

Module 6: Reactive Planning in Static Environments

A reactive motion planner is one that takes in local information from the robot's surroundings in order to **generate a trajectory** that is collision-free and makes progress towards some goal location.

- この定義だと、Reactive PlannerはLocal Plannerらしいけど。同じ? 多分同じ。

Lesson 1: Trajectory Propagation

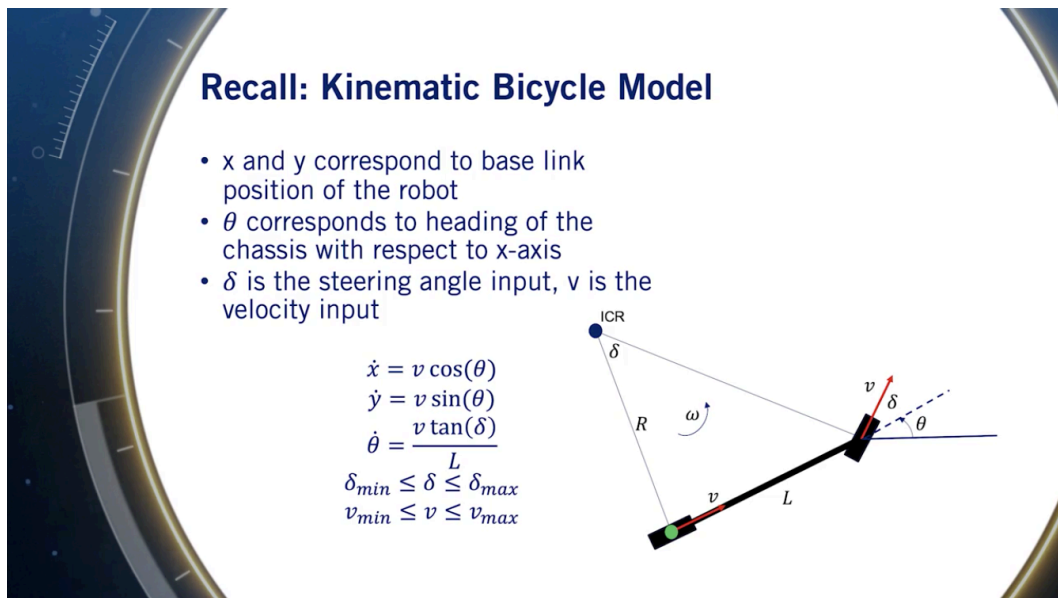
内容

- Difference between kinematic and dynamic motion models.
- Recall the bicycle model.
- Generate trajectories given control inputs and a motion model.

Kinematic vs. Dynamic Model

Particle Kinematic Model	Particle Dynamic Model
$\ddot{x} = a$	$M\ddot{x} + B\dot{x} = F$
Disregards mass and inertia of the robot.	Takes mass and inertia into consideration.
Uses linear and angular velocities (and/or derivatives) as input.	Uses forces and torques as inputs.

For path planning and trajectory optimization, often **focus on kinematic models** to make the motion planning problem more computationally tractable and **leave the issues raised by the simplification of the dynamics to the controller**.



Recall: Kinematic Bicycle Model

- x and y correspond to base link position of the robot
- θ corresponds to heading of the chassis with respect to x -axis
- δ is the steering angle input, v is the velocity input

$$\dot{x} = v \cos(\theta)$$
$$\dot{y} = v \sin(\theta)$$
$$\dot{\theta} = \frac{v \tan(\delta)}{L}$$
$$\delta_{min} \leq \delta \leq \delta_{max}$$
$$v_{min} \leq v \leq v_{max}$$

Recall: Kinematic Bicycle Model (大事)

- State: x, y, θ .
- Input: v, δ .
- We often do not have direct access to the state of the robot.
 - 上記の式から見ても、直接にアクセスできるのは $\dot{x}, \dot{y}, \dot{\theta}$ ですね。

- direct accessの意味はつまり積分はいらない。瞬時の変数。
- しかも車にとって、欲しいのは目的地の座標ではなく、そこに行けるthrottle, brake, steering sequenceだ!
- We **cannot tell the robot directly to go to a specific position in x and y.**
- We can, however, **devise a sequence of control inputs** that will allow us to reach said x, y position according to the kinematic equations.
- The sequence of control inputs will correspond to a trajectory that the robot will follow.

Kinematic Model Discretization

- Discretization of differential equations allows for efficient computation of trajectories.
- Recursive definition saves computation time.

$$x_n = \sum_{i=0}^{n-1} v_i \cos(\theta_i) \Delta t = x_{n-1} + v_{n-1} \cos(\theta_{n-1}) \Delta t.$$

$$y_n = \sum_{i=0}^{n-1} v_i \sin(\theta_i) \Delta t = y_{n-1} + v_{n-1} \sin(\theta_{n-1}) \Delta t.$$

$$\theta_n = \sum_{i=0}^{n-1} \frac{v_i \tan(\delta_i)}{L} \Delta t = \theta_{n-1} + \frac{v_{n-1} \tan(\delta_{n-1})}{L} \Delta t.$$

- For a given control sequence, we can compute the vehicle's trajectory.
- Also useful for prediction, where we know a kinematic model of a different agent in the driving scenario and we **have an educated guess on the control inputs they will take.**

Varying Input for Obstacle Avoidance

- To avoid obstacles, we require more complex maneuvers.
- We can vary the steering input according to a steering function to navigate complex scenarios.
- Local Planningの目標: Main objective of local planning is to compute the control inputs (or trajectory) required to navigate to goal point without collision. (大事)

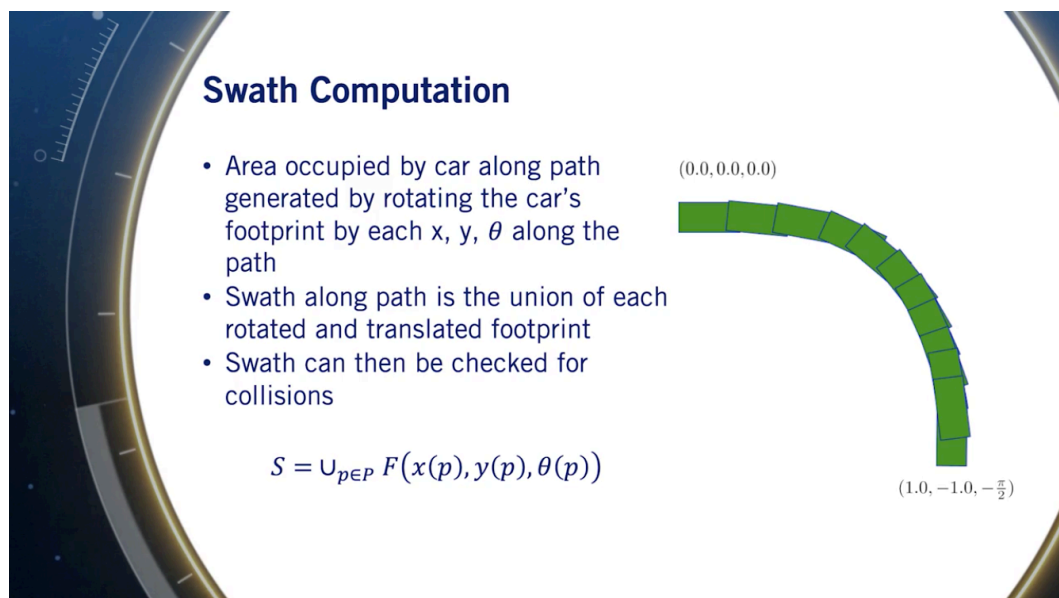
Lesson 2: Collision Checking

内容

- Challenges present in collision checking.
- How and when to use swath-based or circle-based collision checking.
- Some of the pitfalls posed by imperfect information and discretization errors, and how to mitigate them through conservative approximation.

Collision Checking Challenges

- Computationally intensive.



- Especially for complex shapes and detailed environments.
- Require perfect information to guarantee safety.
- Needs to be approximated, and must be robust to noise.
 - The information given to us by the occupancy grid is an imperfect estimate, which means that we need to add buffers to our collision checking algorithms to make them more error tolerant.

Swath Computation

- 明らかに疑われるところは車のfootprintsは離散だ。車速が速い時は隙間が出るでしょう。
 - Discretization Resolution.

Discretized Example

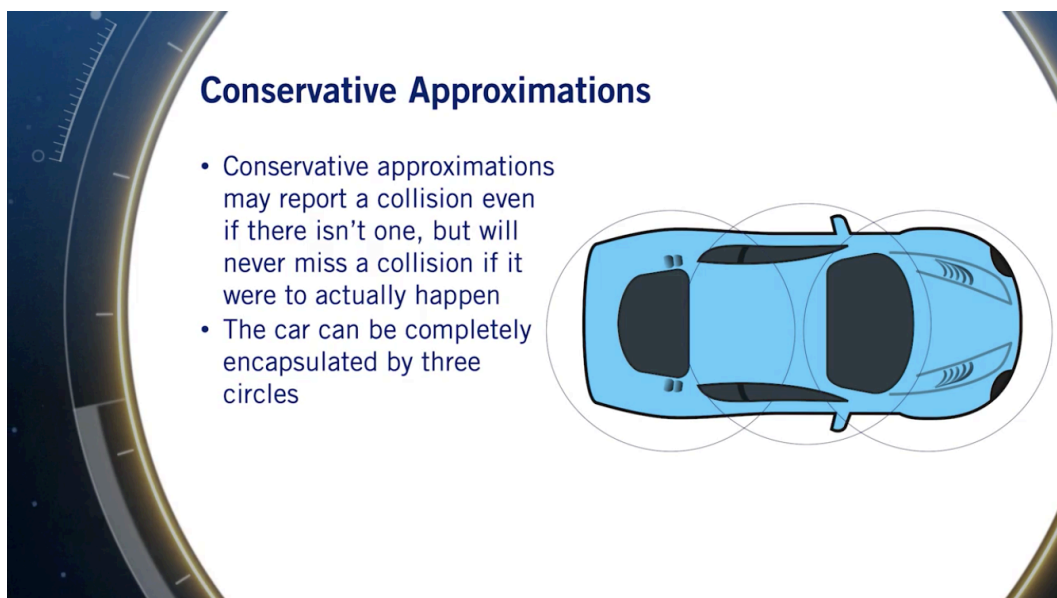
- Initial state of the vehicle in the occupancy grid, with **base link at the origin**.
 - 車は今3点囲んでる。(0,0), (1,0), (2,0).
 - 回転中心はbase linkだ。
- Will need to rotate and translate to get the new footprint at point $(1,2,\frac{\pi}{2})$.
- First, rotate the footprint about the origin by $\frac{\pi}{2}$.
 - そうすると、車が囲んでる3点(0,0), (0,1), (0,2)になる。
- Next, translate each point by (1,2).
 - 3点は(1,2), (1,3), (1,4)になる。
- 必ず先に回転する!
- 最後にOccupancy Grid Map座標系に変換。Image Coordinateへの変換と似てる。ただ occupancy grid mapの原点は左下。

Lattice Planner Swaths

- Swath based methods are useful for lattice planners, as the swath sets can be computed offline.
 - swath setsはcontrol setを適用した後の車のパスたち。Path Candidateだ!
- Online collision checking is then simplified using **lookup tables**.
- 復習Lattice Planner (Module 1, Lesson 4).
 - Local Plannerの一種。車両予測のやり方らしい。
 - Constrain the search space by limiting actions available to the robot (control set).
 - Layers of control actions form a graph, which can be searched using Dijkstra's or A*.
 - Conformal lattice planner fits the control actions to the road structure.

Speed and Robustness

- Need to improve speed.



- Need to be robust to noise.
- Use conservative approximations to solve both of these problems.
- Want **algorithmic speedup** without sacrificing path quality.

Conservative Approximations

- Conservative approximations may report a collision even if there isn't one, but will never miss a collision if it were to actually happen.
- The car can be completely encapsulated by three circles.

Circle Collision Checking

- Circle approximation is effective because it is fast to check if an occupancy grid point lies within a circle of radius r centered at (x_c, y_c) .
 - If obstacle in occupancy grid lies within circle, a collision is reported.
 - Otherwise, due to conservative approximation, no collision is possible.
- Conservative approximationの課題: They may eliminate all feasible collision-free paths from a problem even though a path exists or eliminate safe passages through narrow openings.
 - This can cause the planner to get stuck when it shouldn't or can cause the planner to compute a much more circuitous route than is necessary.

Discretization Resolution

- Collision checking accuracy is impacted by the resolution of our discretization.
- Higher fidelity collision checking requires a finer resolution for occupancy grids and path points, and will require more computational resources.

単語

- Prohibitive: If the cost of something is prohibitive, it is so high that many people cannot afford it.
- Amount to: If you say that one thing amounts to something else, you consider the first thing to be the same as the second thing.
- Circuitous: A circuitous route is long and complicated rather than simple and direct.

Performing quick and efficient collision checking especially with dynamic obstacles is currently an area of active research.

Lesson 3: Trajectory Rollout Algorithm (大事)

内容

- Implement the trajectory rollout algorithm.
 - Trajectory propagation.
 - Collision checking.
 - Path selection.
- Understand the concept of receding horizon planning.

Trajectory Rollout Planner

- Uses trajectory propagation to generate candidate set of trajectories.
- Among collision-free trajectories, select trajectory that makes the most progress to goal.
- By performing this repeatedly, we end up with a **receding horizon planner** that reacts to the environment while **making steady progress** towards the goal.

Trajectory Set Generation

- Each trajectory corresponds to a fixed control input to our model.
 - For multiple steps over a constant time horizon.
 - **multiple steps内ずっと同じcontrol input**を適用する。
 - ステップ毎にやるので、ステップとステップの適用するcontrol inputは違うかも。
 - Typically **uniformly sampled across range of possible inputs**.
- More sampled trajectories leads to more maneuverability.
- Fewer sampled trajectories improves computation time.
 - Each additional trajectory will need to be generated, checked for collisions, and scored.

Trajectory Propagation

- Holding the velocity constant and varying the steering angle gives candidate set of trajectories.
 - 例えば $[-\frac{\pi}{4}, \frac{\pi}{4}]$.
 - kinematic equationsを使う。
 - We now have a set of arcs as our candidate trajectories.

Swath Based Collision Checking

- Swath is generated for each candidate trajectory.
 - By **sweeping** the body of the robot along the path, and taking the union of all the footprints at each time step of a given trajectory.
- The footprint of the car will correspond to a set of indices in the occupancy grid.
 - So each rotated and translated point along the path will also correspond to different indices in the occupancy grid.
 - このindexはgridかgridを囲む4点か?
 - gridだったら、conservative approximationになりそう。でもlesson 2の例から見ると、そうじゃない。車が含まれているgridの頂点を使っているらしい。
- Collision checking is performed for each point in the swath using the occupancy grid.
 - $S = \cup_{p \in P} F(x(p), y(p), \theta(p))$.
 - Iterating through each point in the swath set and checking the **associated index in the occupancy grid for an obstacle**.
 - If any point in the swath is occupied, then that trajectory contains a collision.

Objective Function

- Rewarding progress to a goal point is the ultimate goal of motion planning.
 - $J = \|x_n - x_{goal}\|$.
 - distance to goal.
 - パスの最後の点だけ使っている?
- $J = \alpha_1 \|x_n - x_{goal}\| + \alpha_2 \sum_{i=1}^n \kappa_i^2 + \alpha_3 \sum_{i=1}^n \|x_i - P_{center}(x_i)\|$.
 - curvature penalty and deviation from center line.
- Sometimes, we also want to reward paths that **maximize the distance to the nearest obstacle**, to maximize the **flexibility** of feasible paths available to **future time steps** in the planner.

Example

- Conformal Lattice Plannerの例が欲しいな! この例はHD Mapを使っていない。
- Steering angle bounded by $|\delta| \leq \frac{\pi}{4}$.
 - $\frac{\pi}{8}$ step size.
 - constant velocity of 0.5m/s.
- Time discretization of 0.1s, with a 2s planning horizon.
- Planning cycle is shorter than trajectory length.
 - The vehicle will execute only the first few points of the cycle.
 - The exact number depends on the **planning frequency** and our planning horizon will be shifted forward depending on our progress.
- Only 1s of 2s trajectory is executed at each planning iteration.
 - receding horizon plannerのやり方.
 - Planning horizonのend timeがrecedes towards time point when goal is reached.
 - つまりパスの一部だけ実行。
- This planner is **greedy and sub-optimal**, but is fast enough to allow for online planning.

Lesson 4: Dynamic Windowing

内容

- How to add linear and angular acceleration constraints to the bicycle model.
- How these constraints impact our planner.
- Handle these constraints in the planning process using dynamic windowing.

Kinematic Bicycle Model

- No consideration of higher-order terms such as acceleration or jerk.
 - These higher-order terms are what cause discomfort for passengers in the car, so we should try to address this in our kinematic model.

Bicycle Model + Acceleration Constraints

- Higher order terms handled by adding constraints.
- More comfort for passengers, but less maneuverability.
- $\ddot{\theta}_{min} \leq \ddot{\theta} \leq \ddot{\theta}_{max}$.
- $\ddot{x}_{min} \leq \ddot{x} \leq \ddot{x}_{max}$.

Constraint in terms of Steering Angle

- Angular acceleration constraint may prevent us from selecting certain maneuvers based on current angular velocity.
- Change in steering angle between planning cycles is bounded.

$$\dot{\theta} = \frac{v \tan(\delta)}{L}.$$

$$|\ddot{\theta}| = \left| \frac{\dot{\theta}_2 - \dot{\theta}_1}{\Delta t} \right|.$$

$$\left| \tan(\delta_2) - \tan(\delta_1) \right| \leq \frac{\ddot{\theta}_{max} L \Delta t}{v}.$$

- Similar logic applies for changes in linear velocity inputs between planning cycles.