Trajectory Tracking Control and Behavior Path Planning on Mobile Robot
Introduction
Motivation

• **Trajectory Tracking Control**
  - The most basic part of mobile robot motion control
  - An efficient solution for transportation in the manufacturing industry

• **Behavior Path Planning**
  - A key technology that reflects mobile robot’s intelligence level
  - The most important tool for exploration in the unknown environment
Challenges

- **Trajectory Tracking Control**
  - A mobile robot is an nonlinear dynamic MIMO system
  - Unknown disturbance and nonholonomic constraints

- **Behavior Path Planning**
  - GPPs are just used in known environment
  - LPPs are difficult to describe the complex environment as a model
AmigoBot mobile robot
Trajectory Tracking Control
Plan

- The methods I used to design tracking controller
- Kinematic controller design
  - Kinematic model
  - Tracking control diagram
  - Sliding-Mode Controller based on Backstepping
- Dynamic controller design
  - Dynamic model
  - Tracking control diagram
  - Adaptive Dynamic Controller
The methods I used to design tracking controller

- Modern Control Methods
  - Sliding Mode controller based on Backstepping
    1. Backstepping controller
    2. Sliding Mode controller
  - Adaptive Dynamic Controller
    1. Kinematic controller—Lyapunov controller
    2. Adaptive controller

- Model-based Tracking Controller
  - Kinematic Model of Mobile robot
    - Kinematic Controller
    - Dynamic Controller
  - Dynamic Model of Mobile robot
Plan

• The methods I used to design tracking controller
  
• Kinematic controller design
  • Kinematic model
  • Tracking control diagram
  • Sliding-Mode Controller based on Backstepping

• Dynamic controller design
  • Dynamic model
  • Tracking control diagram
  • Adaptive Dynamic Controller
Kinematic model
• **Kinematic Model**

\[
\dot{q} = \begin{bmatrix}
\dot{x}_c \\
\dot{y}_c \\
\dot{\theta}_c
\end{bmatrix} = \begin{bmatrix}
\cos \theta_c & 0 \\
\sin \theta_c & 0 \\
0 & 1
\end{bmatrix} \begin{bmatrix}
v_c \\
w_c
\end{bmatrix}
\]

• If robot satisfies the conditions of pure rolling and no slipping in wheels, the **Constraint Equation** is as follows:

\[
\dot{y}_c \cos \theta_c - \dot{x}_c \sin \theta_c = 0
\]
Kinematic model, Cont'd

Target mobile robot

Reference trajectory

Real mobile robot
Kinematic model, Cont'd

• **Error Posture** in Local Coordinate System

\[
\begin{bmatrix}
  x_e \\
  y_e \\
  \theta_e
\end{bmatrix} =
\begin{bmatrix}
  \cos \theta_c & \sin \theta_c & 0 \\
  -\sin \theta_c & \cos \theta_c & 0 \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x_r - x_c \\
  y_r - y_c \\
  \theta_r - \theta_c
\end{bmatrix}
\]

• **Tracking error model** is obtained as follows

\[
\begin{bmatrix}
  \dot{x}_e \\
  \dot{y}_e \\
  \dot{\theta}_e
\end{bmatrix} =
\begin{bmatrix}
  y_e w_c - v_c + v_r \cos \theta_e \\
  -x_e w_c + v_r \sin \theta_e \\
  w_r - w_c
\end{bmatrix}
\]
Tracking control diagram

Tracking problem: find $v$ and $w$, when $t \to \infty$, $x_e \to 0, y_e \to 0, \theta_e \to 0$
SMC based on backstepping

- Backstepping Controller

1. Lyapunov Function (Partial)
   \[ V_y = \frac{1}{2} y_e^2 \]
   \[ y_e \to 0 \]

2. Problem of controller design
   \[ x_e \to 0 \quad \bar{\theta}_e = \theta_e + \varphi(y_e, v_r) \to 0 \]

   Lyapunov Function (Whole)
   \[ V = \frac{1}{2} (x_e^2 + y_e^2) + 2(1 - \cos \frac{\varphi}{2}) \]

   \[
   \begin{bmatrix}
   v \\
   w
   \end{bmatrix}
   =
   \begin{bmatrix}
   v, \cos \theta_e - wv_r \frac{d\varphi}{dz} \sin \frac{\bar{\theta}_e}{2} + k_i x_e \\
   w_r + v_r \frac{d\varphi}{dz} \sin \theta_e + \frac{d\varphi}{dz} y_e v_r + 2v_r y_e \cos \frac{\bar{\theta}_e}{2} - \varphi + k_z \sin \frac{\bar{\theta}_e}{2}
   \end{bmatrix}
   \]

\[ \varphi \in \Psi_\varepsilon^\infty = \{ \varphi : R \to (-\pi + \varepsilon, \pi - \varepsilon) : \varphi \in C^\infty, \varphi(0) = 0, z\varphi(z) > 0 \quad \forall z \neq 0 \text{and } \varphi' \text{ is bounded} \]
SMC based on backstepping, Cont'd

- Sliding Mode Controller
  - Sliding mode control is a popular nonlinear control method
  - It is difficult to select sliding surface that makes good performance

### Sliding Surface using Backstepping

\[
s = \begin{bmatrix} s1 \\ s2 \end{bmatrix} = \begin{bmatrix} x_e \\ \theta_e \end{bmatrix}
\]

\[
(s\dot{s} < 0)
\]

### Exponential Approach Law

\[
\dot{s} = \begin{bmatrix} \dot{s}1 \\ \dot{s}2 \end{bmatrix} = \begin{bmatrix} -k_1s_1 - \delta_1 \text{sgn}(s_1) \\ -k_2s_2 - \delta_2 \text{sgn}(s_2) \end{bmatrix}
\]

### Control Rule

\[
\begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} wy_e + v_r \cos \theta_e + k_1s_1 + \delta_1 \text{sgn}(s_1) \\ w_r + v_r^2 \frac{d\phi}{dz} \sin \theta_e + \frac{d\phi}{dz} y_e \dot{\nu}_r + k_2s_2 + \delta_2 \text{sgn}(s_2) \end{bmatrix}
\]

\[1 + x_e v_r \frac{d\phi}{dz}\]
Plan

• The methods I used to design tracking controller

• Kinematic controller design
  • Kinematic model
  • Tracking control diagram
  • Sliding-Mode Controller based on Backstepping

• Dynamic controller design
  • Dynamic model
  • Tracking control diagram
  • Adaptive Dynamic Controller
Dynamic model

- Dynamic model based on **torque vector control**

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{\theta} \\
\dot{v} \\
\dot{w}
\end{bmatrix} = \begin{bmatrix}
v \cos \theta \\
v \sin \theta \\
w \\
0 \\
0
\end{bmatrix} + \begin{bmatrix}
0 & 0 \\
0 & 0 \\
1/mr & 0 \\
0 & 1/rr
\end{bmatrix} \begin{bmatrix}
\tau_v \\
\tau_w
\end{bmatrix} + \begin{bmatrix}
0 \\
0 \\
\delta_v \\
\delta_w
\end{bmatrix}
\]

- Dynamic model based on **velocity vector control**

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{\theta} \\
\dot{v} \\
\dot{w}
\end{bmatrix} = \begin{bmatrix}
v \cos \theta \\
v \sin \theta \\
w \\
0 \\
0
\end{bmatrix} + \begin{bmatrix}
0 & 0 \\
0 & 0 \\
1/\theta_4 & 0 \\
0 & 1/\theta_1
\end{bmatrix} \begin{bmatrix}
v_{ref} \\
w_{ref}
\end{bmatrix} + \begin{bmatrix}
0 \\
0 \\
\delta_v \\
\delta_w
\end{bmatrix}
\]
1. Find $v_{ref}^c$ and $w_{ref}^c$, when $t \to \infty$, $x_e \to 0$, $y_e \to 0$, $\theta_e \to 0$

2. Find $v_{ref}$ and $w_{ref}$, when $t \to \infty$, $v \to v_{ref}^c$, $w \to w_{ref}^c$
Tracking control diagram, Cont'd
Adaptive Dynamic Controller

Lyapunov Controller (KC)

\[
\begin{bmatrix}
    v_{ref}^c \\
    w_{ref}^c
\end{bmatrix} = 
\begin{bmatrix}
    v_e \cos \theta_e + k_1 x_e \\
    w_e + v_e (k_2 y_e + k_3 \sin \theta_e)
\end{bmatrix}
\]

Adaptive Dynamic Control (DC)

\[
\begin{align*}
    v_{ref} &= x_e + \hat{\theta}_1 \dot{v}_{ref}^c + \hat{\theta}_4 v_{ref}^c + k_v \tanh(\delta \psi_1) \\
    w_{ref} &= \sin \theta_e / k_2 + \hat{\theta}_2 \dot{w}_{ref}^c + \hat{\theta}_6 w_{ref}^c + k_v \tanh(\delta \psi_2) \\
    \dot{\psi}_1 &= -\beta k \psi_1 + k z_1 \\
    \dot{\psi}_2 &= -\beta k \psi_2 + k z_2 \\
    \dot{\theta}_1 &= r_1 \dot{v}_{ref}^c z_1, \\
    \dot{\theta}_2 &= r_2 \dot{w}_{ref}^c z_2, \\
    \dot{\theta}_4 &= r_4 v_{ref}^c z_1, \\
    \dot{\theta}_6 &= r_6 w_{ref}^c z_2
\end{align*}
\]
Behavior Path Planning
Plan

- The methods I used to design path planner
- Behavior Planner based on Priority Arbitration
  - The diagram of Behavior Priority
  - Obstacle Avoiding Behavior Design Idea
  - Target Steering Behavior Design Idea
- Behavior Planner based on Fuzzy Fusion
  - Fuzzy Controller
  - Obstacle Avoiding Behavior Design
  - Target Steering Behavior Design
  - Behaviors Weight Controller Design
  - Behavior Fusion Design
The methods I used to design path planner

Task

Path planner

Tracking controller

Mobile Robot System

Behavior Path Planner

behavior planner based on priority arbitration

behavior planner based on fuzzy fusion

Task: going to a given goal in unknown environment

1 Obstacle Avoiding Behavior Design

2 Target Steering Behavior Design
Plan

• The methods I used to design path planner

• Behavior Planner based on Priority Arbitration
  • The diagram of Behavior Priority
  • Obstacle Avoiding Behavior Design Idea
  • Target Steering Behavior Design Idea

• Behavior Planner based on Fuzzy Fusion
  • Fuzzy Controller
  • Obstacle Avoiding Behavior Design
  • Target Steering Behavior Design
  • Behaviors Weight Controller Design
  • Behavior Fusion Design
Behavior Planner based on Priority Arbitration

• Obstacle Avoiding Behavior Design Idea
  • Setting a safe distance $Do$
  • When distance between robot and obstacle is less than $Do$, the robot will adjust its orientation.

• Target Steering Behavior Design Idea
  • According to the target position and current location, the robot will compute the relative distance and direction between them.
  • According to the results, it will move to the target at a given speed.

• The diagram of behavior priority

![Diagram of Behavior Priority]

- Obstacle Avoiding Behavior
- Target Steering Behavior
- Control signal
- S
Plan

• The methods I used to design path planner

• Behavior Planner based on Priority Arbitration
  • The diagram of Behavior Priority
  • Obstacle Avoiding Behavior Design Idea
  • Target Steering Behavior Design Idea

• Behavior Planner based on Fuzzy Fusion
  • Fuzzy Controller
  • Obstacle Avoiding Behavior Design
  • Target Steering Behavior Design
  • Behaviors Weight Controller Design
  • Behavior Fusion Design
1. **Fuzzification**: converts inputs into fuzzy set to activate rules.

2. **Rule-base**: the expert’s description of how to achieve good control.

3. **Inference mechanism**: interpret knowledge about the rules-mamdani fuzzy model

4. **Defuzzification**: converts the conclusions into actual inputs—“center of gravity” (COG)
Obstacle Avoiding Behavior Design

• Three group sonar sensors

\[ D_i = \min(D_{1i}, D_{2i}), \quad D_{1i} = \min(D_1, D_2), \quad D_{3i} = \min(D_3, D_4), \quad D_{5i} = \min(D_5, D_6) \]

Di: distance between the robot and obstacle by using the ith sonar sensor

• Fuzzification

  • Input variables: DL, DF, DR
  • Output variables: Vo, Wo
Target Steering Behavior Design

• The relative location of robot, target and obstacle

\( \alpha \) - robot heading angle
\( \theta \) - robot steering angle
\( \gamma \) - angle of Obstacle
\( \beta \) - angle of Goal

• Fuzzification

• Input variables: \( \theta \), \( Dt \)--distance between robot and target

• Output variables: \( Vt \), \( Wt \)
Behaviors Weight Controller Design

• Fuzzification

  • Input variables:

    D_{min} - \text{min} (D_1, D_2, D_3, D_4, D_5, D_6)

    Dot - minimum distance to the obstacle located along the relative target

  • Output variables: \( \lambda \)

• Linear velocity fusion: \( V = \lambda V_t + (1-\lambda)V_0 \)

• Angular velocity fusion: \( W = \lambda W_t + (1-\lambda)W_0 \)
Behavior Fusion Design

- Mobile Robot
  - Obstacle avoiding fuzzy controller
    - $V_0$, $W_0$
  - Behaviors weight fuzzy controller
    - $\lambda$
  - Target steering fuzzy controller
    - $V_t$, $W_t$

- Behavior fusion
  - $V$, $W$

- $D_L$, $D_F$, $D_R$, $D_t$, $\theta$
Conclusion
Conclusion

- **Trajectory Tracking**
  - Track the complex desired trajectory *rapidly, stably, precisely and robustly*.

- **Behavior Path Planning**
  - Improve the *adaptability of robot* in the unknown environment
  - Increase mobile robot’s *planning awareness*.
Reference


Thank you!