A Real-time Local Path Planning Method Based on SVM for UGV

Chengchen Zhuge
School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China

Zhenmin Tang*
School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China
**Corresponding author(E-mail: tzm.cs@mail.njust.edu.cn)*

Abstract
Path planning is one of essentials of unmanned ground vehicle (UGV). For the case of poor lighting and weather, traditional vision based methods can not extract effective route boundaries to generate reasonable path stably in unstructured road. By taking advantage of distance-sensing technology (e.g. 64-beam LiDAR), this paper proposes an efficient real-time path planning approach. In this approach, given grid map from 64-beam LiDAR, obstacles on both sides of the road are regarded as two classes fed to Support Vector Machine (SVM) to generate an initial safe path. During driving, a time weight based least square fitting is adopted to refine path from multiple safe paths which will be described by quartic polynomial, providing stable driving route. Combined with UGV's state, controls points from the refined path are adopted to generate the final path through Bezier curve fitting. Experiments on real UGV under different road scenario are conducted, showing that the proposed method can obtain stable and reasonable path with promising performance.

Key words: Real-time, Path planning, UGV, LiDAR, SVM, Least square.

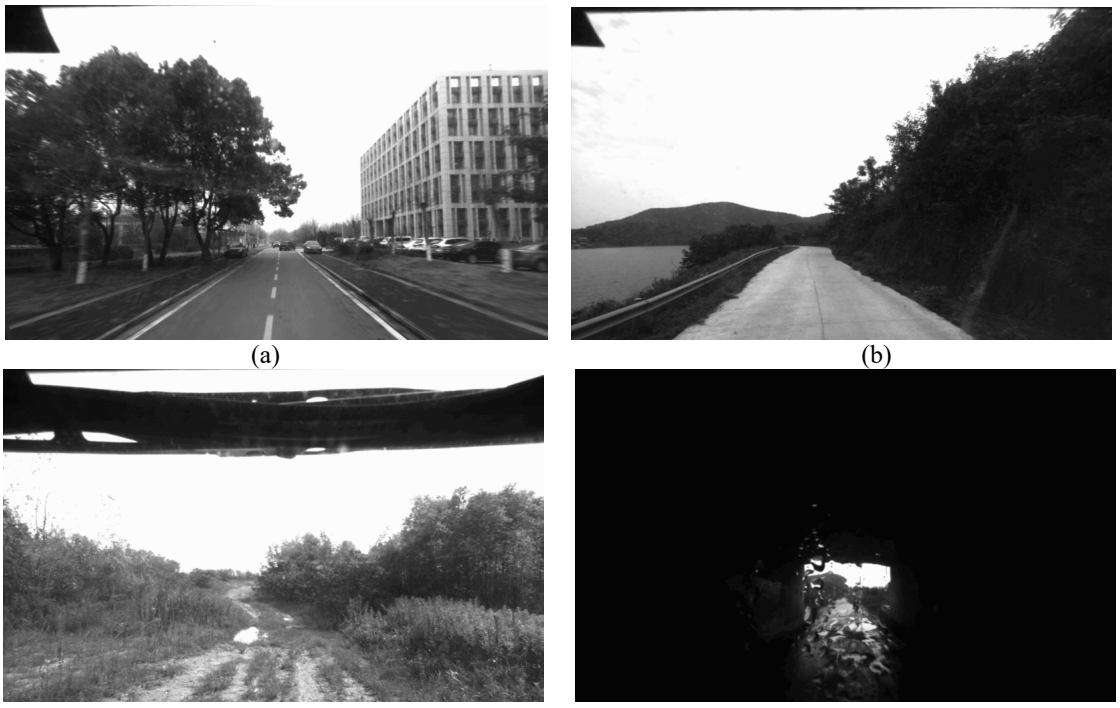
1. INTRODUCTION

In recent years, Unmanned Ground Vehicle (UGV) has become a hot research topic in the field of artificial intelligence (Fassbender, Mueller and Wuensche, 2014; Wei et al., 2014). UGV system is usually composed of perception module, fusion module, path planning module and control module. UGV relies on equipped sensors (e.g. camera, LiDAR, inertial navigation system) to perceive and model environment. Such tasks are achieved through road detection, obstacle detection, target recognition and so on. As one of the essential parts, path planning module is to generate a safe path effectively and efficiently given current environmental information, tasks and state of the UGV.

If road boundaries can be detected effectively, path planning module can generate a trajectory along the road and guide UGV to drive safely. According to curvature of the road ahead, the upfront steer and speed control can ensure the UGV pass the curve road safely and smoothly. At present, most of the road detection methods are vision based (Amini and Karasfi, 2016; Maarir and Bouikhalene, 2016; Wang, Sun and Zhao, 2015; Zu et al., 2015), which can obtain excellent results in structured road (see Fig. 1(a)) and semi structured road (see Fig. 1(b)). However, vision based methods cannot works stably in unstructured environment because of the lack of lane, fuzzy boundary, and uneven surface materials (see Fig. 1(c)). Sometimes, the poor lighting and weather also influence the performance of the vision based methods (see Fig. 1(d)).

On the contrary, grid map generated by 64-beam LiDAR can show road structure information effectively (see Fig. 2). Many typical approach such as Voronoi graph (Canny, 1985; Takahashi and Schilling, 1989) or potential field (Khatib, 1986; Koren and Borenstein, 1991) can be adopted on the grid map to generate a cost map which implies the path information, then a search method is adopted to find the safe path. Yu and Qiu (2013) generates Voronoi graph from grid map as road skeleton, then uses Fast Marching Method (FMM) to search the path. However, the generated path is neither safe nor smooth. The Fast Marching square Method (FM^2) (Gómez, Mavridis and Garrido, 2014) is then proposed to compute safe and smooth trajectories effectively. FM^2 firstly applies the FMM to generate a velocity map, then uses the FMM on the velocity map to search a safe and smooth path to the goal point. However, the FM^2 needs a lot of computing resources. Linear Discriminant Analysis (LDA) is adopted to classify the obstacles on both sides of the road by Liu, Tang and Ren (2011). Road boundaries are then fitted which can be used for path planning in real-time. However, the LDA mentioned cannot deal with the road with large curvature (see Fig. 2).

To effectively address above shortcomings, in this paper, a local path planning method based on non-linear SVM and least square fitting is proposed. This method can generate a safe, smooth, and executable path for UGV in real-time. It can effectively deal with the problem of path planning in unstructured road environment with grid map. Non-linear SVM ensures accurate road boundaries detection and least square fitting provides smooth and consistent path generation. The validity and correctness of the proposed method is verified on a real vehicle.



(a) (b)

(c) (d)

Figure 1. Different road scenarios. (a) Structured road (b) Semi structured road (c) Unstructured road (d) Rain and in the tunnel

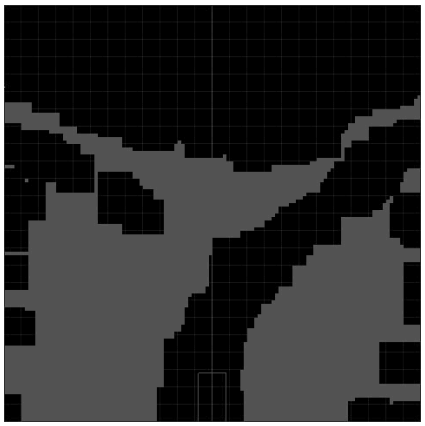


Figure 2. Local grid map

The overall organization of this paper is as follows: In Section 2, path extraction based on non-linear SVM is introduced. Section 3 mainly focuses on the whole local path planning method. In Section 4, experiments and analysis in real-world case is presented. Conclusions are drawn in Section 5.

2. TRAJECTORY GENERATED BASED ON NON-LINEAR SVM

SVM (Bron et al., 2015; Gómez, Mavridis, and Garrido, 2014) is one of the most commonly used classifier learning methods. The obtained optimal classification hyperplane can maximize the class margin. In Fig. 3, obstacles on both sides of the road can be regarded as two kinds of targets, which can be served as samples fed to SVM. SVM can provide a hyperplane with a maximal distance to obstacles. The projected trajectory of the classified hyperplane on the 2D grid map is not only smooth but also far away from the obstacles, so it can be tracked by the UGV.

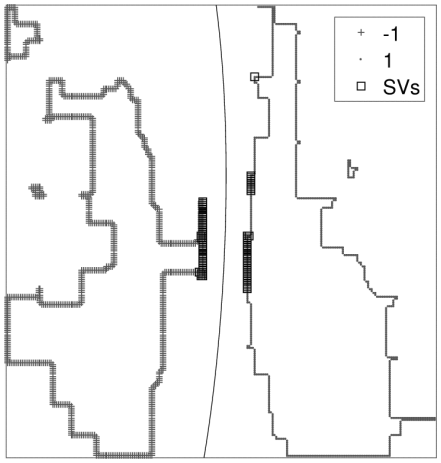


Figure 3. SVM classification hyperplane schematic diagram

Let (p_j, γ_j) be obstacle sample, where $p_j = (x_j, y_j)$ represents coordinate of the j^{th} obstacle under the vehicle-centered coordinate system and the $\gamma_j \in \{-1, +1\}$ represents class label. The discrimination function is given by

$$f(p_j) = w^T p_j + b = \sum_{i=1}^{n} \alpha_i \gamma_i p_i^{\mathrm{T}} p_j + b \tag{1}$$

where n represents the number of support vectors, α_i represents Lagrange multiplier.

Note that linear classifier cannot deal with road with large curvature. SVM is adopted to determine non-linear separating hyperplane with kernel tricks. In this paper, Radial Basis Function (RBF) kernel is utilized for training and testing due to its excellent performance than other kernels in our experiments. RBF kernel $K(\cdot,\cdot)$ is defined as

$$K(p_i, p_j) = \exp\left(-\frac{\| p_i - p_j \|^2}{\delta^2}\right) \tag{2}$$

where δ is kernel parameter.

Then the training process is to solve the dual problem as follows:

$$\begin{gathered} \max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j \gamma_i \gamma_j K(p_i, p_j) \\ s.t., \alpha_i \geq 0, i = 1, \cdots, n \\ \sum_{i=1}^{n} \alpha_i \gamma_i = 0 \end{gathered} \tag{3}$$

Finally, the hyperplane equation is obtained:

$$f(p_j) = \sum_{i=1}^{n} \alpha_i \gamma_i K(p_i, p_j) + b = 0 \tag{4}$$

w and b, solutions of the primal problem, can also be obtained.

The points on the separating function Eq. 4 can be treated as a safe path since it can maximally separate obstacles on both sides of the road. Then points are sampled with equal-step from the hyperplane trajectory. As shown in Fig. 3, the SVM classification hyperplane trajectory can effectively separate obstacles on both sides of the road. Due to the property that the obstacles corresponding to the support vectors are the nearest points to the trajectory (safe path) and they are distributed on the narrowest parts of the road, these obstacles can be used to estimate the traversability of the road. The distance from support vector to the hyperplane can be calculated as follows:

$$D(p_i) = \frac{|w \cdot p_i + b|}{\|w\|} \tag{5}$$

The road is not safe if there is a support vector p_i such that:

$$D(p_i) < w_{\text{car}} / 2 + d_{\text{safe}} \tag{6}$$

where w_{car} represents car width, d_{safe} is a safety parameter which can be determined experimentally.

3. REAL-TIME LOCAL PATH PLANNING METHOD

When UGV is moving, the local grid map will be updated at a fixed frequency which is decided by the working frequency of the 64-beam LiDAR. A new trajectory (i.e. hyperplane) is obtained in each update. Due to uncertainty of environment (e.g. new obstacles, measurement errors caused by vehicle turbulence), the trajectories obtained are not consistent all the time.

Fig. 4 shows the trajectories in multiple updates. If no otherwise specified, the unit of the values in the figures is cm. Note that these trajectories are projected within the same coordinate system. It is clear to see that these trajectories are not coincident. In this paper, least square fitting (Goldhammer et al., 2015; Matchen and Nadler, 2014; Wilson et al., 2015) is used to estimate the path model from multiple trajectories. The whole process to generate the final safe local path is illustrated in Fig. 5. There are mainly three phases: 1) initial safe path generation; 2) path refinement by least square fitting; 3) final path generation.

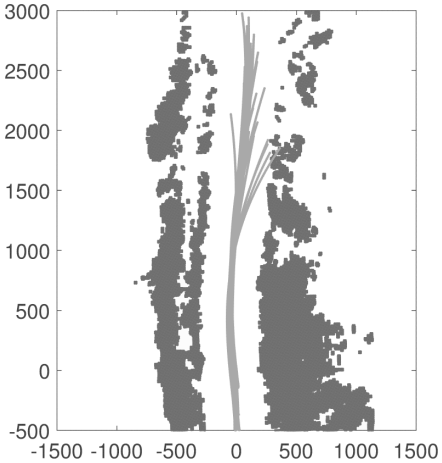


Figure 4. Multi-frame projected data schematic diagram

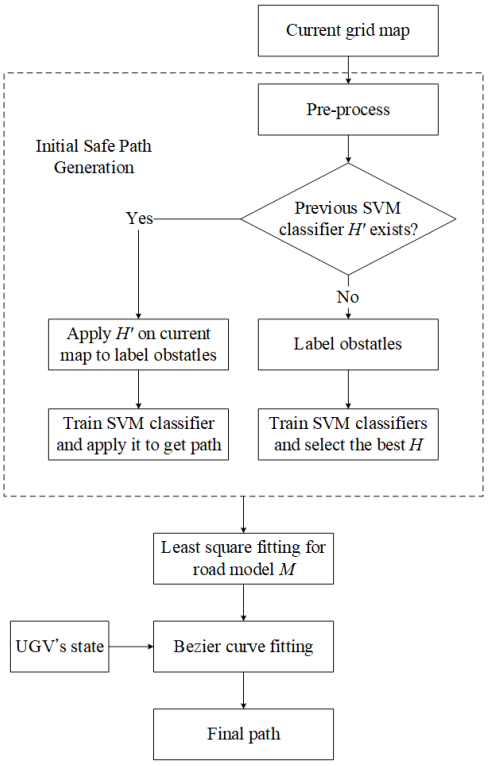


Figure 5. Flow diagram of the local path planning algorithm

3.1. Initial Safe Path Generation

In this paper, the size of local grid map is 30m×30m, and the grid resolution is 12.5cm×12.5cm. The original grid map is shown in Fig. 6(a). The original grid data is not suitable to be fed to SVM directly. It needs to be pre-processed and labeled. The specific process is described as follows.

The original grid map is treated as a binary image and processed with dilation and erosion (see Fig. 6(b)). The window size is 17×17 (pixels) which is approximately equal to the car width in real-world. Then chain code tracking is applied to get the obstacles' contour information (Singh et al., 2015). Each closed contour is labeled as an obstacle target (see Fig. 6(c)).

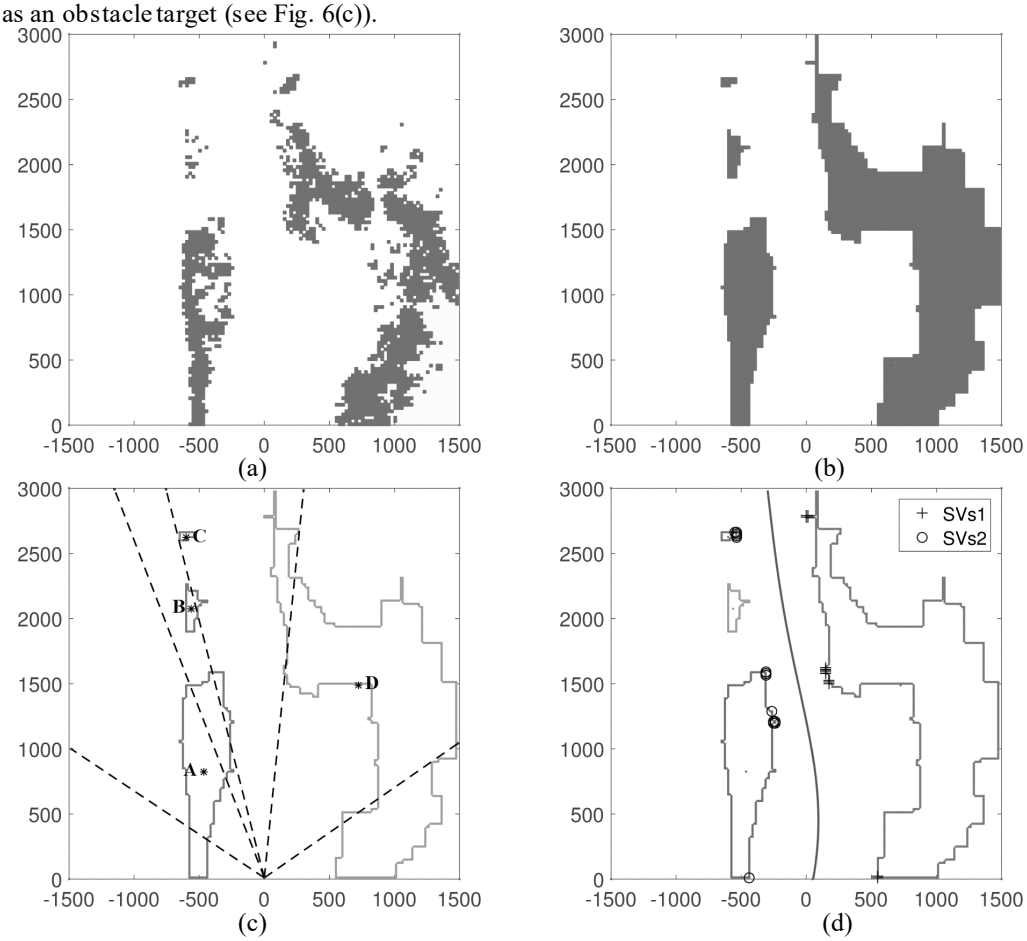


Figure 6. Data pre-processing and classification. (a) Original grid map (b) Dilation and erosion (c) Extract and separate the obstacles (d)The best result

Let $p_{ij} = (x_{ij}, y_{ij})$ be the coordinate of the j^{th} contour point of the i^{th} obstacle, $p_{ci} = (x_{ci}, y_{ci})$ be the barycentric coordinate of the i^{th} obstacle, and (p_{ij}, γ_i) is the training sample used in this paper. The N_{obs} obstacles will be transformed into the polar coordinate and be separated into two groups.

If there is no SVM classifier from previous map, obstacles need to be labeled for training by SVM. The obstacles on the left of the separating line will be labeled with $\gamma_i = -1$ and the obstacles on the right of the separating line will be labeled with $\gamma_i = +1$. For example, there are four obstacles labeled as region A, B, C and D in Fig. 6(c), and there are five separating results corresponded to $\varnothing$ vs. $\mathrm{A} \cup \mathrm{B} \cup \mathrm{C} \cup \mathrm{D}$, A vs. $\mathrm{B} \cup \mathrm{C} \cup \mathrm{D}$, $\mathrm{A} \cup \mathrm{B}$ vs. $\mathrm{C} \cup \mathrm{D}$, $\mathrm{A} \cup \mathrm{B} \cup \mathrm{C}$ vs. D, $\mathrm{A} \cup \mathrm{B} \cup \mathrm{C} \cup \mathrm{D}$ vs. $\varnothing$ When the group is $\varnothing$, points on the boundary of the local map are chosen as the training samples. Then $N_{\text{obs}} + 1$ groups of the training sample (p_{ij}, γ_i) will be trained to obtain $N_{\text{obs}} + 1$ sets of (w, b) and the corresponded hyperplanes. Finally, the best hyperplane H_{best} with the maximum distance to the two categories of obstacles is saved. As shown in Fig. 6d, the projected trajectory of the hyperplane H_{best} in grid map is a safe and smooth path for the UGV.

When UGV is moving, the extended trend of the road can be considered to be changing slowly. Therefore, when a new frame of grid map arrives, the obstacles in the new frame can be classified by the SVM hyperplane H' of the previous frame. In this case, the classified obstacles can be treated as new training samples to obtain a new SVM hyperplane H as the new frame. In contrast, without SVM hyperplane H' from previous frame, multiple training groups are generated to train multiple SVM hyperplanes. The detail process is described as follows: Firstly, all the $p_{ij} = (x_{ij}, y_{ij})$ of N_{obs} obstacles in current frame are transformed into the vehicle-centered coordinate system of previous frame, denoted by $p_{ij}' = (x_{ij}', y_{ij}')$. Then, these $p_{ij}' = (x_{ij}', y_{ij}')$ are classified by the SVM hyperplane H' of previous frame and be labeled by $\gamma_i \in \{-1, +1\}$. It is assumed that the extended trend of the road changes slowly, the label γ_i of the same obstacle in the adjacent frames should be the same, although the coordinates are different in the adjacent frames. Finally, the new training samples (p_{ij}, γ_i) are trained to obtain the new SVM hyperplane H.

3.2. Path Refinement

The road model is approximated with a quartic polynomial as follows:

$$f(y) = a_0 + a_1 \cdot y + a_2 \cdot y^2 + a_3 \cdot y^3 + a_4 \cdot y^4 \tag{7}$$

where $f(y)$ is the road model to be estimated and a_i is the model parameter. The positive y-axis is the heading of the UGV in the vehicle-centered coordinate system. In this paper, a time weight based least square fitting is adopted on the multi-frame projected data (see Fig. 4) to estimate the road model. The least square problem can be defined as follows:

$$\min_{a_i} \Phi(a_0, a_1, a_2, a_3, a_4) = \min_{a_i} \sum_{j=1}^{N} \omega_j (f(y_j) - \sum_{i=0}^{4} a_i y_j^i)^2 \tag{8}$$

where N is the number of the fitting points and $\omega_j \in (0, 1]$ is the time weight. Optimal solution for the parameter can be obtained with gradient when the objective function becomes zero (see Eq.9).

$$\frac{\partial \Phi}{\partial a_i} = 0, \quad i = 0, \ldots, 4 \tag{9}$$

The pseudo code for the time weight based least square fitting is shown in Algorithm. 1. At time t, the hyperplane trajectory H_t is sampled into discrete points which are stored in P_t. Then P_t is assigned with an initial time weight $\omega_t = 1$ (see Line 2-3 in Algorithm. 1) which will reduced by $1/T$ each time. T decides how long the time queue will be maintained. It means that the new hyperplane trajectory has greater effect than the old ones in the least square fitting procedure. The time weight of the historical projected points, which are already stored in *global_points*, is updated before the historical projected points merge with the P_t (see Line 4 in Algorithm. 1).

After the historical projected points in the global coordinate system are transformed into the local vehicle-centered coordinate system and be saved in *local_points*, P_t is also added to *local_points* (see Line 5 in Algorithm. 1). The local points which have positive y value and positive time weight are retained in the *global_points* and severed as the input for the next cycle (see Line 7-10 in Algorithm. 1).

Algorithm 1. Path refinement based on the least square

Input :
global_points : The multi-frame projected points in the global map;
H_t : SVM hyperplane at time t

1. **Initialization** : *local_points* ← ∅;
2. P_t ← Sample(H_t);
3. $P_t.\omega$ ← ω_t;
4. UpdateWeight(*global_points*);
5. *local_points* ← TransformToLocal(*global_points*) ∪ P_t;
6. *global_points* ← ∅;
7. **for** each point p_i **in** *local_points* **do**
8. **if** ($p_i.y > 0$ **and** $p_i.\omega > 0$) **then**

9. *global_points* ← TransformToGlobal(p_i);
10. **end**
11. **end**
12. EstimateModel(*global_points*);

Output:
Road model *M*.

Fig. 7 shows the different paths fitted respectively by least square fitting (LSF) and time weight based least square fitting (TWLSF). The light asterisks are the projected result of the multi-frame SVM hyperplane trajectories at time t while the dark ones are the results. As shown in Fig. 7, the small obstacle hasn't been detected by 64-beam LiDAR in the early time, so the path fitted by all the projected points is close to the obstacle. The path fitted by time weight based least square is relatively safer due to historical projected points have a little effect in the fitting procedure.

Note that, the SVM hyperplane trajectories are not coincident, the refined trajectory is decided by the large number of points which belong to SVM hyperplane trajectories. The refined trajectory is also away from the obstacles, and the complex hyperplane equation is transformed into a simple fourth order polynomial which is convenient for the following process.

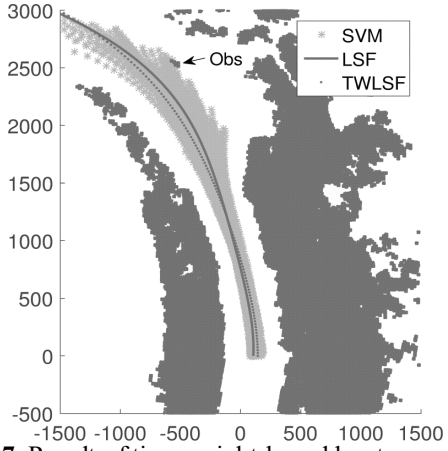


Figure 7. Result of time weight based least square fitting

3.3. Final Path Generation

In the real world, UGV's state should be considered during path planning. In this section, this information is combined with the least square path to generate the final path by Bezier function.

The least square path is divided evenly into five segments and the length of each segment is l. Then the control point C_1 is selected. As shown in Fig. 8, the angle between vector $\overrightarrow{OC_1}$ and the direction of the vehicle's instant velocity is φ_1, point O is the front axle center, $\varphi_1 \in [-\Delta\varphi, \Delta\varphi]$, φ_t represents the steering angle at time t, $\Delta\varphi$ is the maximum executable steering angle in each execution cycle. The coordinate of Point C_1 can be calculated by Eq. 10 in which the value of φ_1 should minimize the value of the Eq. 11.

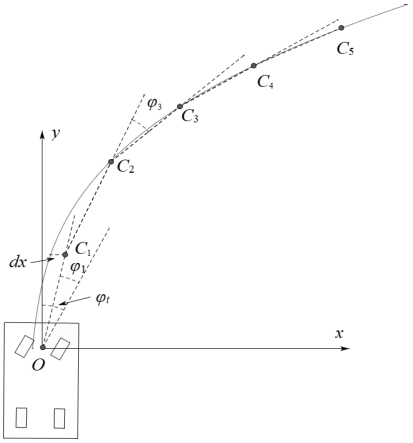


Figure 8. The selection of control points

$$\begin{cases} x_{C_1} = l \cdot \sin(\varphi_t + \varphi_1) \\ y_{C_1} = l \cdot \cos(\varphi_t + \varphi_1) \end{cases} \tag{10}$$

$$\varphi_i = \underset{\varphi \in [-\Delta\varphi, \Delta\varphi]}{\operatorname{argmin}} (w_1 \cdot \frac{|\varphi|}{\Delta\varphi} + w_2 \cdot \frac{d_x}{d_{\max}}) \tag{11}$$

where d_x represents the horizontal distance between point C_i and the least square fitting path, $d_{\max} = \max_{\varphi_i}(d_x)$. The first term of the Eq. 11 controls the curve of the path and the second term makes the path close to the least square fitting path which means keeping away from the obstacles.

The subsequent control points are selected by Eq. 11. When selecting the i^{th} control point C_i, the angle between $\overrightarrow{C_{i-1}C_i}$ and $\overrightarrow{C_{i-2}C_{i-1}}$ is φ_i which satisfies $|\varphi_i| \le \Delta\varphi$. The control point C_i is determined when φ_i minimizes the value of the Eq. 11. Then the final executable path is fitted by Bezier function with those control points.

$$B(t) = \sum_{i=0}^{5} \frac{5!}{i!(5-i)!}(1-t)^{5-i} t^i C_i, t \in [0,1] \tag{12}$$

4. EXPERIMENTS AND ANALYSIS

4.1. Experimental comparison

In order to verify the feasibility of the proposed method, experiments in the unstructured road environment is conducted on the UGV "Xingjian" (see Fig. 9).

Figure 9. Experiment platform

The method proposed in this paper is compared with FM^2, LDA method and Circle-based method (Chen, Rickert and Knoll, 2016). In the Circle-based method, the circles, which radius are the distance to the closest obstacle, extend from the start position in a tree-like manner. When the goal position is reached, a heuristic search which considers the kinodynamic vehicle model is adopted on the circle-path to generate an obstacle clearance path. An example of Circle-based method is shown in Fig. 10. The analysis is based on the experimental data of several typical scenes including straight road, branch road and curve road. The test results of these four methods are based on our own implementation. All tests run on a machine with 2.3GHz CPU and 8GB RAM.

In the experiment, two endpoints of the trajectory generated by proposed method is set be the starting and ending point of the FM^2 and Circle-based method. As shown in Fig. 11, the trajectories generated by proposed method is smoother than FM^2. In the velocity map generated by FM^2, the farther the area is away from the obstacle, the higher the velocity of the area is, so FM^2 will search the path along the high velocity area (see Fig. 11(f)). However, this will add extra length into the total path in some cases, such as, on the branch road while the proposed method and LDA method will not. The Circle-based method also suffers from the same problem as shown in Fig. 11(h).

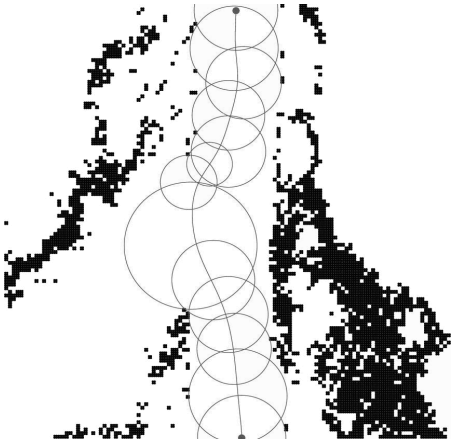
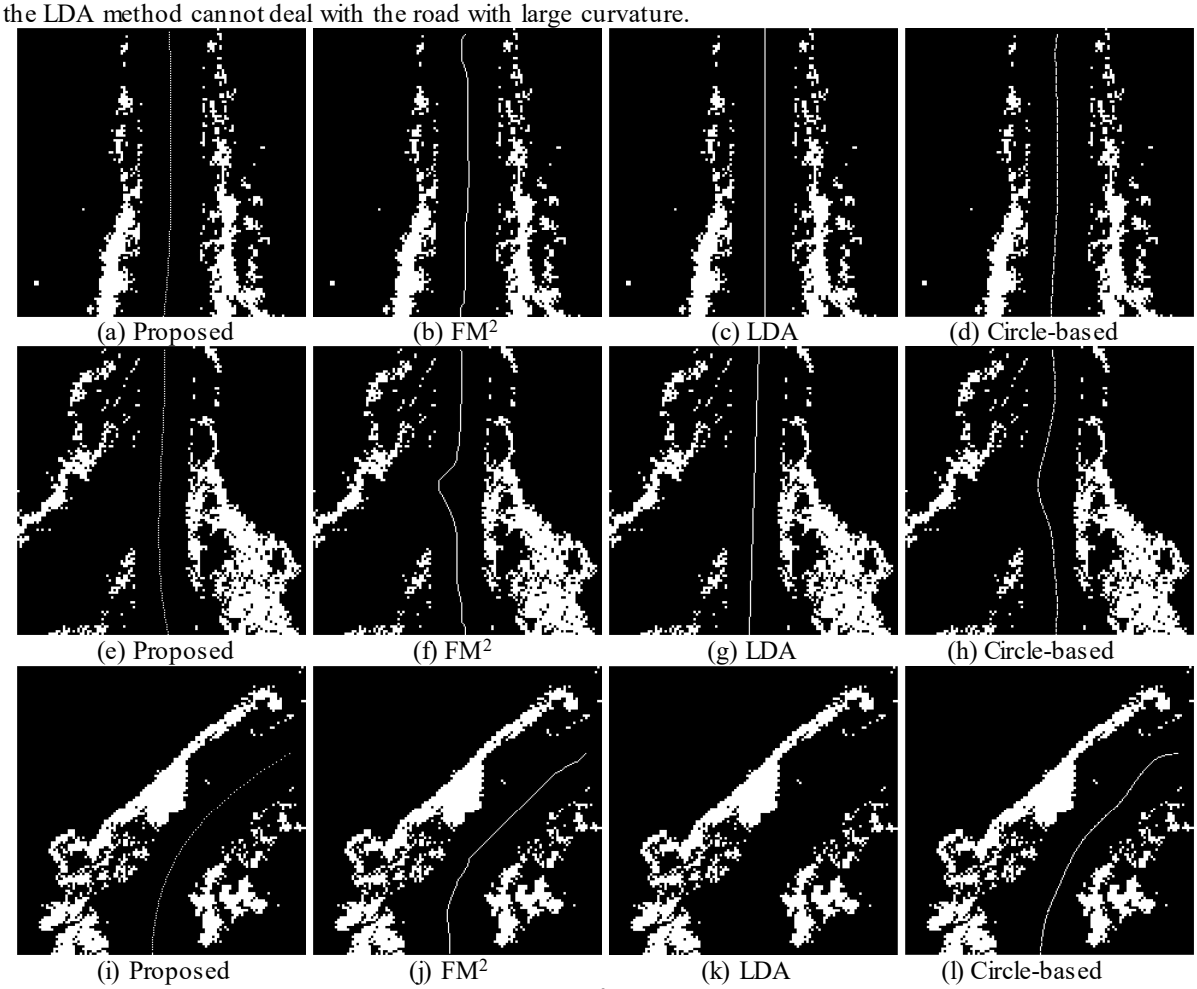


Figure 10. Circle-based path planning method

In the straight road case, the path of the proposed method is a little curve because of non-linear SVM (see Fig. 11(a) and Fig. 11(e). However, the path can be considered similar to a straight path. The results of the LDA method are more reasonable relatively (see Fig. 11(c) and Fig. 11(g)) due to its linearity. As shown in Fig. 11(k), the LDA method cannot deal with the road with large curvature.



(a) Proposed (b) FM^2 (c) LDA (d) Circle-based

(e) Proposed (f) FM^2 (g) LDA (h) Circle-based

(i) Proposed (j) FM^2 (k) LDA (l) Circle-based

Figure 11. The results of our proposed method, FM^2, LDA method and Circle-based method for different scenarios. (a)-(d) scenario 1: straight road; (e)-(h) scenario 2: branch road; (i)-(l) scenario 3: curve road

The maximum, minimum and average distance between the path and obstacle are shown in Fig. 12. From the point view of the path safety, the proposed method is better than LDA method and Circle-based method in straight road and curve road scenarios, and FM^2 only has one item better than the proposed method in straight road and curve road scenarios. In the branch scenario, FM^2 and Circle-based method has larger maximum distance than the proposed method due to the characteristic described above, and the proposed method also has relatively safety distance away from the obstacles.

The runtime of four methods are shown in Fig. 13. FM^2 has the worst performance, LDA method is almost within 50ms and Circle-based method is almost within 80ms. The running time of the proposed method is almost within 25ms which can satisfy the requirement of the real-time path planning for UGV.

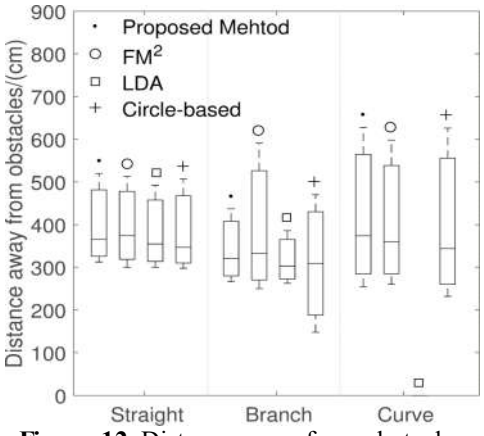


Figure 12. Distance away from obstacles

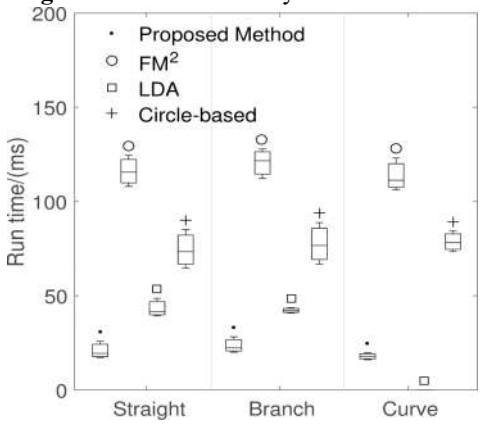


Figure 13. Run time

4.2. Path smoothness analysis

The comparisons of the path smoothness in different scenarios are shown in Fig. 14-16. Because of the result of LDA method is just a straight line, the comparisons only take account the other three methods. Note that, the result of FM^2, which is grid-based, has been fitted by Bezier function. Then the paths of FM^2, Circle-based method and the proposed method are all resampled uniformly. Let q_{i-1}, q_i and q_{i+1} be the continuous resampling points. The angle between $\overrightarrow{q_{i-1}q_i}$ and $\overrightarrow{q_iq_{i+1}}$ is used to represent the change rate of q_i.

As shown in Fig. 14-16, the change rate of the whole path of the proposed method is relatively small in contrast with the other two methods. It is smooth enough for UGV to trace. The change rate of the whole path of FM^2 changes greatly. Although the path of FM^2 is safe enough, it can not be traced by the UGV smoothly. The path of Circle-based method is affected by the positions of the circles which extend randomly. However the kinodynamic vehicle model is considered, the path is also not smooth enough.

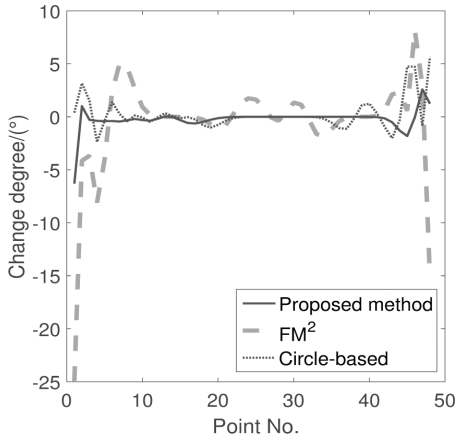


Figure 14. The comparison of the change rate on straight road

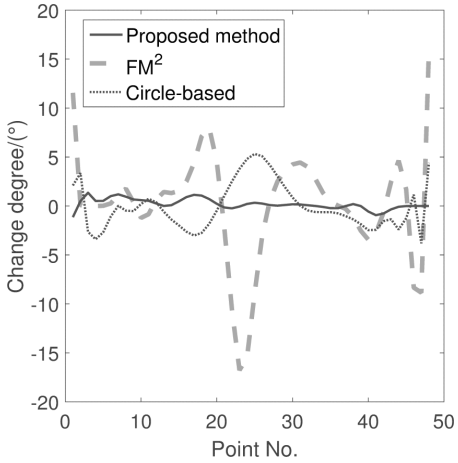


Figure 15. The comparison of the change rate on branch road

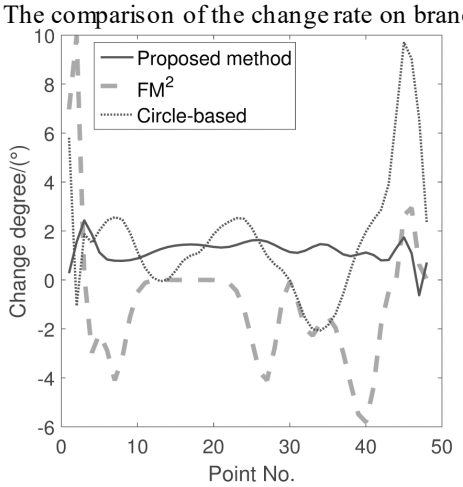
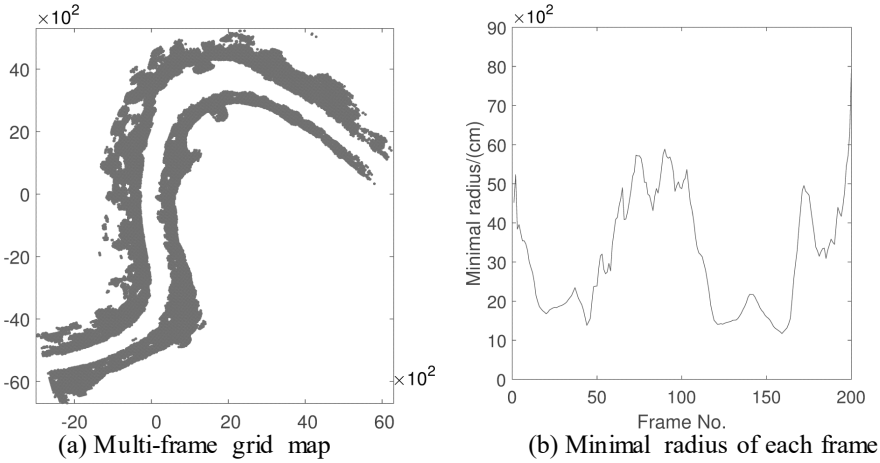


Figure 16. The comparison of the change rate on curve road

4.3. Road curvature analysis

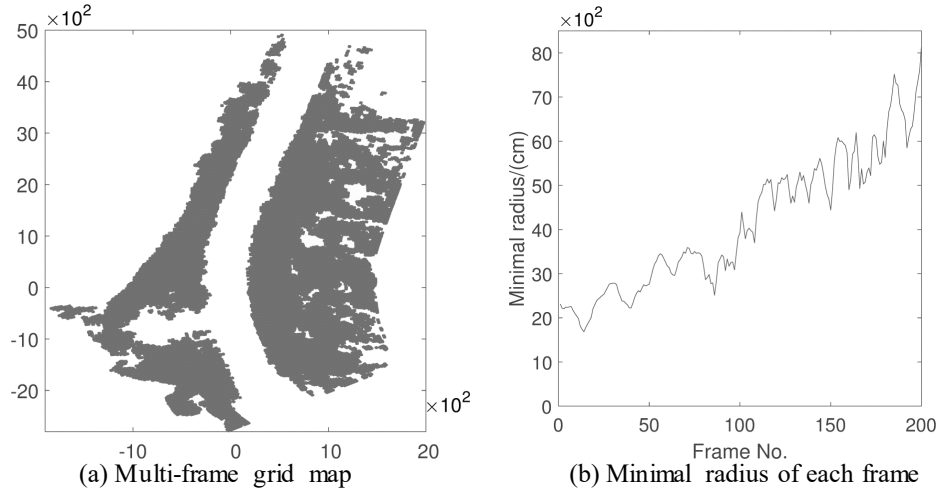
The road curvature analysis can effectively control speed and improve safety and comfort when the UGV is driving. So in this paper, the road radius of three different road sections are compared and analyzed based on the fourth order polynomial. The minimal radius of the refined path in each frame by time weight based least square fitting can be calculated by Eq. 13. Three kinds of road sections are analyzed. Road section 1 is a twisty road, road section 2 is the case of leaving the sharp turn, and road section 3 is a straight road. The results are shown in Fig. 17-19.

$$r = \frac{(1+f(y)'^2)^{3/2}}{|f(y)''|} \tag{13}$$



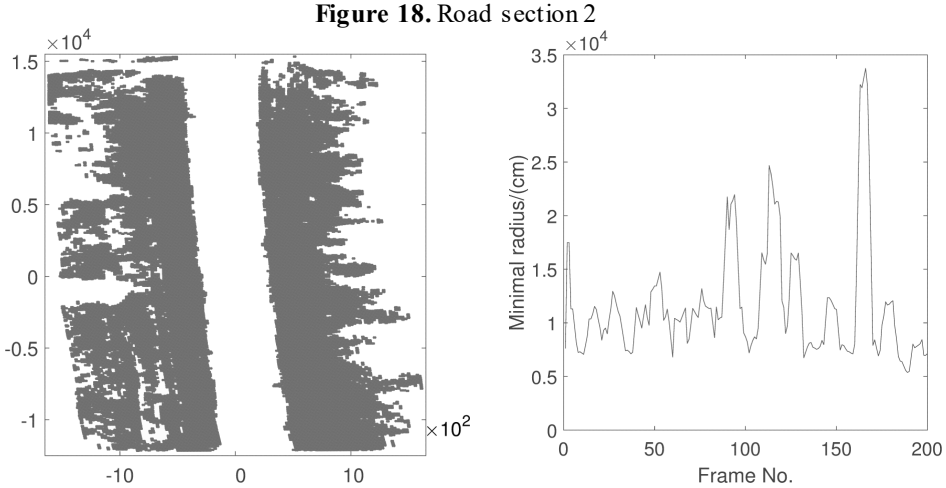
(a) Multi-frame grid map (b) Minimal radius of each frame

Figure 17. Road section 1



(a) Multi-frame grid map (b) Minimal radius of each frame

Figure 18. Road section 2



(a) Multi-frame grid map (b) Minimal radius of each frame

Figure 19. Road section 3

As shown in Fig. 17, the minimal radius profile of road section 1 has two valleys (see Fig. 17(b)) which represents the two curves (see Fig. 17(a)). As shown in Fig. 18, the minimal radius profile of road section 2 becomes big gradually (see Fig. 18(b)) which represents the case of leaving the sharp turn (see Fig. 18(a)). As shown in Fig. 19, the minimal radius profile of road section 3 is obviously larger (see Fig. 19(b)) than that of the other two road sections. This result is in accord with the fact that road section 3 is a straight road (see Fig. 19(a)). However, the minimal radius profile of road section 3 has a large degree of fluctuation, the refined path by least square fitting has no drastic changes from macro point of view due to the value of minimal radius is very large. So it has almost no influence on judging the road section 3 as a straight road.

Note that, the minimal radius is not the true value of the road. According to the trend of the minimal radius, the curve degree of the road can be identified and be treated as prior information for the steering control and velocity control of the UGV.

4. CONCLUSION EXPERIMENTS AND ANALYSIS

A real-time local path planning method based on 64-beam LiDAR and SVM is proposed. This method can effectively extract the path from the grid map and make up for the performance of vision based road detection algorithms is limited by poor lighting and weather in unstructured road environment. The final path is smooth, safe and easy to be tracked by controller. The validity and the correctness of the proposed method is verified on a real vehicle. The curve degree of the road has been simply identified according to the road model based on fourth order polynomial. Future work will focus on the further processing of curvature data and how to combine the curvature data with the steering control and velocity control of the UGV.

Acknowledgements

This work is supported by National Science Foundation of China (No.61403202 and No.61371040); China Post Doctor Foundation (2014M561654); Specialized Research Fund for the Doctoral Program of Higher

Education (No.20133219120035); National Major Project of Core Electronic Devices, High-end Generic Chips and Basic Software (No.2015zx01041101).

REFERENCES

Amini, H., and Karasfi, B. (2016) "New Approach to Road Detection in Challenging Outdoor Environment for Autonomous Vehicle", *2016 Artificial Intelligence and Robotics (IRANOPEN)*, pp. 7-11.

Bron, E. E., Smits, M., Niessen, W. J., and Klein, S. (2015) "Feature Selection based on the SVM Weight Vector for Classification of Dementia", *IEEE journal of biomedical and health informatics*, 19(5), pp. 1617-1626.

Canny, J. (1985) "A Voronoi Method for the Piano-movers Problem", Proceedings. *1985 IEEE International Conference on Robotics and Automation*, pp. 530-535.

Chen, C., Rickert, M., and Knoll, A. (2016) "Combining Task and Motion Planning for Intersection Assistance Systems", *2016 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1242-1247.

Fassbender, D., Mueller, A., and Wuensche, H. J. (2014) "Trajectory Planning for Car-like Robots in Unknown, Unstructured Environments", *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems,* pp. 3630-3635.

Goldhammer, M., Köhler, S., Doll, K., and Sick, B. (2015) "Camera based Pedestrian Path Prediction by Means of Polynomial Least-squares Approximation and Multilayer Perceptron Neural Networks", *2015 SAI Intelligent Systems Conference (IntelliSys)*, pp. 390-399.

Gómez, J. V., Mavridis, N., and Garrido, S. (2014) "Fast Marching Solution for the Social Path Planning Problem", *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2243-2248.

Gupta, S., Saurav, S., Singh, S., Saini, A. K., and Saini, R. (2015) "VLSI Architecture of Exponential Block for Non-linear SVM Classification", *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 528-532.

Khatib, O. (1986) "Real-time Obstacle Avoidance for Manipulators and Mobile Robots", *The international journal of robotics research*, 5(1), pp.90-98.

Koren, Y., and Borenstein, J. (1991) "Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation", *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, pp. 1398-1404.

Liu, Z., Tang, Z., and Ren, M. (2011) "Algorithm of Real-time Road Boundary Detection based on 3D Lidar", *Journal of Huazhong University of Science and Technology*, 39(S2), pp. 351-354.

Maarir, A., and Bouikhalene, B. (2016) "Roads Detection from Satellite Images Based on Active Contour Model and Distance Transform. International Conference on Computer Graphics", *2016 13th International Conference on Computer Graphics, Imaging and Visualization (CGiV),* pp. 94-98.

Matchen, T. D., and Nadler, B. R. (2014) "Image-based Target Tracking Using Least-squares Trajectory Estimation without a Priori Knowledge", *2014 IEEE Aerospace Conference*, pp. 1-12.

Singh, D., Khan, M. A., Bansal, A., and Bansal, N. (2015) "An Application of SVM in Character Recognition with Chain Code", *2015 Communication, Control and Intelligent Systems (CCIS)*, pp. 167-171.

Takahashi, O., and Schilling, R. J. (1989) "Motion Planning in a Plane Using Generalized Voronoi Diagrams", *IEEE Transactions on robotics and automation*, 5(2), pp. 143-150.

Wang, J., Sun, S., and Zhao, X. (2015) "Unstructured Road Detection and Path Tracking for Tracked Mobile Robot", *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pp. 535-539.

Wei, J., Snider, J. M., Gu, T., and Dolan, J. M. (2014) "A Behavioral Planning Framework for Autonomous Driving", *2014 IEEE Intelligent Vehicles Symposium Proceedings,* pp. 458-464.

Wilson, A. D., Schultz, J. A., Ansari, A. R., and Murphey, T. D. (2015) "Real-time Trajectory Synthesis for Information Maximization Using Sequential Action Control and Least-squares Estimation", *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4935-4940.

Yu, C., and Qiu, Q. W. (2013) "Hierarchical Robot Path Planning Algorithm based on Grid Map", *Journal of University of Chinese Academy of Sciences*, 30(4), pp. 528-538, 546.

Zu, Z., Hou, Y., Cui, D., and Xue, J. (2015) "Real-time Road Detection with Image Texture Analysis-based Vanishing Point Estimation", *2015 IEEE International Conference on Progress in Informatics and Computing (PIC)*, pp. 454-457.