotic Systems, 75(3), 2014, 493–516. http://dx.doi.org/10.1007/s10846-013-9861-3

[12] Minguez, J.; Lamiraux, F; Laumond, J.-P.: Chapter 35: Motion Planning and Obstacle Avoidance. In: Bruno, S; Oussama, K.: Springer Handbook of Robotics, Springer, The Netherlands. 2008. http://dx.doi.org/10.1007/978-3-540-30301-5_36

[13] Nagarajan, U.; Kantor, G.; Hollis, R.: Integrated Motion Planning and Control for Graceful Balancing Mobile Robots, The International Journal of Robotics Research, 32(9–10), 2013, 1005–1029. http://dx.doi.org/10.1177/0278364913488011

[14] Nieto-Granda, C.; Rogers, J.G.; Christensen, H.: Coordination Strategies for Multi-robot Exploration and Mapping, The International Journal of Robotics Research, 33(4), 2014, 519–533. http://dx.doi.org/10.1177/0278364913515309

[15] Parker, L. E.: ALLIANCE: An Architecture for Fault Tolerant Multirobot Cooperation, IEEE Transactions on Robotics and Automation, 14(2), 1998, 220–240. http://dx.doi.org/10.1109/70.681242

[16] Parker, L.E.: Path Planning and Motion Coordination in Multiple Mobile Robot Teams. In: Meyers, R.: Encyclopedia of Complexity and System Science, Springer, The Netherlands 2009 http://dx.doi.org/10.1007/978-0-387-30440-3_344

[17] Piegl, L.; Tiller, W.: The NURBS Book (Second Edition). Springer-Verlag. New York, USA. 1997. http://dx.doi.org/10.1007/978-3-642-59223-2

[18] Piriyanont, B.; Uchiyama, N.; Sano, S.: Model Reference Control for Collision Avoidance of a Human-Operated Quadrotor Helicopter, Journal of Advanced Computational Intelligence and Intelligent Informatics, 15(5), 2011, 617–623.

[19] Player/Stage Project, http://playerstage.sourceforge.net/, Accessed Dec. 2014.

[20] Prautzsch, H.; Boehm, W.; Paluszny, M.: Bézier and B-Spline Techniques. Springer. Berlin, Germany. 2002. http://dx.doi.org/10.1007/978-3-662-04919-8

[21] Richards, A.; Bellingham, J.; Tillerson, M.; How, J.: Coordination and Control of Multiple UAVs, In: Proceedings of AIAA Conference on Guidance, Navigation, and Control, 3–5 August, 2002. Monterey, USA. pp. 146–153. (2002). http://dx.doi.org/10.2514/6.2002-4588

[22] Shikin, E. V.; Plis, A.I.: Handbook on Splines for the User. CRC Press, Florida, USA. 1995.

[23] Sud, A.; Gayle, R.; Andersen, E.; Guy, S.; Lin, M.; Manocha, D.: Real-time Navigation of Independent Agents Using Adaptive Roadmaps. In: Proceedings of The ACM Virtual Reality Software and Technology, 5–7 November 2007, Newport Beach, California, USA. pp.99–106. (2007).

[24] TALON robots, http://en.wikipedia.org/wiki/Foster-Miller_TALON, Accessed Dec. 2014.

[25] Zhu, W.K.; Choi, S.H.: A Closed-loop Bid Adjustment Approach to Dynamic Task Allocation of Robots, Engineering Letters, 19(4), 2011, 279–288.