Keywords: Path-planning, nonholonomic, robots, control, collaborative

ABSTRACT
In this paper a path-planning algorithm is presented for two nonholonomic robotic vehicles engaged in a docking maneuver. This algorithm was implemented in a tracking controller for two parallel-steered robots required to maneuver and dock with one another. In the coordination control of multiple robots, compatible trajectories for each robot are essential. This is especially critical for nonholonomic robots performing collaborative tasks. A polynomial based trajectory is developed using a bisection search method to ensure that the minimum turning radius of each vehicle is not violated for any part on the trajectory. A tracking controller is developed to ensure each vehicle traverses the prescribed trajectory to a docking location.

INTRODUCTION
Autonomous maneuvering can be a difficult and broad issue for multiple robots performing various tasks. To ensure proper movements performed by the robots, trajectories in time must first be achieved to guarantee efficient, non-impeding traversals. The applications where this is pertinent are transportation, surveying, formation and following. Position feedback and path planning are two major areas of study that have developed from these endeavors. Position feedback can be determined by on board [Das et al. 2001] or off board [Dixon et al. 2001] "cameras" while processing of the data can be performed anywhere. Path planning may involve a number of different methods. These paths can be discrete, continuous, or piecewise defined. Discrete paths such as quadtrees [Yahja et al. 1998] and grids can be formed into continuous paths by adding splines [Egerstedt et al. 2001] or other numerical techniques. Continuous and piecewise [Nagy et al. 2001] defined curves are also used to generate paths for nonholonomic vehicles. Usually robots can be tracked as they follow these paths using classical control schemes.

In this paper, an architecture is presented and implemented that uses classical control methods to track the position of two robots in a workspace. The goal of their maneuvers is to have the parallel-steered robots traverse a predetermined trajectory and result in a configuration that allows for front-to-front docking. The predetermined trajectory is obtained using a simplified polynomial-bisection search technique. Using a local frame of reference, this technique lends itself to real-time path-planning generation and control.

POSITION SYSTEM AND WORKSPACE
Using an off-board 3D camera system and robot’s equipped with LEDs, two robots are tracked and their position fed back to the controller. Coordinate transformations are performed to construct a reference frame, E, whose basis (\(\hat{e}_1, \hat{e}_2\)) contains all of the maneuvers performed by the robots.

Robot B’s initial configuration defines the origin and the orientation of the E frame of reference as denoted in figure 1. Robot B’s initial heading defines the \(\hat{e}_2\) axis. Robot A (on the right) may be located in the first or fourth quadrant of the \((\hat{e}_1, \hat{e}_2)\) plane and its heading, \(\psi_A\), must range from 90° to 270°. The position \((x_A, y_A)\) and heading \(\psi_A\) of Robot A are defined in equations 1 and 2. The quantities, \(\theta_A\), \(R_A\), \(\hat{a}, \hat{b}, \hat{\epsilon}_1, \hat{\epsilon}_2\), \(\hat{\epsilon}_b\), and, \(\hat{\epsilon}_e\), are obtained via the position system and measured LED signals.

\[
\begin{align*}
x_A & = R_A \cos \theta_A \\
y_A & = R_A \sin \theta_A \\
\hat{\epsilon}_b & = \frac{\hat{a} \cdot \hat{\epsilon}_1}{\|\hat{a}\| \|\hat{\epsilon}_1\|} \\
\hat{\epsilon}_e & = \text{sgn}(\hat{a} \cdot \hat{\epsilon}_1)
\end{align*}
\]

The above equations can also be applied to Robot B by the substitution of B for A and the heading vector \(\hat{b}\) for \(\hat{a}\). These definitions are shown in figure 1.

TRAJECTORY PLANNER
The trajectory, which allows the robots to maneuver toward each other for docking, is described by a fourth order polynomial. At least a third order polynomial is needed to satisfy the four end conditions (robots’ pose) of the trajectory. The constraint induced by the minimum turning radius of the vehicle is utilized to construct a fourth order polynomial. The five constraints are used below to develop an acceptable polynomial trajectory.
A general equation for a fourth order polynomial is
\[ y(x) = ax^4 + bx^3 + cx^2 + dx + e \]  
where the initial conditions of Robot B at the origin are
\[ y(0) = y_B(x_B) = 0 \]  
\[ y'(0) = y'_B(x_B) = 0 \]  
Substituting the conditions (4 and 5) into (3) gives
\[ y(x) = ax^4 + bx^3 + cx^2 \]  
The initial heading for Robot A is described by
\[ y_A = y(x_A) \]  
Conditions (7) and (8) can be used to rewrite (6) as
\[ y(x) = a(c)x^4 + b(x^3) + cx^2 \]  
The fifth constraint produced by the turning radii of the vehicle is defined to be
\[ \rho = \frac{(1 + y'(x))^2}{y''(x)} > \rho_{min} \quad \forall x \]  
The single unknown, \( c \), is obtained from the turning radii constraint, which is given in (10).

In addition to the above constraints previously mentioned, it was observed that as the minimum turning radius of the trajectory increases the length of the curve decreases (figure 2). The trajectory’s minimum turning radius over all \( x \) is therefore maximized to reduce the length of the curve and ensure feasibility for the robot. The robot’s minimum turning radius is 500 mm and therefore the minimum is required to be greater than 500 mm. That is,
\[ \max_{x} \{\min_{\rho} \} > 500 \text{ mm} \]  

The polynomial-bisection algorithm is given pictorially in figure 3. Equations 10 and 11 form the basis of a trajectory planning algorithm in which the coefficients of the polynomial are determined by finding \( c \) for a given set of end conditions. Using a modified bisection search, an “optimal” \( c \) is obtained that satisfies the constraint in equation 11.
The implemented bisection search determines the value of $c$ that maximizes $\rho_{\text{min}}$. The process is outlined in figure 4. Given initial guesses of $c_1$ (left bound) and $c_2$ (right bound), $\rho'_{\text{min}}(c)$ is numerically evaluated and its sign checked at $c_1$ and $c_2$. If $\rho'_{\text{min}}(c_1)$ is positive and $\rho'_{\text{min}}(c_2)$ is negative then a new $c_1$ is selected narrowing the bound on $c$. That is, $c_1$ moves half the distance to $c_2$. With the updated $c_1$ and $c_2$ values, the convergence of the search is checked by comparing $c_2-c_1$ to a selected error bound, $\epsilon$. If the search is not terminated, then a new $c_1$ or $c_2$ is generated based on signs of $\rho'_{\text{min}}(c_1)$ and $\rho'_{\text{min}}(c_2)$, repeating the process until the convergence condition is met. If not and $\rho(c_1) > \rho(c_2)$, $c_2$ becomes $c_1$, and $c_1$ becomes the previous $c_1$. Then the criteria for length and minimum radius of curvature are checked. If both are satisfied, $c$ is set to $c_1$. The initial guesses converge to a maximum rather quickly ($<1$ sec).

An alternate search approach for obtaining $c$ can be seen from figure 5. Large values of $\rho$ imply the robot's trajectory is nearly straight (e.g. the spike near $(0,0)$). The area of interest is not in the peaks of the plot but where the radius of curvature is very small. The minimum radius of curvature over the trajectories in $x$ for each $c$ can be viewed by looking down the $x$-axis, i.e. projection in the $\rho$-$c$ plane (figure 6). The bottom edge of the shaded areas is $\rho_{\text{min}}$ for each $c$ value (figure 7). The results from either approach were within 99.4% of each other.
TRACKING CONTROL

With the navigational information provided by the camera system, tracking control was determined to command the robots as they traverse the selected path. Classical control schemes are used to provide appropriate speed and steering commands such that the two robots’ deviation from the trajectory is minimized.

Speed Controller

Figure 8 shows the closed loop position feedback implemented for speed control. As the robot approaches its target location its speed decreases. When the error between the commanded position and the current location (13) is large the commanded speed (14) is increased.

\[
R_s = \sqrt{(X-x)^2 + (Y-y)^2} \tag{12}
\]

\[
\text{Error}_{\text{speed}} = R_s \cdot \text{sgn}(x - X) \tag{13}
\]

\[
\text{speed} = k_{p_{\text{speed}}} \cdot \text{Error}_{\text{speed}} + k_{d_{\text{speed}}} \cdot \frac{\Delta \text{Error}_{\text{speed}}}{dt} \tag{14}
\]

where \(\Delta \text{Error}\) defines the change in error between two time increments.

When the robots approach each other along the same trajectory from opposite ends and the error from the commanded position is larger than the distance between the robots the error is described by (15) such that the robots slow down as they near each other.

\[
\text{Error}_{\text{speed}} = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2} \cdot \text{sgn}(x_A - x_B) \tag{15}
\]

Steering Controller (Heading and Position)

The steering of each robot is controlled using a proportional-derivative (PD) controller where the error signal consists of the position (16) and heading (17). This framework is presented in figure 9. The commanded steering angle is provided in (17).

\[
\text{Error}_{\text{heading}} = \tan^{-1}\left(\frac{Y - y}{X - x}\right) - \psi \tag{16}
\]

\[
\text{Error}_{\psi} = (Y' - \tan \psi) \tag{17}
\]

\[
\phi = \phi_0 + \text{Error}_{\text{heading}} \cdot k_{p_{\text{heading}}} + \frac{\Delta \text{Error}_{\text{heading}}}{dt} \cdot k_{d_{\text{heading}}} + \text{Error}_{\psi} \cdot k_{p_{\psi}} + \frac{\Delta \text{Error}_{\psi}}{dt} \cdot k_{d_{\psi}} \tag{18}
\]

Figure 9. Closed loop controller for the steering

COMBINED TRAJECTORY PLANNER AND TRACKING CONTROL

Combining the tracking controller with the polynomial-bisection algorithm, a series of experiments were conducted using two nonholonomic wheeled mobile robots. The speed controller was first tuned to determine which control gain constants would result in superior performance.

Speed Control Tuning

The speed controller was implemented such that the vehicles were commanded to travel on a straight path. This allowed for focusing primarily on the speed controller while neglecting the steering controller. The desired path as a function of time was calculated by applying a desired speed to the trajectory. Figure 10 shows the resulting proportional-derivative (PD) controller’s performance for Robot B. Table 1 summarizes the results of the implemented speed controllers for each robot. The negative gains are required to convert the frame of reference of Robot B.

The plot on the right of figure 10 shows the error of the robot’s position in both \(x\) and \(y\), defined by the RMS of the error, \(R_s\). Comparing to \(x\) errors, the large errors in \(R_s\) were mainly due to errors in \(y\) implying the need for an improved steering controller. The pauses seen in the left of figure 10 are attributed to the position error’s sign as seen in equation...
12. When the error in the x direction becomes positive (meaning the robot is ahead of its commanded position) the drive motors are disengaged causing the robot to slow.

![Diagram of controller and deviation from commanded position](image1)

**Figure 10.** Robot B’s x response and RMS errors with respect to time using PD control.

<table>
<thead>
<tr>
<th></th>
<th>Robot A</th>
<th>Robot B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proportional ($K_p$)</td>
<td>1100</td>
<td>-3000</td>
</tr>
<tr>
<td>Derivative ($K_d$)</td>
<td>20</td>
<td>-30</td>
</tr>
<tr>
<td>Maximum Lag (mm)</td>
<td>128</td>
<td>33</td>
</tr>
<tr>
<td># of Pauses or Delays</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

**Table 1.** Results of the speed controller

Steering Controller Results

To enhance the speed control the steering controller is implemented for reducing positional errors. Several tests were performed for each robot and representative results for Robot A are shown in figures 11 and 12 for different steering control gains, $K_{ap(x,y)}$.

![Diagram of steering control](image2)

**Figure 11.** Steering control of Robot A: $K_{ap} = 500$

Figure 11 demonstrates a system that had too much gain on heading and drove the system away from the trajectory while trying to drive parallel to a slope ahead of it until the position controller pulled it back to the curve (figure 11, sections 1 and 2). The heading controller gain was then tuned to reduce the dominance of the bottom controller loop in figure 9 until the robot followed the path while maintaining the proper heading (figure 12).

![Diagram of steering control](image3)

**Figure 12.** Steering controller of Robot A: $K_{ap} = 200$

**Robot Coordination (Steering Control)**

After tuning the robot controllers individually, the robots were commanded to follow the generated trajectory and dock with one another as a fundamental step toward full collaborative control. As a sample collaborative scenario, figure 13 shows Robot A’s approach from the right and Robot B from the left and docking. The two robots are considered docked when they are within a desired distance offset of 40mm.

![Diagram of collaborative docking](image4)

**Figure 13.** Collaborative docking of Robots A and B along the poly-bisect trajectory

The combined controller subject to various trajectories exhibiting a scope of $\rho_{max}$ was found to be acceptable over a
range of radii of curvature in the trajectory. The tabulated results of the implemented controllers are given in table 2.

<table>
<thead>
<tr>
<th></th>
<th>Robot A</th>
<th>Robot B</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K_{p(x,y)}$</td>
<td>-1300</td>
<td>1800</td>
</tr>
<tr>
<td>$K_{d(x,y)}$</td>
<td>-600</td>
<td>400</td>
</tr>
<tr>
<td>$k_{pspeed}$</td>
<td>200</td>
<td>-10</td>
</tr>
<tr>
<td>$K_{pspeed}$</td>
<td>1100</td>
<td>-3000</td>
</tr>
<tr>
<td>$K_{dspeed}$</td>
<td>20</td>
<td>-30</td>
</tr>
<tr>
<td>Maximum deviation (mm)</td>
<td>21</td>
<td>12</td>
</tr>
<tr>
<td>Final Position Error (mm)</td>
<td>5.8</td>
<td>6.6</td>
</tr>
<tr>
<td>Final Heading Error (rad)</td>
<td>0.06</td>
<td>0.04</td>
</tr>
</tbody>
</table>

CONCLUSIONS

A framework for creating a trajectory via five constraints applied to a fourth order polynomial has been presented and implemented. Using classical control schemes for tracking control, the algorithm was proven. Both the speed and steering controls required delicate tuning of gains for accurate responses. Various experiments were performed while modifying the initial poses of the vehicles. Tuning methods were used as opposed to a model-based approach because the inaccuracies of the steering mechanism and drive gearbox would have negated the advantages achieved by the dynamical model obtained. The results are promising but could be improved if other control methodologies were used (e.g. fuzzy control).

ACKNOWLEDGEMENTS

This work has been funded in part through NASA Cooperative Agreement NCC3-994, the “Institute for Future Space Transport” University Research, Engineering and Technology Institute (URETI). The authors also acknowledge Tracy McSheery and Phasespace Inc. for their interest and contributions of the position equipment.

NOMENCLATURE

\(x_A\) = the initial x position of Robot A in the E frame coordinates  
\(\psi_{A0}\) = The initial heading of Robot A  
\(y_A\) = the y position of Robot A in the E frame coordinates  
\(y_A'\) = the heading of Robot A in the E frame coordinates  
\(k_{pspeed}\) = proportional control constant for the error from current desired position  
\(k_{dspeed}\) = derivative control constant for the change in error of position  
\((X(t),Y(t))\) = the desired position of the robot, which changes with time  
\((x,y)\) = the vehicles current position in the E frame  
\(\phi_c\) = center angle of the steering  
\(\phi\) = steering angle measured via PWM  
\(dt\) = the change in time between measurements (not considered constant)  
\(k_{p(x,y)}\) = control constant for the heading and vehicle position  
\(k_{d(x,y)}\) = control constant for the heading and vehicle position  
\((\theta_A,R_A)\) = position of Robot A in cylindrical coordinates  
\(\psi_A\) = the heading of Robot A in the E frame  
\(\hat{\dot{a}}\) = the directional vector of Robot A  
\(\hat{\dot{b}}\) = the directional vector of Robot B  
\((\hat{e}_1,\hat{e}_2)\) = the first and second axes of the E coordinate frame  
\(k_{bpspeed}\), \(k_{bdspeed}\) define the proportional and derivative control gains for the speed of Robot B, respectively  
\(k_{ap(x,y)}\), \(k_{ad(x,y)}\) define the proportional and derivative control gains for the position of Robot A, respectively  
\(k_{ap}\) defines the proportional control gain for the heading of Robot A

REFERENCES


