A Novel Path Planning Method Based on Extreme Learning Machine for Autonomous Underwater Vehicle

Diya Dong, Bo He*, Yang Liu, Rui Nian
School of Information Science and Engineering
Ocean University of China
Qingdao, China
dongdiya007@163.com; bhe@ouc.edu.cn;
oucliuyang@126.com; nianrui_80@163.com

Tianhong Yan
School of Mechatronic Engineering
China Jiliang University
Hangzhou, China
yanth@163.com

***Abstract*—Most existing path planning algorithms focus on generating piecewise linear functions, while the smoothness of the path is usually ignored. Route stability and energy saving are key issues for an autonomous underwater vehicle (AUV) working in the underwater environment. To ensure that AUV can pass through the target area steadily and safely, a path smoother is required. This paper puts forward a novel path planning method based on extreme learning machine (ELM), which generates a smooth and safe path at a quite fast speed. First a Voronoi preprocessor is employed to label the obstacles in the map and generate a rough path connecting the initial and the goal positions. Due to the property of nonlinear separating surface as well as fast learning speed, ELM is then applied to regenerate and smooth the path to ensure that the vehicle drives automatically and safely. Related experiments also validate the performance of our proposed method as expected. More analysis and some possible limitations are also discussed.**

Keywords—path planning; autonomous underwater vehicle; extreme learning machine; smooth

I. INTRODUCTION

In the field of marine robot, due to the superiority of autonomy, AUV shows great potential to execute various marine tasks including underwater surveys, oceanographic features tracking, hull inspection, etc [1]. To accomplish these missions, one key issue is how to make the vehicle drive reasonably and safely at work. This involves in some factors that have to be considered like obstacle avoidance [2], [3], route selection, vehicle maneuverability, planning efficiency, etc. Once a path planning decision is made, the lower level of control is then capable of manipulating or maintaining basic actions of vehicle such as depth, pitch, roll and heading.Hence adopting a rational path planning method is crucial for the ability of AUV to achieve the prescribed missions.

The path planning problem can be defined as: in the workspace of a vehicle, given a start point, a goal point and a map containing randomly distributed obstacles, an optimal collision-free path is to be found connecting the start and the goal position. A planned path is theoretically a curve but considering practical usage, it is represented by a set of waypoints. These waypoints give the vehicle a reference to track for the purpose of obstacle avoidance. Completing waypoints generation, the guidance agent is then able to send appropriate steering commands, thus realizing automatic driving.

Pervasive path planning approaches are mainly heuristic-based reactive methods and search-based algorithms (like A* search [4]).The paths generated by these algorithms have to be constructed in a manually simplified map (like grid cells) on which a lot of information is omitted. Furthermore, the possible orientations that the robot can choose are extremely limited and quite coarse especially for AUV. Besides it is time-consuming to search the map. Hence, without further process, this kind of path design might not be suitable for real practice in high demand of precision and safety. Another theory of path planning is inspired by potential field [5], and it is a series of approaches widely used in real-time obstacle avoidance, i.e. dynamic path planning requiring rapid computation. One big drawback is that without some special operations, this method usually suffers from local minima, which leads to global planning failure. And also, many biological planning algorithms like artificial neural networks and genetic algorithm (GA) are employed to solve AUV planning problems [6-8]. It should be noted that a common problem among these approaches is the discontinuity of the paths which cannot guarantee the safety of the vehicle during moving unless extra smooth measure involves. In addition, such a route also tends to be instable and energy-wasted. Recently, a new method was proposed by Jun Miura [9], i.e., support vector path planning, also referred to as SVM-PP. Totally different from the conventional methods, this method applies the classification technique to path planning, creating a brand new path planner which makes the route safe and smooth properly. Some modifications and applications in [10-12] also confirm the effectiveness of this model.

Motivated by SVM-PP, this paper proposes a path generator which adopts extreme learning machine to produce a smooth and safe path for AUV. Although pertaining to classification or regression models, ELM can cope with path planning problems fairly fast and accurate.

ELM is such a simple structure, and thus robust enough to maintain the reliability of the whole auto-driving system, as long as appropriate parameters are selected. For the same reason, ELM is capable of obtaining a good result in very short time. Hence compared with other learning algorithms, ELM

978-0-933957-43-5 ©2015 MTS

seems to be more qualified to serve autonomous vehicles requiring immediate responses in collaboration systems.

We presents the ELM-based path planning algorithm which is referred to as ELM-PP in this paper. First a basic ELM algorithm is briefly reviewed in Section II and then an introduction of Voronoi diagram is made in Section III. Merging Voronoi and ELM method, ELM-PP is described in detail in Section IV which is also the critical part of this paper. The results of the experiments are presented in Section V. Finally an evaluation and a conclusion are made in Section VI and Section VII respectively.

II. Basic Extreme Learning Machine

Derived from single hidden layer feedforward neural networks (SLFNs), ELM assigns the parameters of the hidden nodes randomly and no further tuning will proceed. All the parameters of hidden nodes are independent of each other. Therefore ELM is actually considered as generalized SLFNs [13].

Given a training set $\{(x_i, t_i) | x_i \in R^d, t_i \in R^m, i = 1, \cdots, N\}$, hidden node output function $G(a_i, b_i, x_j)$, and the number of hidden nodes L. Then SLFNs with $\tilde{N}$ hidden nodes can be expressed as:

$$f_{\tilde{N}}(x_j) = \sum_{i=1}^{\tilde{N}} \beta_i G(a_i, b_i, x_j) = t_j, j = 1, \cdots, N \quad (1)$$

(a_i, b_i) are parameters of hidden nodes. They can be but are not limited to Sigmoid and RBF functions, which can be respectively defined as follows:

$$\text{Sigmoid: } G(a_i, b_i, x) = g(a_i \cdot x + b_i) \quad (2)$$

$$\text{RBF: } G(a_i, b_i, x) = g(b_i \|x - a_i\|) \quad (3)$$

And β_i in (1) is the weight vector connecting the i th hidden node and the output node. Then (1) can be simplified as

$$T = H\beta \quad (4)$$

where

$$H(a_1, \cdots, a_N, b_1, \cdots, b_{\tilde{N}}, x_1, \cdots, x_N) = \begin{bmatrix} G(a_1, b_1, x_1) & \cdots & G(a_{\tilde{N}}, b_{\tilde{N}}, x_1) \\ \vdots & \ddots & \vdots \\ G(a_1, b_1, x_N) & \cdots & G(a_{\tilde{N}}, b_{\tilde{N}}, x_N) \end{bmatrix}_{N \times \tilde{N}}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\tilde{N}}^T \end{bmatrix}_{\tilde{N} \times m} \quad \text{and} \quad T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m}$$

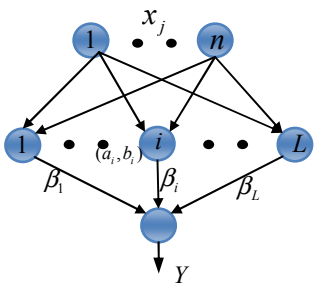


Fig. 1. Structure of basic ELM.

H is called the hidden layer output matrix of the neural network, and the i th column of H is the output of the i th hidden node with respect to inputs $x_1, x_2, \cdots$. Once the training samples are available and the random parameters of hidden nodes are assigned, H can be calculated soon. Therefore the output weights β can be inferred:

$$\beta = H^{\dagger} T \quad (5)$$

Where $H^{\dagger}$ is the Moore-Penrose generalized inverse of hidden layer output matrix H.

Fig. 1 illustrates the structure of ELM algorithm. To be specific, a three-step description of ELM model is given below [15]:

Input: training set $\{(x_i, t_i) | x_i \in R^d, t_i \in R^m, i = 1, \cdots, N\}$, hidden node output function $G(a, b, x)$, and the number of hidden nodes L.

a) Randomly assignhidden node parameters (a_j, b_j) where $j = 1, \cdots, L$.

b) Calculate the hidden layer output matrix H.

c) Calculate the output weights $\beta = H^{\dagger} T$.

The reason for using ELM is that the learning speed of ELM is extremely fast compared with artificial neural networks and support vector machines due to dispensing with tuning parameters [14], [19]. ELM tends to reach the solutions straightforwardly without some trivial issues like local minima, overfitting or improper learning rate appearing in gradient-based learning algorithms.

III. Voronoi Preprocessor

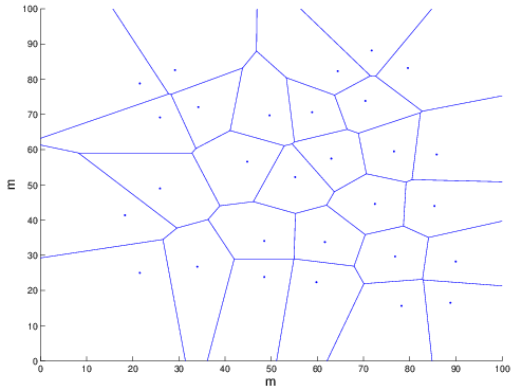
To label the obstacles in the map, one preliminary step needs to be manipulated first, i.e., the Voronoi tessellation [17] which can be expressed as:

$$\Gamma = \{z | z \in C_{free}, \exists x_i, x_j \in C_{obst}, x_i \neq x_j, s(x_i, z) = s(x_j, z)\} \quad (6)$$

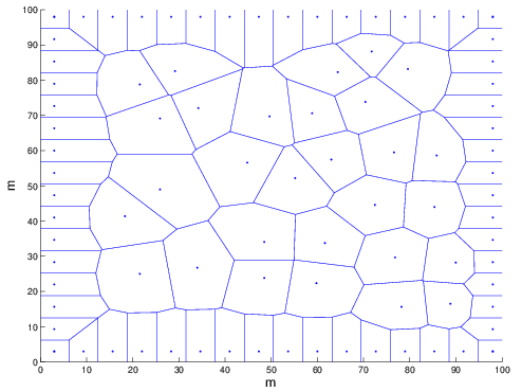
where

$$s(x, z) = \min_{x \in C_{obst}} \text{distance}(x, z) \quad (7)$$

And z is the set of points that constitutes convex polygons so that each polygon contains one obstacle point x, and any point on the polygon is closer to its obstacle (x_i or x_j) than to



(a)



(b)

Fig. 2. Voronoi diagrams with (a) and without POOF's (b).

any others.

A. Pseudo-obstacles for Outer Frame

One necessary operation that needs to be done before utilizing Voronoi tessellation is to set pseudo-obstacles for outer frame (POOF), as is depicted in Fig. 2. This operation is in fact an auxiliary to make Voronoi diagram better satisfy the real situation of path selection. Basically, Voronoi preprocessor can provide a rough and temporary path and be used to separate the obstacles into two classes. There are many choices for us to pick up a path connecting the start and the goal point in Voronoi diagram [16]. Therefore, a reasonable or short path needs to be manually selected depending on actual requirement of the missions. After divided by the selected path into two districts, the map is composed of positive and negative parts where the obstacles are labeled with the corresponding classes. In this way, the planar path planning problem is transformed into a binary classification problem.

B. Pseudo Obstacles for the Start and the Goal Points

Another issue that should be noted is that the path generated by classifier is still hard to precisely pass through the start and goal points. Usually there exists some bias between the path and the desired position. Then another pseudo obstacle trick is employed. Concretely, these virtual point-configured obstacles are in parallel to the predicted path. And equivalent

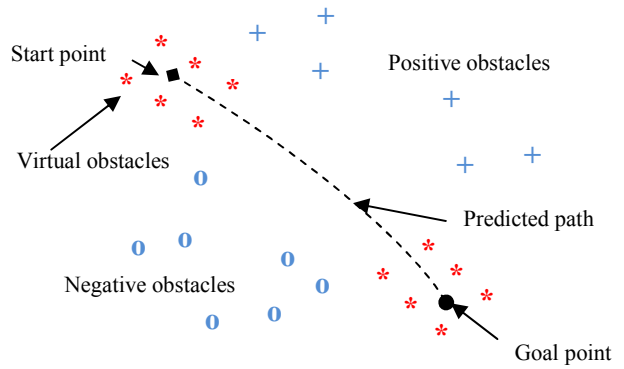


Fig. 3. Virtual obstacles (red stars) located around the start and the goal points.

positive and negative obstacles are set around the start and the goal points. Empirically three virtual obstacles on each side for one point are appropriate. As shown in Fig. 3, these virtual obstacles are set in order to give constraints of the decision boundary in classification process such that the path can reach the start and the goal points.

However, this measure can only decrease the deviation of the desired path rather than eliminate it. Hence, it should be avoided to view the start and the goal positions as accurate data to put into computation, but as virtual references.

C. Summary of Preprocessing Procedures

The preprocessing operation can be summarized as 4 steps:

a) Step 1: map preprocessing. First extract the vertices of all the obstacle polygons as the training samples.

b) Step 2: set the start and the goal points and divide the domain with the line connecting the start and the goal points in Voronoi diagram.

c) Step 3: set the obstacles on one side as the positive samples and the one on the other side as negative samples.

d) Step 4: set pseudo obstacles around the start and the goal points.

IV. ELM-BASED PATH PLANNING

A. Decision Function

For safety and efficiency, a continuous and smooth approximated path, rather than a piecewise linear path, is preferred. After the workspace of AUV is modeled by Voronoi preprocessor, ELM method follows. As preceding section mentioned, ELM calculates the output weights $\beta = H^{\dagger}T$. According to ELM theory, a real number is assigned to β and the result is then calculated as

$$Y = H\beta \tag{8}$$

For binary classification problem, the model judges a certain sample as the positive or negative class according to which corresponding output node receives the larger Y value. In this case, the decision function can be decided:

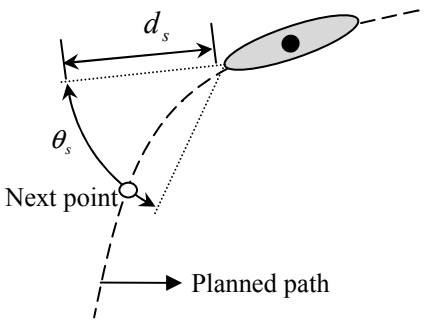


Fig. 4. Searching for the next point.

$$H\beta = 0 \tag{9}$$

This is not only the decision boundary used to divide the obstacles, but also the final path learned from the ELM model. Compared with the rough path in Voronoi diagram, ELM actually acted as a path smoother, which fits a curve of the piecewise linear path produced by the preprocessor.

Different from ordinary classification model, the prediction step is not needed in ELM-PP model. In other words, the model does not predict the unknown samples any more, but end up finishing the separating line learning.

B. Randomness of ELM Parameters

For randomness of the parameters of ELM, the routes are not exactly identical in repetitive experiments. This is also the difficulty of this method to be coped with. Original Voronoi path therefore is considered to be used as a reference to measure the error. Because Voronoi path is actually comprised of a series of perpendicular bisectors through pairs of adjacent obstacles, it is the maximum safe distance between the path and the surrounding obstacles. We can compare whether the paths produced by ELM-PP generator fits the Voronoi path enough. And then the best fitting one can be seen as the safest path.

To build a criterion to measure the performance of the planned path, the squared error method is adopted. Suppose the Voronoi path is configured by m points $\{(x_i, y_i)|i = 1,...,m\}$, and the corresponding ELM waypoints calculated by (8) are represented by $\{(x_i, \hat{y}_i)|i = 1,...,m\}$. The squared error function is:

$$\varepsilon = \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 \tag{10}$$

After n repeated experiments have been implemented, we can also obtain n ε values, and then the path P_{j^*} with the lowest ε is selected as the final path by the discriminant:

$$P_{j^*} = \arg\min_{j\in\{1,2,...,n\}} \{\varepsilon_j\} \tag{11}$$

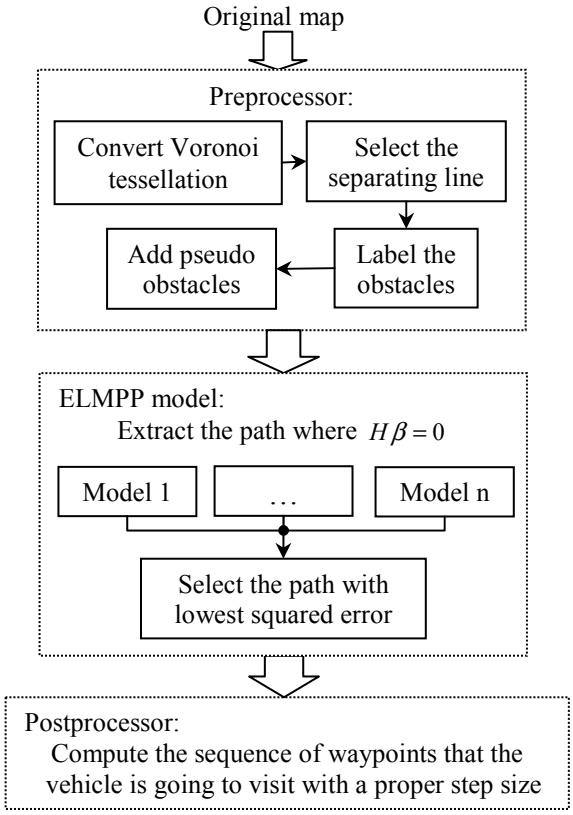


Fig. 5. Flow chart of ELM-based path planning.

C. Discrete Sampling

So far, a path has been planned out, yet it cannot be put into application directly. An available path to the vehicle is constituted of a sequence of waypoints. In conventional path planning approaches, it is often a piecewise linear path represented as a set of waypoints and thus can be used by the system immediately. Unlike this pattern, ELM-PP transforms the decision boundary into a waypoint set at a certain step size depending on the maneuvering characteristics of AUV. Taking advantage of such a continuous path curve, our proposed method not only can give the waypoints as desired interval, but also can provide the vehicle with waypoints whenever needed to refer during dynamic planning.

As shown in Fig. 4, we determine the separating line by repeatedly searching for the next point where value Y in (8) is zero with a fixed step d_s or time interval. Finally, we obtain successive waypoints for the vehicle to visit.

Now the whole ELM-PP algorithm can be summarized as the flow chart in Fig. 5 and the chart can be explained by 4 phrases:

a) Building the map of workspace, in which the block-obstacles are decomposed into a set of points to enable the ELM classifier to work.

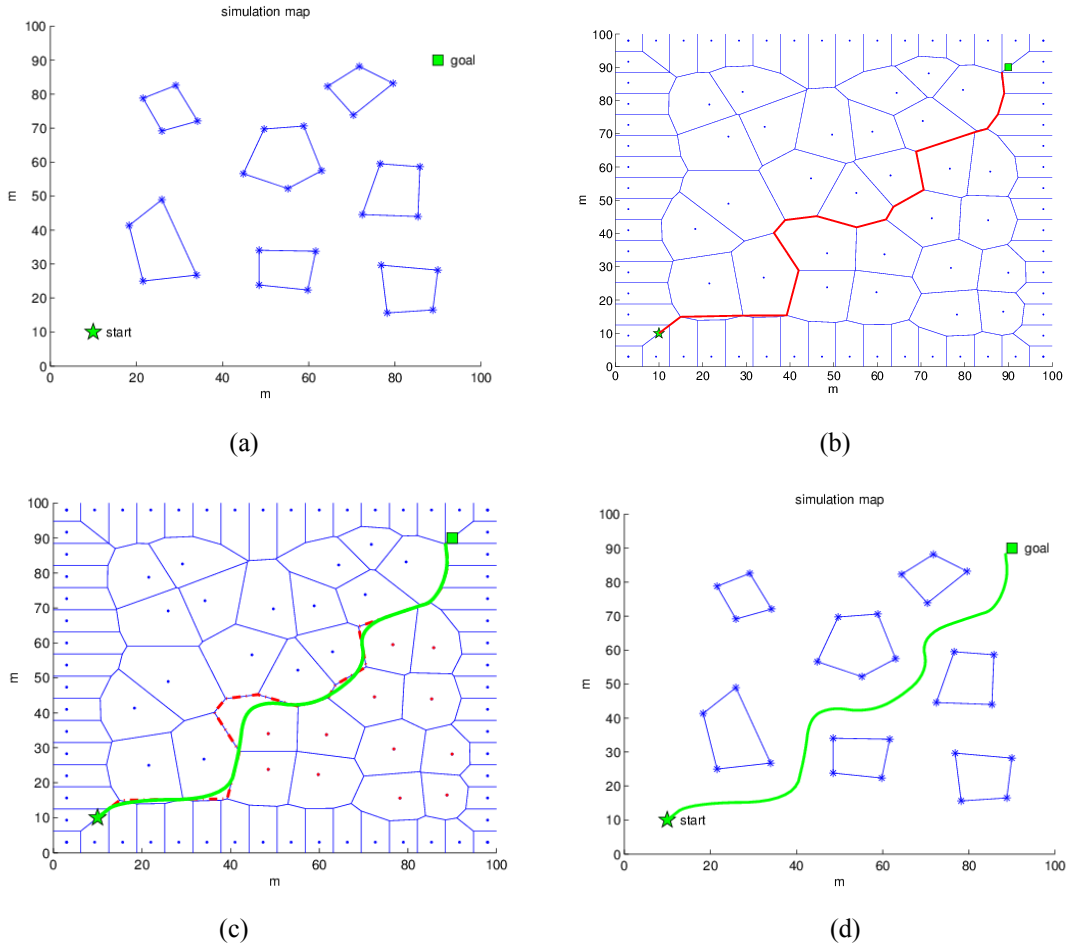


Fig. 6. Experimental results of ELM-based path planning. (a) is the original map, (b) is the rough path (red line) selected in Voronoi diagram, (c) is the smoothed path (green line) processed by ELM and (d) displays the final path as well as the original map.

b) Using Voronoi tessellation to roughly generate a desired path. Meanwhile, label the obstacles as the positive and negative samples according to the separation by the path.

c) Training ELM with these samples and extracting that discriminant plane (a line in 2D case) which is exactly the path that the vehicle is going to track.

d) As mentioned previously, the path curve can be expressed by a sequence of waypoints to give reference for AUV guidance according to the rate, speed and orientation of the vehicle.

V. EXPERIMENTS

Experiments were carried out to verify the proposed method. In an attempt to find out the best parameter combination, various activation functions (including sigmoid, radial basis function, polynomial, etc.) and different number of hidden nodes were tried. Finally number of hidden nodes is set to 20 and sigmoid function is considered suitable for our path planning problem. All experiments were conducted on Matlab R2012b platform. The experiments have repeated 50 times due to the randomness of the parameters of ELM. Fig. 6(a)-(d) illustrate the main procedures of ELM-PP algorithm. For the sake of simplicity, some operations like pseudo obstacles, squared error computation and waypoints generation are omitted in the figure. Fig. 6(c) and (d) straightforwardly present the best path which has the lowest squared error. For detailed conceptual elaborations please refer to previous sections.

TABLE I. COMPARISONS OF SVM-PP AND ELM-PP

	Average elapsed time	*Number of hidden nodes*	*Activation function*	*Squared error*
SVM-PP	0.68s	-	RBF	14.49
ELM-PP	0.12s	20	RBF	15.32

Comparisons of SVM-PP and ELM-PP were also made, which is performed on the map in Fig. 6. After 50 repeated experiments for both methods, according to the results shown in Table I, the squared error of SVM-PP is a little lower than that of ELM-PP, while the elapsed time of the latter is significantly less than that of the former. Therefore, compared with SVM-PP, ELM-PP is much faster than SVM-PP and receives a comparable result as well, which indicates that the proposed method seems more suitable for underwater environment requiring real-time running.

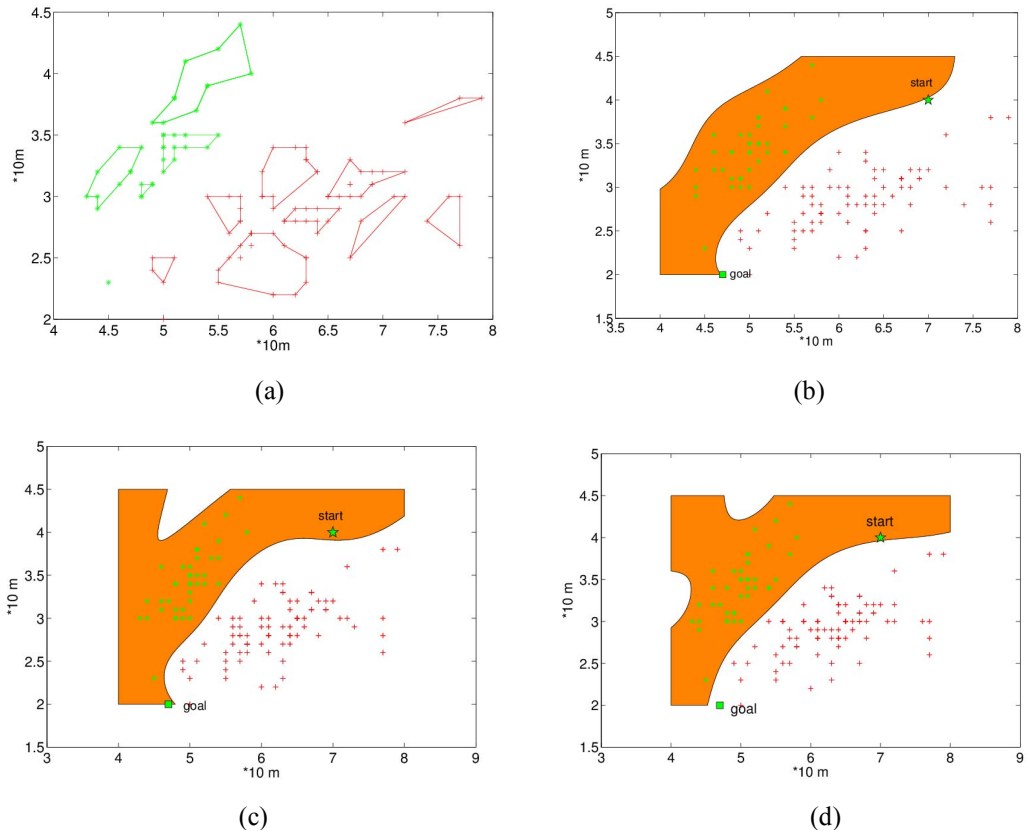


Fig. 7. Repeated tests for ELM-PP. (a) is the original map, and (b) to (d) are three of the results. The squared errors are respectively 5.06 for (b), 7.85 for (c) and 10.66 for (d).

The results of Fig. 6 and Table I convincingly confirm that ELM-PP can smooth the path in a short time.

Another set of point-obstacles were also tested in the same way. This test indicates the influences of the random parameters of ELM. Fig. 7(b) to Fig. 7(d) are the contour maps where the contours actually represent the possible paths that ELM have planned without considering the start and the goal positions. The line that traverses the start and the goal points is adopted as the final path. Although inconsistent paths are planned each time, they still succeed in obstacle avoidance. The test repeated for 50 times and 92% achieved successful results. Compared with the other two paths, Fig. 7(b) gets the relatively lower squared error and turns out to be the optimal safety path which also can be observed in the figures.

VI. EVALUATIONS

A. Current Problems

The experiments above indeed validate the efficiency of our proposed method. However, there still exist some problems that should be noted. First of all, due to randomness of the parameters in ELM method, the paths keep changing constantly in repeated tests. Therefore without an extra recorder, previous path which is perhaps the best one might be lost forever. Second, for the same reason, ELM-PP is still an unstable algorithm so far, which might result in planning failure. Due to the instability of ELM-PP, it is less secure than SVM-PP or popular path planning methods and still not a feasible approach to put into application yet. Finally, the improper manual classification of obstacles is also a fatal factor that causes a collision-free path will never be found out.

B. Future Recommendations

Although currently it is inappropriate to put ELM-PP into practical application as an independent agent, our proposed ELM-PP is helpful to act as an optimizer or combine with other methodologies. For example, the piecewise linear path generated by search-based path planning methods can be further smoothed by ELM-PP, thus providing with a safer path for AUV control. Moreover, the online sequential ELM (OS-ELM) [18] can be also considered to be used in dynamic path planning in an environment without a priori map.

In addition, our proposed planner can be only applied in the environments containing no threatening obstacles. Further research still needs to be done on how to settle threatening obstacles underwater.

VII. CONCLUSIONS

This paper introduces a novel path planning method based on ELM model. The proposed planner is a combination of Voronoi and ELM algorithm. Concretely, Voronoi diagram is adopted as an auxiliary to build a classification model for ELM. Ultimately the decision function of ELM is extracted as the path that AUV system needs to track. Simulated tests prove that our proposed method works well on smoothing the planned path so as to ensure safety. Taking advantage of the simple structure of ELM, the path can be generated very

shortly. This property can effectively facilitate energy saving and driving automatically and safely for AUV in underwater environment.

ACKNOWLEDGMENT

This paper is partially supported by the Natural Science Foundation of China (41176076, 51075377, and 51379198), the High Technology Research and Development Program of China (2006AA09Z231, 2014AA093410).

REFERENCES

[1] M. L. Seto, "Path Planning for Automomous Underwater Vehicles," in *Marine Robot Autonomy*. New York: Springer-Verlag, 2013, pp. 177-224.

[2] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *1985 IEEE Int. Conf. Robotics and Automation*, St. Louis, March 25-28, 1985, pp. 500-505.

[3] L. R. Fodrea and A. J. Healey, "Obstacle Avoidance Considerations for the REMUS Autonomous Underwater Vehicle," *ASME 2003 22nd Int. Conf. Offshore Mechanics and Arctic Eng.*, vol. 3, Cancun, Mexico:, 2003, pp. 537-543.

[4] B. Garau, A. Alvarez and G. Oliver, "Path planning of Autonomous Underwater Vehicles in current fields with complex spatial variability: an A* approach," *Proc. 2005 IEEE Int. Conf. Robotics and Automation*, Barcelona, Spain, 2005, pp. 194-198.

[5] Y. Wang and N. Ahuja, "A potential field approach to path planning," *IEEE Trans. Robot. Autom.,* vol. 8, pp. 23-32, Feb 1992.

[6] C. Cheng, K. Fallahi, H. Leung and C. Tse, "An AUVs path planner using genetic algorithms with a deterministic crossover operator," *IEEE Int. Conf. Robotics and Automation*, Anchorage, Alaska, May 2010, pp. 2995–3000.

[7] H. Li, S. Yang and M. Seto, "Neural network based path planning for a multi-robot system with moving obstacles," *IEEE Trans. Syst. Man and Cybern. C, Appl. Rev.* vol. 39, pp. 410-419, May 2009.

[8] Y. V. Pehlivanoglu, "Path planning for autonomous UAV via vibrational genetic algorithm," *Aircraft Eng. Aerospace Tech.*: *Int. J.,* vol. 79, no. 8, pp. 532–539, 2007.

[9] J. Miura, "Support Vector Path Planning," *Proc. 2006 IEEE/RSJ. Int. Conf. Intelligent Robots and Syst.*, Beijing, China, Oct 9-15, 2006, pp. 2894-2899.

[10] K. Su, F. Lian and C. Yang, "Navigation design with SVM path planning and fuzzy-based path tracking for wheeled agent," *Fuzzy Theory and it's Applicat. (iFUZZY), 2012 Int. Conf.*, Taichung, Taiwan, 16-18 Nov 2012, pp. 273-278.

[11] C. Yang, J. Yang and F. Lian, "Safe and smooth: mobile agent trajectory smoothing by SVM," *Int. J. Innovative Computing Inform. and Contro*l, vol. 8, no. 7B, pp. 4959–4978, 2012.

[12] C. Xia, W. Nan and D. Yong, "The Semi-Supervised Support Vector Machine of Path Planning," *Measuring Technology and Mechatronics Automation (ICMTMA), 2013 5th Int. Conf.*, H. K., China, 16-17 Jan 2013, pp. 1230-1232.

[13] G. Huang, "Classification ability of single hidden layer feedforward neural networks," *IEEE Trans. Neural Netw.*, vol. 11, pp. 799-801, May 2000.

[14] G. Huang, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, pp. 489-501, 2006.

[15] G. Huang, Q. Zhu, and C. Siew, "Extreme Learning Machine: A New Learning Scheme of Feed forward Neural Networks," *Proc. Int. Joint Conf. Neural Networks*, 25-29 July, 2004, Budapest, Hungary, 2004, pp. 985-990.

[16] H. Choset, et al., *Principles of robot motion: theory, algorithms, and implementation*. Cambridge, MA: MIT Press, 2005.

[17] D. G. Kirkpatrick, "Efficient computation of continuous skeletons," *Proc. 20th IEEE Annu. Symp. Found. Comput. Sci.*, San Juan, Puerto Rico, 1979, pp. 18-27.

[18] G. Huang, N. Liang and H. Bong, "On-Line Sequential Extreme Learning Machine," *IASTED Int. Conf. Computational Intell.*, Calgary, Canada, July 4-6, 2005.

[19] G. Huang, "An Insight into Extreme Learning Machines: Random Neurons, Random Features and Kernels," *Cognitive Computation*, vol. 6, no. 3, pp. 376–390, 2014.