2008 IEEE Intelligent Vehicles Symposium
Eindhoven University of Technology
Eindhoven, The Netherlands, June 4-6, 2008

A Parking Algorithm for an Autonomous Vehicle

Ming Feng Hsieh and Ümit Özguner

***Abstract*—Practical aspects of a real time auto parking controller are considered. A parking algorithm which can guarantee to find a parking path with any initial positions is proposed. The algorithm is theoretically proved and successfully applied to the OSU-ACT in the DARPA Urban Challenge 2007.**

I. Introduction

Many parking control algorithms have been developed in recent years [1]-[13]. Some approaches use the nonlinear control theories [2], [3], such as chained form conversion [3], to control the steering and speed. However, it is not easy to be implemented in practice because of highly nonlinear and computationally expensive conversion equations [5]. Some others [1], [4]-[7] use path planning methods to design a path to the parking position, after that use a path following controller to follow the path and park. This kind of method is more computationally efficient, but usually the generated path is very winding. It's more likely to have path following errors when following a winding path than following a uniform path. And a car may fail to park due to path following errors. Many researches concentrated on tracking and posture stabilization to solve this problem [8]-[13]. But results came back to be computationally expensive and always take a long time to stabilize a car.

An arc path parking algorithm is proposed in this paper. The parking path is a circular arc connecting the car position to the goal point. And the tangent angle of the arc at the goal point equals to the desired car yaw angle. A car is supposed to follow this path to park. But a car may not be able to follow this path initially due to an angle difference between the car yaw angle and the path angle. However, a car can always find an arc path (a final path) which is able to be followed by pulling forward or backward with a maximum steering angle (will be proved in the Appendix). Since an arc path can be generated very efficiently and can be followed very precisely, the problems of expensive computation and winding path are solved.

The details of the algorithm will be introduced in section II. Section III shows the simulation results and section IV shows the experimental result.

II. The Parking Algorithm

A. Car Model

Assuming the tire slip angle is zero during a parking procedure, a vehicle kinematic model with Ackerman steering as shown in Figure 1 and Eq. 1 is used in the parking study.

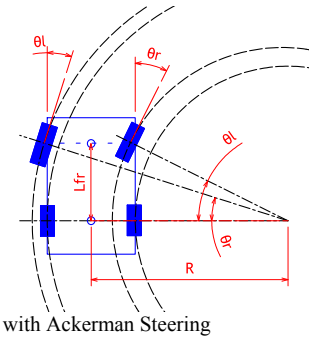

Figure 1 Vehicle Model with Ackerman Steering

$$\theta_{steer} = \frac{\theta_l + \theta_r}{2}$$
$$R = \frac{L_{fr}}{\tan\theta_{steer}} \qquad \text{Eq. 1}$$

In the equation above, θ_{steer} is the average steering angle and R is the vehicle turning radius. According to Ackerman steering theory, the vehicle yaw angle always equals to the velocity direction of the rear axle. In order to have a path direction equals to the car yaw angle, the vehicle position in this paper always means the center of the rear axle.

B. Local Coordinate

A local coordinate is defined in order to simplify expressions of equations.

The origin of the local coordinate is the center of the parking area in the horizontal direction. In the vertical direction, the origin should be as close to the final position as possible, and a car should be able to reach this point with the maximum steering angle and without hitting any boundaries. To satisfy these requirements, the origin of the local coordinate is defined by Eq. 2 and Figure 2.

$$R_{\min} = R_{turn} - W_c / 2$$
$$R_{\max} = \begin{cases} Parking \quad Forward : \sqrt{(R_{turn} + W_c/2)^2 + L_{fr}^{\ 2}} \\ Parking \quad Backward : \sqrt{(R_{turn} + W_c/2)^2 + (L_c - L_{fr})^2} \end{cases}$$
$$X0 = Wp/2$$
$$if\,(R\max^2 - (R\min + (Wp/2)^2)^2) >= 0$$
$$Y0 = \sqrt{R\max^2 - (R\min + (Wp/2)^2)^2}$$
$$if\,(R\max^2 - (R\min + (Wp/2)^2)^2) < 0$$
$$Y0 = -\sqrt{(WcWp/2) - (Wc/4)^2}$$

Eq. 2

In Eq. 2, Wp is the width of the parking spot, Wc is the car width, L_c is the car length, L_{fr} is the distance from front bumper to rear axle, R_{turn} is the vehicle minimum turning radius by the vehicle center, and $R_{\min}$ / $R_{\max}$ are the radiuses of the minimum/maximum circles a vehicle body can go through according to R_{turn}. The minimum turning radius of the car can always be obtained from vehicle specification.

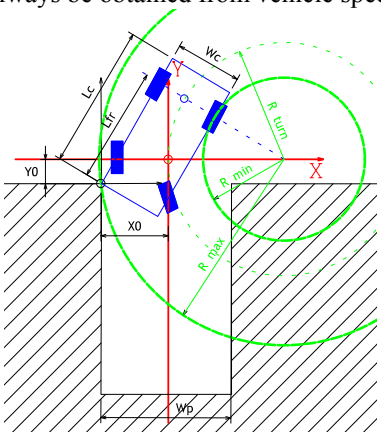

Figure 2 Local Coordinate

C. Parking Procedure

The concept of the proposed parking algorithm is to use a circular arc as a parking path as shown in Figure 3. An arc path has the advantage of small path following error and computation simplicity. However, a car may not be able to follow this path initially because the car yaw angle could be different from the tangent angle of the path as shown in Figure 5. An idea of using the maximum steering angle driving forward or backward to find a point where the car yaw angle equals to the tangent angle of the path is proposed. The Appendix will prove the point always exists, and the path radius is always larger than the car's minimum turning radius. After the point is reached, a car can follow the path to the desired parking position.

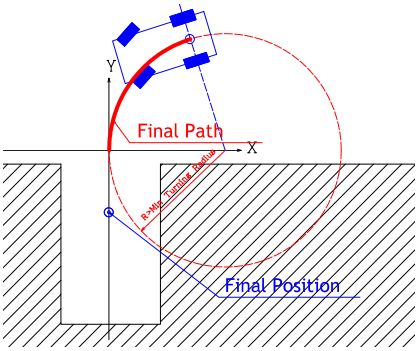

Figure 3 Final Path

The parking algorithm includes three steps as shown in Figure 4.

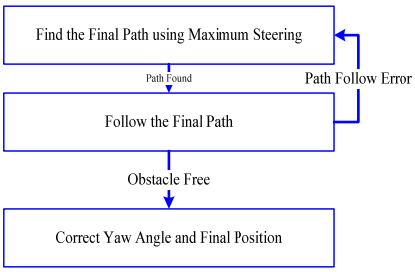

Figure 4 Parking Procedure

The general scenario will be introduced in 1). The general scenario means the car can pull either forward or backward into the parking spot. In DARPA Urban Challenge, autonomous vehicles were required to pull only "forward" into a parking spot. 2) will introduce the strategy to deal with this special requirement.

1) General Scenario

Figure 5 to Figure 7 show the three parking steps.

In the first step, the car drives forward or backward using the maximum steering angle turning left or right to find a finale path. A final path is a circular arc with a radius larger than the minimum turning radius of the car. The arc circle is generated by two points, the origin of the local coordinate and the car's position, and an angular constraint, the circle tangent to the Y axis, and it is updated in real time. The final path is found when the car's yaw angle equals to the tangent angle of the circle as shown in Figure 6.

Initially, three circles will be generated: the two green dash line circles and the red dash line circle, the final path circle, as shown in Figure 5. The two green dash line circles are the predicted vehicle paths if the car turning left or right using the maximum steering angle. In most situations, as shown in Figure 5, the red dash line circle can not be the final path in the beginning, because the car's yaw angle doesn't match with the path. In this case the car should follow one of the green circles and update the red circle in real time until the yaw angle equals to the tangent angle of the final path.

The choice of the turning direction and the driving direction depends on the final path prediction. There are four final path candidates (the red circles as shown in Figure 20) according to turning left/right and pull forward/backward. Appendix will show at least one of the red circles has radius larger than the minimum turning radius, and the contact point of the two circle is above the X axis (outsides the boundary). The one which tallies with the requirements is predicted as the final path. According to the selected final path, the car decides turning left or right and going forward or backward.

Since path following errors must exist, there is no specific path assigned to the car before final path is found. The car just follows the left/right and forward/backward commands and keep updating the possible final path until the yaw angle match the final pat.

After the final path is found, as shown in Figure 6, the vehicle follows the final path to the desired position. The car can still hit the boundary due to sensor errors or path following errors. In this situation the car backs to the first step

to find the final path again.

After the final path follow is completed, the car is almost in the parking spot and the yaw angle should very close to the desired yaw angle as shown in Figure 7. The last step is to move the car into the parking spot and correct the yaw angle if the error is unacceptable.

The equations of the circles and angles will be presented in section 2) and Appendix.

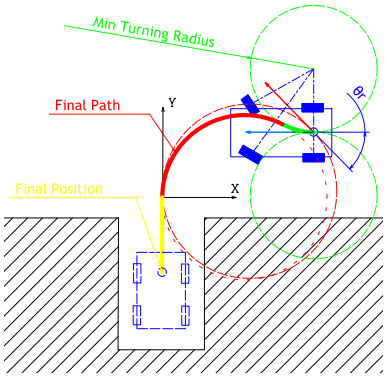

Figure 5 Parking - Step 1

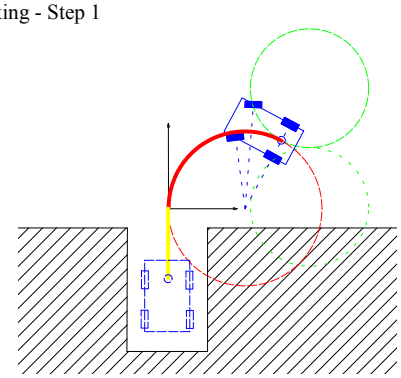
Figure 6 Parking – Step 2

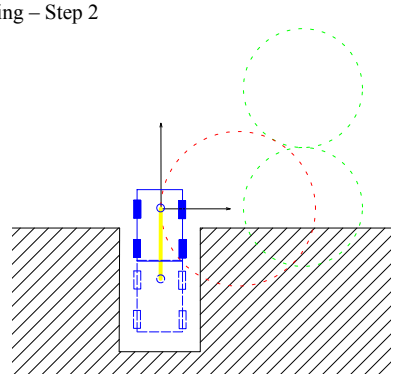
Figure 7 Parking - Step 3

2) DARPA Urban Challenge

In the DARPA Urban Challenge, a parking lot is defined as a "zone". The OSU-ACT is controlled by an "obstacle avoidance controller" in zones. The obstacle avoidance controller can keep the car away from obstacles and bring it to a desired point. For the parking case, the goal point of the obstacle avoidance controller is the final position of the parking spot. The obstacle avoidance controller hands over the control authorities to "parking" when the distance to the final point is smaller than 8 meters. DARPA claims that paths to a parking spot would not be totally blocked. This maneuver can make sure parking starts from the side which is obstacle free.

A rule in DARPA Urban Challenge requires pulling a car forward into the parking spot. This can be done by the following strategies.

The car goes forward initially to approach the parking spot. In order to decrease the θ_r, as shown in Figure 5, the steering direction θ_{steer} can be obtained be the following equations.

$$
\begin{aligned}
&R = |(x^2 + y^2)/(2 * x))| \\
&\theta_{FinalPath} = -\sin^{-1}(y/R) + \pi/2 \\
&\theta_{FinalPath} - \theta_{yaw} = \theta_r \\
&\left\{ if(\theta_{FinalPath} > \theta_{yaw}) : \theta_{steer} = left \right\} \\
&\left\{ if(\theta_{FinalPath} < \theta_{yaw}) : \theta_{steer} = right \right\}
\end{aligned}
\qquad \text{Eq. 3}
$$

In Eq. 3, R is the radius of the possible final path circle, the red dash line circle in Figure 5, x and y is the vehicle position according to the local coordinate, $\theta_{FinalPath}$ is the tangent angle of the possible final path circle at the vehicle position, and θ_{yaw} is the vehicle's yaw angle. The final path is found when $\theta_{FinalPath} = \theta_{yaw}$.

This maneuver can definitely find a final path circle. However, it is possible that the radius of the final path circle is smaller than the minimum turning radius which is impossible to be followed, as shown in Figure 8. In this situation, the car ignores the final path and keeps going forward. In order to minimize θ_r, the car will follow the blue dash line circle in Figure 8. During this course, according to Appendix, a new green dashed line circle will be found which can introduce a final path with radius larger than the minimum turning radius as shown in Figure 9.

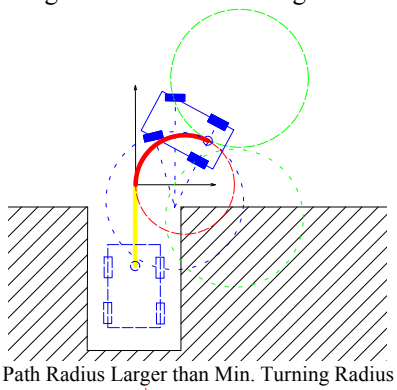
Figure 8 Final Path Radius Larger than Min. Turning Radius

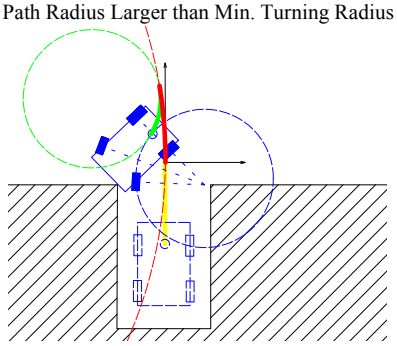
Figure 9 Find Final Path from Second Maneuver

The strategies of the parking algorithm for the DARPA Urban Challenge can be concluded as follow:

1. Pull forward and steer in the direction of θ_{steer} with the maximum steering angle to find a final path.
2. If hit an obstacle or a boundary, change the moving direction from forward/backward to backward/forward.
3. If θ_r =0 and the final path radius is larger than minimum turning radius, follow the final path. If hit any obstacle or a boundary during following, backs to step 2.
4. After the path following is finished, forward straight to the final position.

III. Simulations

Figure 10 to Figure 14 show the simulation results of the parking algorithm.

Figure 10 to Figure 13 show simulations with different initial conditions. Figure 14 is a harder case. The initial position is the same with Figure 13, but the road is narrow. Apparently it is also hard for a human driver. The simulation shows the car successfully parked using two forward-backward maneuvers.

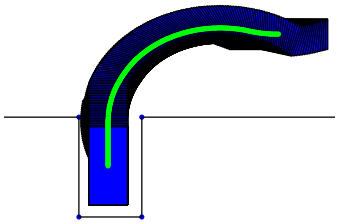

Figure 10 General Situation

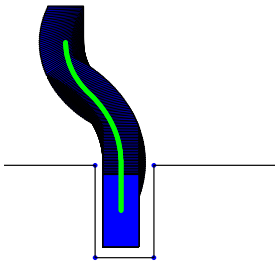

Figure 11 Park from Different Initial Position

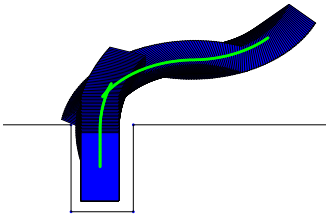

Figure 12 Park from Different Initial Yaw Angle

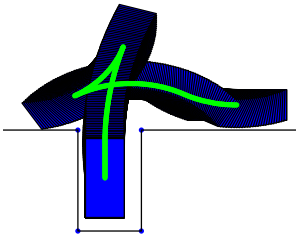

Figure 13 Initial Position Close to Boundary

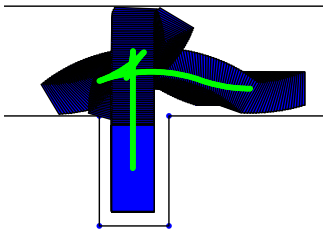

Figure 14 Park at Narrow Road

IV. Experiment Results

The parking algorithm is applied on the autonomous vehicle OSU-ACT as shown in Figure 15.

Figure 15 OSU-ACT

Figure 16 shows a zone in the DARPA Urban Challenge. The red line shows the path of the OSU-ACT. The green rectangle highlights the parking area. In the parking area, two other cars were parked on the left and right sides of the assigned parking spot. The OSU-ACT successfully park into the spot every time.

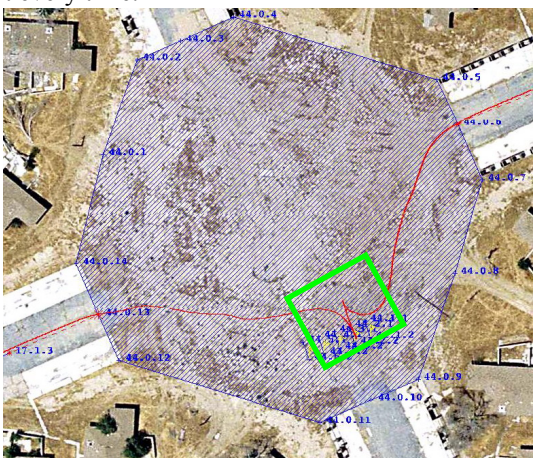

Figure 16 Parking in Urban Challenge 2007

Figure 17 to Figure 19 show other experiment results with different situations. The red points are the vehicle trajectory, blue lines are the boundaries, and the green points are obstacles from the LIDAR scan.

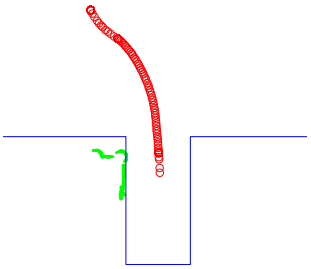

Figure 17 General Situation

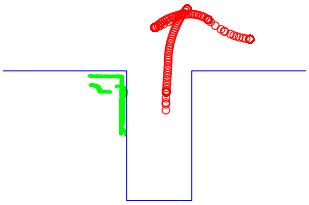

Figure 18 Initial Position Close to Boundary

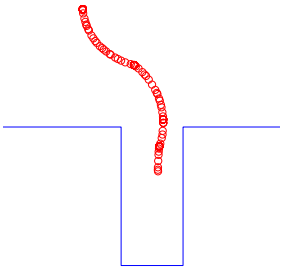

Figure 19 Park from Different Initial Position

V. Conclusion

The proposed parking algorithm provides a practical method of auto parking. No expensive computation is required, and was theoretically and practically proved that the algorithm is able to park with any initial positions. The simulation results also showed the algorithm can control a car to park in a limited space which is hard for a human driver. The algorithm is applied on the OSU-ACT in the DARPA Urban Challenge 2007.

Appendix

The objective of the Appendix is to prove that in most situations there exists at least one final path which has radius larger than the minimum turning radius $R_{MinTurn}$ and the connecting point of the corresponding vehicle path (minimum radius circle) and the final path circle is above the x axis. There is a small region where a car cannot find a final path. But in general situations a car would not start parking from this region. Even if a car start parking from this region, a car can just pull forward or backward with any steering angle for a short distance to get out of this region.

As shown in Figure 20, C1, C2, C3, and C4 are the four possible final path circles according to the vehicle initial position (X_c, Y_c). Radiuses of C1 and C2 are smaller than $R_{MinTurn}$ and the tangent point of C3 is under x axis. In this situation C4 is chosen as the final path.

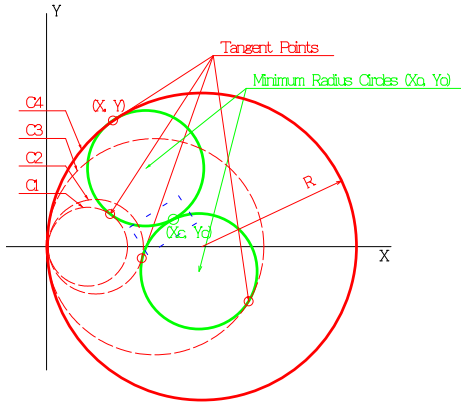

Figure 20 Proof of the Final Path Circle

(X_c, y_c) is the car's initial position. Because a car could not in boundaries, y_c must be greater than 0. According to this constraint, one of the minimum radius circle centers (X_0, Y_0) must conforms to $Y_0 > 0$. The equation of the minimum radius circle is:

$$(X - X_0)^2 + (Y - Y_0)^2 = R_{MinTurn}^2 \qquad \text{Eq. 8}$$

And the equation of the corresponding final path circle is:

$$(X - R)^2 + Y^2 = R^2 \qquad \text{Eq. 9}$$

Since the final path circle is tangent to the minimum radius circle, the following relation can be obtained.

$$\frac{R - X}{Y} = \frac{X0 - X}{Y - Y0} \qquad \text{Eq. 10}$$

Solve the three equations, the tangent point (X, Y) and the radius of the final path circle R are obtained as:

$$\begin{aligned} X &= \frac{(R_{MinTurn} + X_o)(X_o^2 + Y_0^2 - R_{MinTurn}^2)}{(R_{MinTurn} + X_o)^2 + Y_0^2} \\ or\ &\frac{(R_{MinTurn} - X_o)(R_{MinTurn}^2 - (X_o^2 + Y_0^2))}{(R_{MinTurn} - X_o)^2 + Y_0^2}; \\ Y &= \frac{Y_0(X_o^2 + Y_0^2 - R_{MinTurn}^2)}{(R_{MinTurn} + X_o)^2 + Y_0^2} \\ or\ &\frac{Y_0(X_o^2 + Y_0^2 - R_{MinTurn}^2)}{(R_{MinTurn} - X_o)^2 + Y_0^2}; \\ R &= \frac{1}{2}\frac{X_o^2 + Y_0^2 - R_{MinTurn}^2}{R_{MinTurn} + X_o} \\ or\ &-\frac{-(X_o^2 + Y_0^2) + R_{MinTurn}^2}{R_{MinTurn} - X_o}; \end{aligned} \qquad \text{Eq. 11}$$

The area where $Y > 0$ and $R > R_{MinTurn}$ can be found as follow.

$$
\begin{aligned}
&Y > 0 \rightarrow {X_0}^2 + {Y_0}^2 > {R_{MinTurn}}^2 \\
&R > R_{MinTurn} \rightarrow \\
&when(X_o > R_{MinTurn}) \\
&{Y_0}^2 > -(X_o - R_{MinTurn})^2 \\
&when(0 < X_o < R_{MinTurn}) \\
&{Y_0}^2 > (3R_{MinTurn} + X_o)(R_{MinTurn} - X_o) \\
&when(0 > X_o > -R_{MinTurn}) \\
&{Y_0}^2 > (3R_{MinTurn} - X_o)(R_{MinTurn} + X_o) \\
&when(X_o < -R_{MinTurn}) \\
&{Y_0}^2 > -(X_o + R_{MinTurn})^2
\end{aligned}
\qquad \text{Eq. 12}
$$

Figure 21 shows the area where the final path circle exist and not.

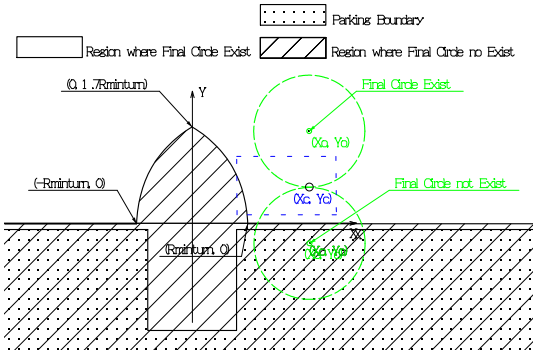

Figure 21 Region where Final Circle Exist/Not Exist

A final path circle exists when at least one of the minimum radius circle is centered in the exist region.

Only in some special initial positions both the minimum radius circles are not centered in the existing region as shown in Figure 22. This kind of situation rarely happens in practice, because a car always starts parking with a distance away from the origin. For example in the DARPA Urban Challenge the OSU-ACT started parking with a distance of 8 meters from the final position. With this distance at least one of the minimum radius circle is centered in the exist region. In the worst case like Figure 22, a car just need to pull away from the parking spot to move one of the (X_0 , Y_0) to the existing region as shown in Figure 23.

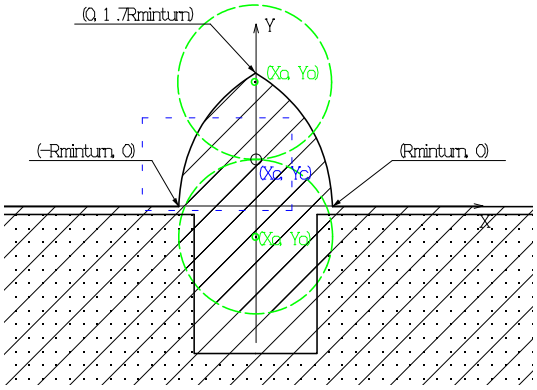

Figure 22 Situation where no Final Circle Exist

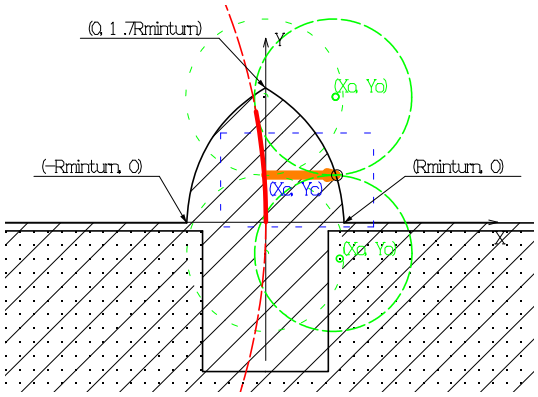

Figure 23 Pull Forward/Backward to Find Final Path

ACKNOWLEDGMENT

Our thanks are also due to Dr. Dan Dailey for the help he has provided.

REFERENCES

[1] Y. Z. Zhu, Y. Zheng, and U Ozguner, "Waypoint Selection in Constrained Domains (for Cooperative System)", Advances in Cooperative Control and Optimization (Lecture Note in Control and Information Science, Vol. 369), Elsevier, 2007.

[2] T. Inoue, M. Q. Dao, and K. Liu, "Development of an Auto-Parking System with Physical Limitation", SICE 2004 Annual Conference, 2004.

[3] R.M. Murray, S.S. Sastry "Nonholonomic Motion Planning: Steering Using Sinusoids" IEEE trans. on Automatic Control, 1993.

[4] M. Wada, K. S. Yoon, and H. Hashimoto, "Development of Advanced Parking Assistance System", Industrial Electronics, IEEE Transactions on, 2003.

[5] K. Lee, D. Kim, W. Chung, H. W. Chang, and P. Yoon, "Car Parking Control Using a Trajectory Tracking Controller", SICE-ICASE International Joint Conference 2006.

[6] J. A. Reeds and L. A. Shepp, "Optimal Path for a Car That Goes Both Forward and Backwards", Pacific Journal of Mathematics, 1990.

[7] L. E. Dubins, "On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents", American Journal of Mathematics, 1957.

[8] I. E. Paromtchik and C. Laugier, "Motion Generation and Control for Parking an Autonomous Vehicle", IEEE International Conference on Robotics and Automation, 1996.

[9] I. E. Paromtchik and C. Laugier, "Autonomous Parallel Parking of a Nonholonomic Vehicle", Intelligent Vehicles Symposium, 1996.

[10] T. C. Lee, C. Y. Tsai, and K. T. Song, Fast Parking Control of Mobile Robots: A Motion Planning Approach With Experimental Validation", IEEE Transactions on Control System Technology, VOL. 12, NO. 5, 2004/

[11] T. H. S. Li and S. J .Chang, "Autonomous Fuzzy Parking Control of a Car-Like Mobile Robot", IEEE Transactions on Systems, Msn, Aan Cybernetics, 2003.

[12] D. Gorinevsky, A. Kapitanovsky, and A. Goldenberg, "Neural network architecture for trajectory generation and control of automated car parking," Transactions on Systems Technology., vol. 4, pp. 50–56, 1996.

[13] H. An, T. Yoshino, D. Kashimoto, M. Okubo, Y. Sakai, and T. Hamamoto, "Improvement of convergence to goal for wheeled mobile robot using parking motion," in IEEE International Conference Intelligent Robots Systems, 1999.