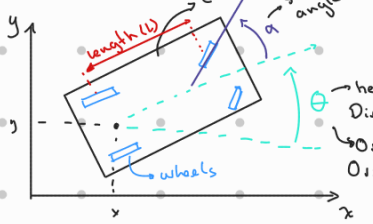
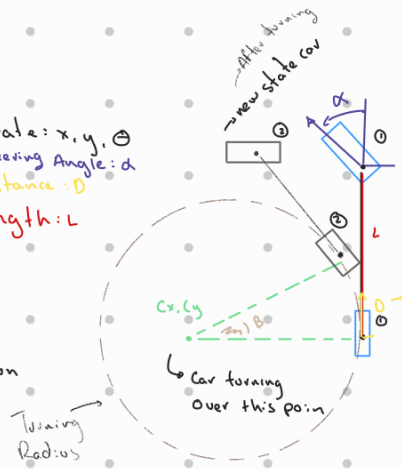
If you want to know the fastest way to a point, u use A*. You divide the map into many cells and use A* Algorithm.

→ What if we can't turn around 360°?
↳ like in an Autonomous car.
we use A* hybrid

Circular motion



state: x, y, θ
Steering Angle: α
Distance: D
Length: L



Turning Angle B:

$$B = \frac{D}{L} \cdot \tan(\alpha)$$

Turning Radius

$$R = \frac{D}{B}$$

* Before motion

$$C_x = x - \sin(\theta) \cdot R$$

$$C_y = y + \cos(\theta) \cdot R$$

New State:
* After motion

$$x' = C_x + \sin(\theta + B) \cdot R$$

$$y' = C_y - \cos(\theta + B) \cdot R$$

$$\theta' = (\theta + B) \bmod 2\pi$$

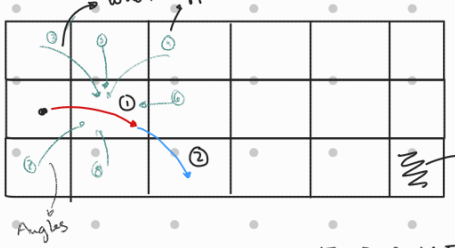
* For $|B| < 0.001$: → when almost straight line

$$x' = x + D \cdot \cos(\theta)$$

$$y' = y + D \cdot \sin(\theta)$$

$$\theta = \theta$$

A* Algorithm.



"branches" of the tree that collide with the optimal path, we eliminate them. Thus, we have less branches.

lack of Completeness: There can be paths that we don't take in consid. but we have correctness

↳ And Optimization

* WE MUST SIMULATE 3 ANGES:
  - Max Steer left
  - Forward.
  - Max Steer right

* Each cell must have 3 angles (Child node) (node)
  ↳ we have to calculate:
  - g(h-cost for each child node. → save steering angle in each node.
  - If Changing Steering Angle, Add another cost
  - If Obstacle, Add another cost.

* Node not valid if colliding with Obstacle or is outside map.
  ↳ not when we closed a cell. we close a cell when the car has visited that same cell with an specific Orientation.
    ↳ It means that the state of cell depends on both position/heading angle.

But, how can we find the lowest f-cost cell?
Using heap method.