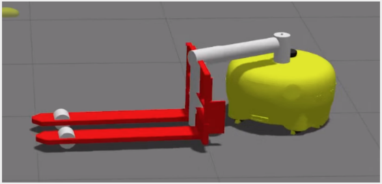
Autonomous Mobile Robotics

Motion Planning and Control

Geesara Kulathunga

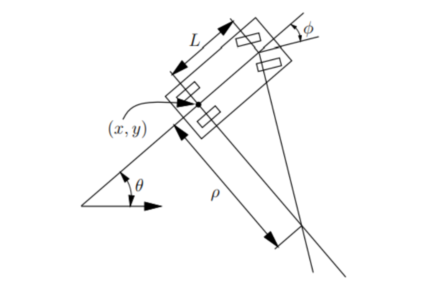


February 7, 2023

Control of Mobile Robots

Contents

- Kinematics of wheeled mobile robots: internal, external, direct, and inverse
  - Differential drive kinematics
  - Bicycle drive kinematics
  - Rear-wheel bicycle drive kinematics
  - Car(Ackermann) drive kinematics
- Wheel kinematics constraints: rolling contact and lateral slippage
- Wheeled Mobile System Control: pose and orientation
  - Control to reference pose
  - Control to reference pose via an intermediate point
  - Control to reference pose via an intermediate direction
  - Control by a straight line and a circular arc
  - Reference path control
- Lateral control (Geometric control)
  - The pure pursuit (or pure tracking controller)
  - Stanley controller
- Dubins path planning



- If the speed v and steering angle ϕ are directly specified by the control inputs u_s and u_ϕ, respectively, transition equation for simple car is

$$\begin{aligned} \dot{x} &= u_s \cdot cos(\theta) \\ \dot{y} &= u_s \cdot sin(\theta) \\ \dot{\theta} &= \frac{u_s}{L} \cdot tan(u_\phi) \end{aligned} \tag{1}$$

- What steering angles are possible? $[-\pi/2, \pi/2]$. It was assumed that the car moves in the direction that the rear wheels are pointing. When $\phi = \pi/2$, the front wheel perpendicular to the rear wheels, then car has to rotate in place. In other words, $\dot{x} = \dot{y} = 0$ because because the center of the rear axle does not translate. This behaviour is usually not possible because the front wheels would collide with the front axle when turned to $\phi = \pi/2$. Thus,

$$|\phi| \leq \phi_{max}$$

- A simple car is moving slowly to safely neglect dynamics, let's assume $|u_s| \leq 1$, where $u_s \in \{-1, 0, 1\}$. However, $0 < u_s < 1$, in this case, car can not drive in reverse

Several interesting variations are possible

- **Tricycle:** $U = [-1, -1] \times [-\pi/2, \pi/2]$, Assuming front-wheel drive, the vehicle can rotate in place if $u_\theta = \pi/2$. This kind of motion can be obtained using unicycle model
- **Simple car:** $U = [-1, -1] \times [-\phi_{max}, \phi_{max}]$, by requiring that $|u_\phi| < \phi_{max} < \pi/2$, a car with minimum turning radius $\rho_{min} = \frac{L}{tan\phi_{max}}$ is obtained
- **Reeds-Sheepp Car:** Further restrict the speed of the car, i.e., $u_s = \{-1, 0, 1\}$ or a car with three gears: reverse, park, and forward.
- **Dubins Car:** After removing reverse speed $u_s = -1$ from a Reeds-Sheepp car, $u_s = \{0, 1\}$ as the only possible speeds

- If you are given start and target poses (s and e), how can you estimate the shortest-path problem with respect to Reeds-Sheepp Car or Dubins Car model?

A bounded-curvature shortest-path problem

- If you are given start and target poses (s and e), how can you estimate the shortest-path problem with respect to Reeds-Sheepp Car or Dubins Car model?
- **Objective**: to minimize the length of the curve as the car travels between s and e

A bounded-curvature shortest-path problem

- If you are given start and target poses (s and e), how can you estimate the shortest-path problem with respect to Reeds-Sheepp Car or Dubins Car model?
- **Objective**: to minimize the length of the curve as the car travels between s and e
- However, ρ_{min}, curvature has to be bounded

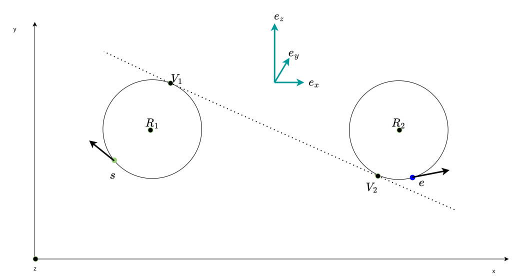
- C: circular arc with minimum turning radius
- $C_{>\pi}$: circular arc with minimum turning radius , angle $> \pi$
- S : straight-line segments

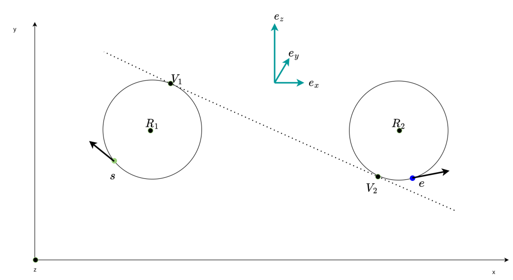
Dubins Path

Shortest paths are either CSC or $CC_{>\pi}C$

- C: R right turn, L left turn
- Dubins curves = {LRL, RLR, LSL, LSR, RSL, RSR}
- After specifying the duration of each primitive, Dubins curves = $\{L_\alpha R_\beta L_\gamma, R_\alpha L_\beta R_\gamma, L_\alpha S_d L_\gamma, L_\alpha SdR_\gamma, R_\alpha S_d L_\gamma, R_\alpha S_d R_\gamma\}$, where $\alpha \in [0, 2\pi), \beta \in (\pi, 2\pi)$, and $d \geq 0$.
- if $\beta < \pi$, there must be another path that is optimal

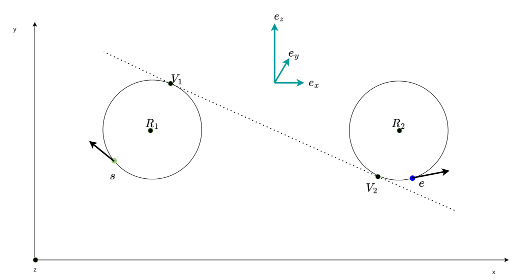
Dubins Path Planning (RSL)





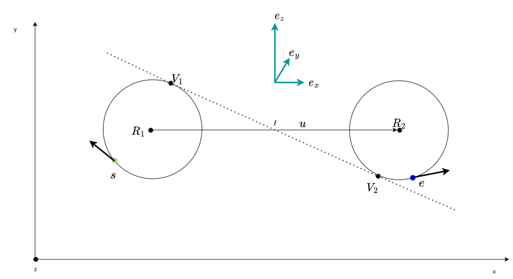
- $\mathbf{R}_1 = \mathbf{s} + r \cdot (\mathbf{e}_s \times \mathbf{e}_z)$

Dubins Path Planning (RSL)



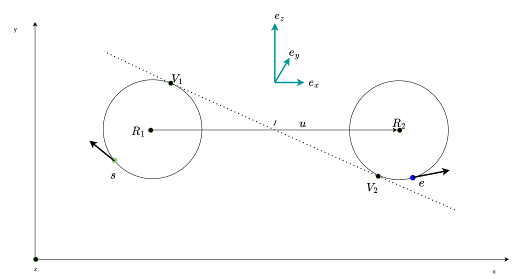
- $\mathbf{R}_1 = \mathbf{s} + r \cdot (\mathbf{e}_s \times \mathbf{e}_z)$
- $\mathbf{R}_2 = \mathbf{e} - r \cdot (\mathbf{e}_e \times \mathbf{e}_z)$

Dubins Path Planning (RSL)



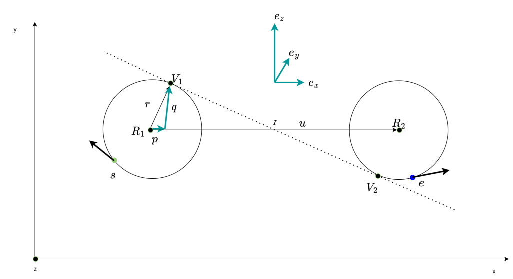
- $\mathbf{u} = \mathbf{R}_2 - \mathbf{R}_1$

Dubins Path Planning (RSL)



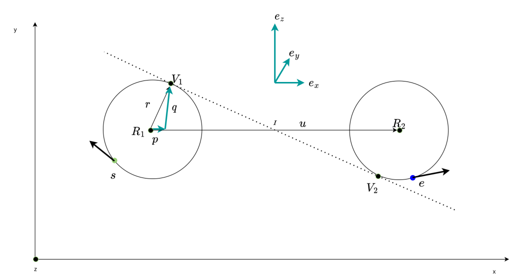
- $\mathbf{u} = \mathbf{R}_2 - \mathbf{R}_1$
- $\mathbf{I} = \mathbf{R}_1 + \frac{\mathbf{u}}{2}$

Dubins Path Planning (RSL)



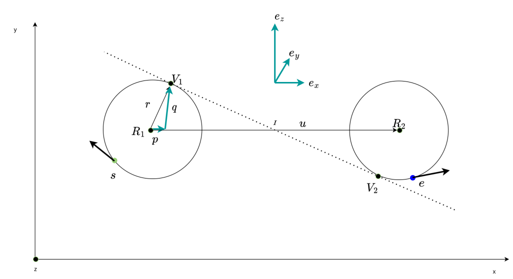
- $\alpha = arccos(\frac{2 \cdot r}{|\mathbf{u}|})$

Dubins Path Planning (RSL)

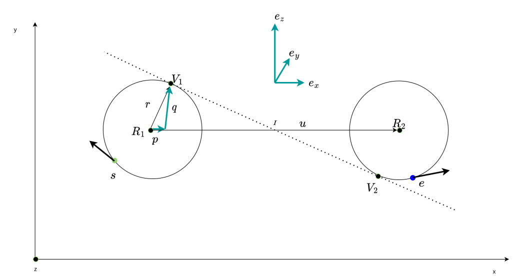


- $\alpha = arccos(\frac{2 \cdot r}{|\mathbf{u}|})$
- $\mathbf{p} = r \cdot cos(\alpha) \cdot \mathbf{e}_u$

Dubins Path Planning (RSL)

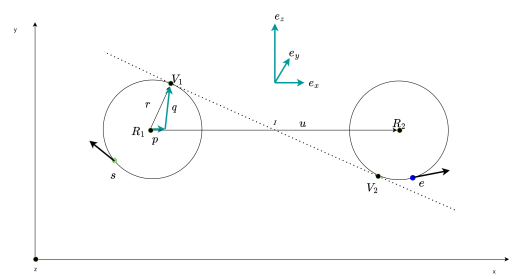


- $\alpha = arccos(\frac{2 \cdot r}{|\mathbf{u}|})$
- $\mathbf{p} = r \cdot cos(\alpha) \cdot \mathbf{e}_u$
- $\mathbf{q} = r \cdot sin(\alpha) \cdot (\mathbf{e}_z \times \mathbf{e}_u)$



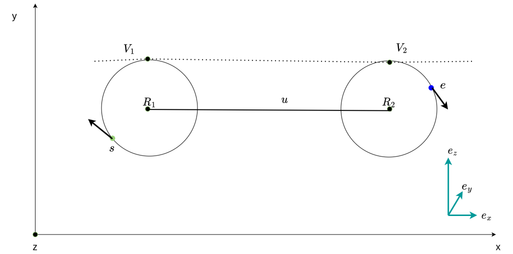
- $\alpha = arccos(\frac{2 \cdot r}{|\mathbf{u}|})$
- $\mathbf{p} = r \cdot cos(\alpha) \cdot \mathbf{e}_u$
- $\mathbf{q} = r \cdot sin(\alpha) \cdot (\mathbf{e}_z \times \mathbf{e}_u)$
- $\mathbf{V}_1 = \mathbf{R}_1 + \mathbf{p} + \mathbf{q}$

Dubins Path Planning (RSL)



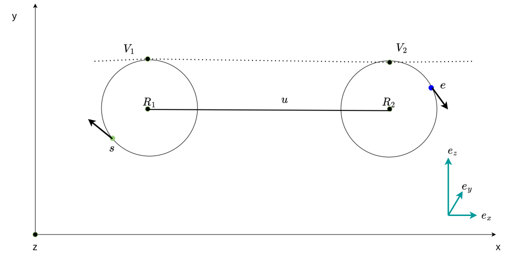
- $\alpha = arccos(\frac{2 \cdot r}{|\mathbf{u}|})$
- $\mathbf{p} = r \cdot cos(\alpha) \cdot \mathbf{e}_u$
- $\mathbf{q} = r \cdot sin(\alpha) \cdot (\mathbf{e}_z \times \mathbf{e}_u)$
- $\mathbf{V}_1 = \mathbf{R}_1 + \mathbf{p} + \mathbf{q}$
- $\mathbf{V}_2 = \mathbf{V}_1 + 2 \cdot (\mathbf{I} - \mathbf{V}_1)$

Dubins Path Planning (RSR)



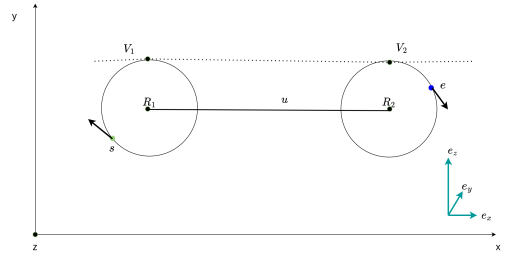
- $\mathbf{R}_1 = \mathbf{s} + r \cdot (\mathbf{e}_s \times \mathbf{e}_z)$

Dubins Path Planning (RSR)



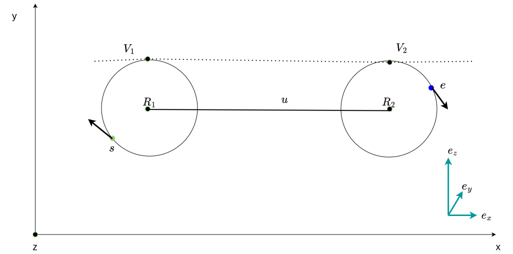
- $\mathbf{R}_1 = \mathbf{s} + r \cdot (\mathbf{e}_s \times \mathbf{e}_z)$
- $\mathbf{R}_2 = \mathbf{e} + r \cdot (\mathbf{e}_e \times \mathbf{e}_z)$

Dubins Path Planning (RSR)

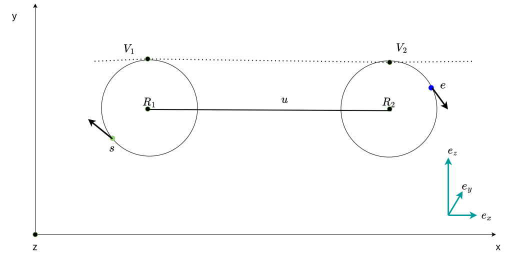


- $\mathbf{R}_1 = \mathbf{s} + r \cdot (\mathbf{e}_s \times \mathbf{e}_z)$
- $\mathbf{R}_2 = \mathbf{e} + r \cdot (\mathbf{e}_e \times \mathbf{e}_z)$
- $\mathbf{u} = \mathbf{R}_2 - \mathbf{R}_1$

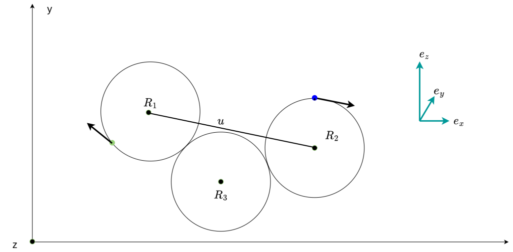
Dubins Path Planning (RSR)



- $\mathbf{R}_1 = \mathbf{s} + r \cdot (\mathbf{e}_s \times \mathbf{e}_z)$
- $\mathbf{R}_2 = \mathbf{e} + r \cdot (\mathbf{e}_e \times \mathbf{e}_z)$
- $\mathbf{u} = \mathbf{R}_2 - \mathbf{R}_1$
- $\mathbf{V}_1 = \mathbf{R}_1 + r \cdot (\mathbf{e}_z \times \mathbf{e}_u)$

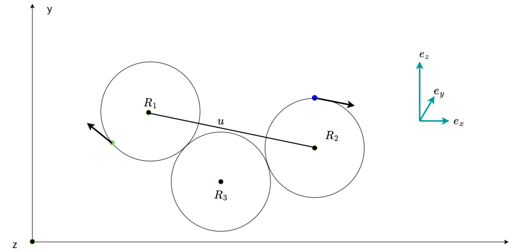


- $\mathbf{R}_1 = \mathbf{s} + r \cdot (\mathbf{e}_s \times \mathbf{e}_z)$
- $\mathbf{R}_2 = \mathbf{e} + r \cdot (\mathbf{e}_e \times \mathbf{e}_z)$
- $\mathbf{u} = \mathbf{R}_2 - \mathbf{R}_1$
- $\mathbf{V}_1 = \mathbf{R}_1 + r \cdot (\mathbf{e}_z \times \mathbf{e}_u)$
- $\mathbf{V}_2 = \mathbf{V}_1 + \mathbf{u}$



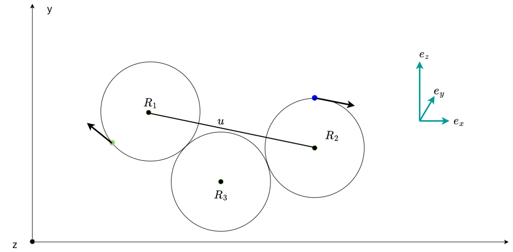
- $\mathbf{R}_1 = \mathbf{s} + r \cdot (\mathbf{e}_s \times \mathbf{e}_z)$

Dubins Path Planning (RLR)



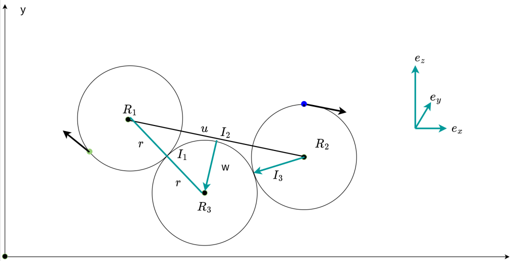
- $\mathbf{R}_1 = \mathbf{s} + r \cdot (\mathbf{e}_s \times \mathbf{e}_z)$
- $\mathbf{R}_2 = \mathbf{e} + r \cdot (\mathbf{e}_e \times \mathbf{e}_z)$

Dubins Path Planning (RLR)

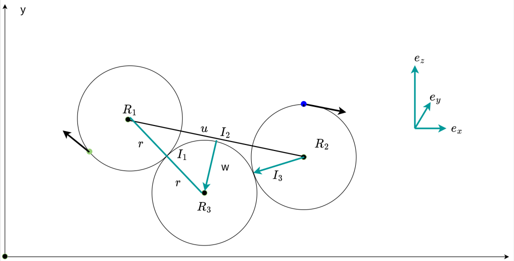


- $\mathbf{R}_1 = \mathbf{s} + r \cdot (\mathbf{e}_s \times \mathbf{e}_z)$
- $\mathbf{R}_2 = \mathbf{e} + r \cdot (\mathbf{e}_e \times \mathbf{e}_z)$
- $\mathbf{u} = \mathbf{R}_2 - \mathbf{R}_1$

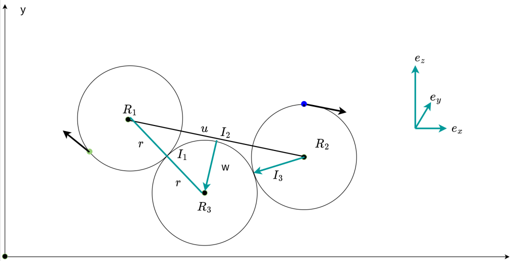
Dubins Path Planning (RLR)



- $\mathbf{I}_2 = \mathbf{R}_1 + \frac{\mathbf{u}}{2}$

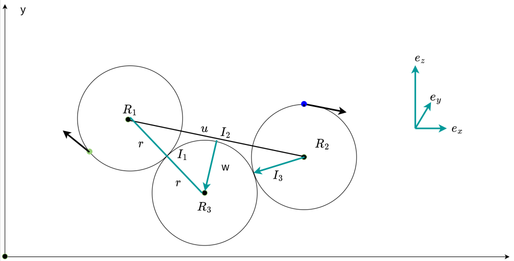


- $\mathbf{I}_2 = \mathbf{R}_1 + \frac{\mathbf{u}}{2}$
- $\mathbf{w} = \sqrt{4 \cdot r^2 - \frac{|u|^2}{4}} \cdot (\mathbf{e}_u \times \mathbf{e}_z)$

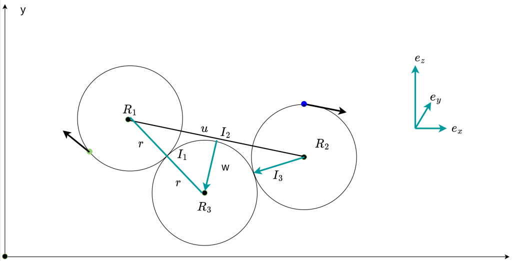


- $\mathbf{I}_2 = \mathbf{R}_1 + \frac{\mathbf{u}}{2}$
- $\mathbf{w} = \sqrt{4 \cdot r^2 - \frac{|u|^2}{4}} \cdot (\mathbf{e}_u \times \mathbf{e}_z)$
- $\mathbf{R}_3 = \mathbf{I}_2 + \mathbf{w}$

Dubins Path Planning (RLR)

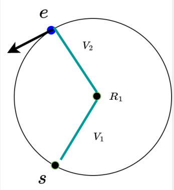


- $\mathbf{I}_2 = \mathbf{R}_1 + \frac{\mathbf{u}}{2}$
- $\mathbf{w} = \sqrt{4 \cdot r^2 - \frac{|u|^2}{4}} \cdot (\mathbf{e}_u \times \mathbf{e}_z)$
- $\mathbf{R}_3 = \mathbf{I}_2 + \mathbf{w}$
- $\mathbf{I}_3 = \mathbf{R}_2 + \frac{1}{2}(\mathbf{R}_3 - \mathbf{R}_2)$



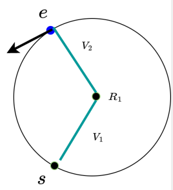
- $\mathbf{I}_2 = \mathbf{R}_1 + \frac{\mathbf{u}}{2}$
- $\mathbf{w} = \sqrt{4 \cdot r^2 - \frac{|u|^2}{4}} \cdot (\mathbf{e}_u \times \mathbf{e}_z)$
- $\mathbf{R}_3 = \mathbf{I}_2 + \mathbf{w}$
- $\mathbf{I}_3 = \mathbf{R}_2 + \frac{1}{2}(\mathbf{R}_3 - \mathbf{R}_2)$
- $\mathbf{I}_1 = \mathbf{R}_1 + \frac{1}{2}(\mathbf{R}_3 - \mathbf{R}_1)$

Computing Arc Length



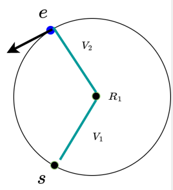
- arc length l = $r\theta$, where the angle between e and s is θ, and radius of circle is r

COMPUTING ARC LENGTH



- arc length l = $r\theta$, where the angle between *e* and *s* is θ, and radius of circle is r
- Let $\mathbf{v}_1 = \mathbf{e} - \mathbf{R}_1$, and $\mathbf{v}_2 = \mathbf{s} - \mathbf{R}_1$ be two vectors that connect start and target position with the center of the circle

Computing Arc Length



- arc length l = $r\theta$, where the angle between e and s is θ, and radius of circle is r
- Let $\mathbf{v}_1 = \mathbf{e} - \mathbf{R}_1$, and $\mathbf{v}_2 = \mathbf{s} - \mathbf{R}_1$ be two vectors that connect start and target position with the center of the circle
- Then depending on the direction of $\theta = atan2(\mathbf{v}_1) - atan2(\mathbf{v}_2)$, i.e, what direction that $\mathbf{v}_1$ rotates to end up at $\mathbf{v}_2$ direction of rotation can be defined: positive rotation is left turn and negative rotation is the right turn

Algorithm 1: Calculate arc length

Input: $\mathbf{v}_1, \mathbf{v}_2, r, d \in left, right$

Result: $|\theta * r|$

$\theta = atan2(\mathbf{v}_2) - atan2(\mathbf{v}_1)$

if $\theta < 0 \quad and \quad d = left$ **then**

  $\theta = \theta + 2\pi$

else

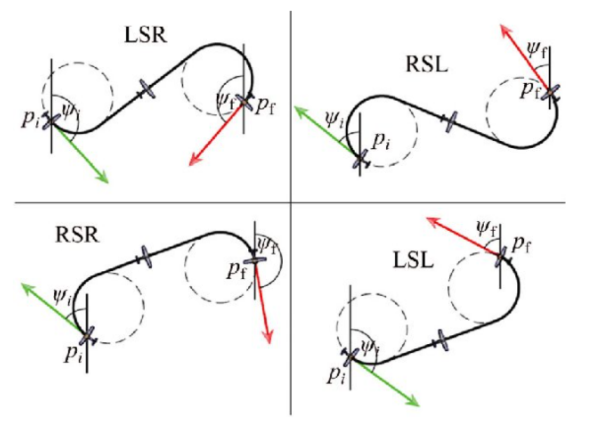
  if $\theta > 0 \quad and \quad d = right$ **then**

    $\theta = \theta - 2\pi$

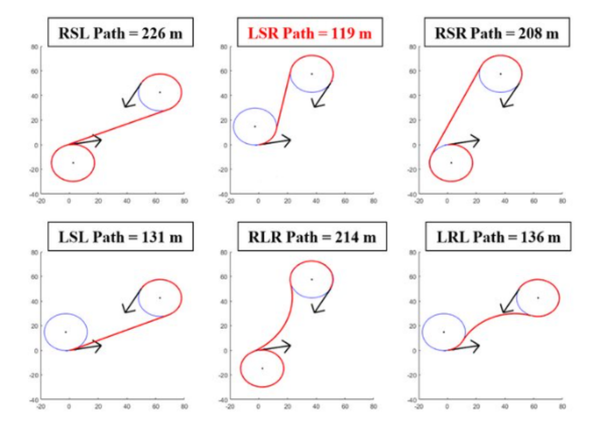
  end

end

Dubins Path Planning



Yardimci, A. G., Karpuz, C. (2019). Shortest path optimization of haul road design in underground mines using an evolutionary algorithm. Applied Soft Computing, 83, 105668.



Chen, Q. Y., Lu, Y. F., Jia, G. W., Li, Y., Zhu, B. J., Lin, J. C. (2018). Path planning for UAVs formation reconfiguration based on Dubins trajectory. Journal of Central South University, 25(11), 2664-2676.

References

Qing-yang Chen, Ya-fei Lu, Gao-wei Jia, Yue Li, Bing-jie Zhu, and Jun-can Lin.
Path planning for uavs formation reconfiguration based on dubins trajectory.
Journal of Central South University, 25(11):2664–2676, 2018.

Steven M LaValle.
Planning algorithms.
Cambridge university press, 2006.

Ahmet Gunes Yardimci and Celal Karpuz.
Shortest path optimization of haul road design in underground mines using an evolutionary algorithm.
Applied Soft Computing, 83:105668, 2019.