PlanningPreview

July 16, 2019

1 Overview

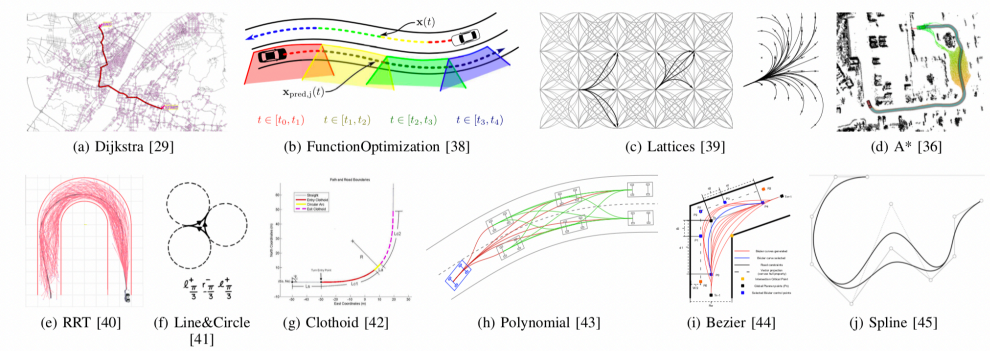
- What is the motion planning?
  - The essence of planning: $\underset{x}{argmin} f(x)$
  - In different fields
    - Robotic fields: generate trajectory to accomplish goals
      - RRT, A*, D* lite, Lattice Planning, and etc.
    - Control Theory: Dynamic system to reach an target state
      - MPC, LQR. and etc.
    - AI: generate state action mapppings
      - Reinforcement learning, end to end imitation learning and etc.
    - Motion planning for autonomous driving: Safely and smoothly maneuver autonomous driving vehicle to the destination
      - X, Y, Time: 3Dimensional Trajectory Optimization Problem

  From motion planning by Steve Lavelle: http://planning.cs.uiuc.edu/par1.pdf

1.1 Motion Planning Approaches

- Traditional robotic trajectory generation of autonomous driving:
  - Configuration space
  - RRT based approach: #TODO
  - Lattice approach
- Modern approaches in autonomous driving
  - Darpa Challenge Approaches
  - Graph search based method: Lattice in frenet Frame
  - Interpolating Curve: Spiral, Polynomial and Splines
  - Functional Optimization
- Example codes: https://github.com/AtsushiSakai/PythonRobotics

Gonzalez, David, et al. "A Review of Motion Planning Techniques for Automated Vehicles", 2016



(a) Dijkstra [29] (b) FunctionOptimization [38] (c) Lattices [39] (d) A* [36]

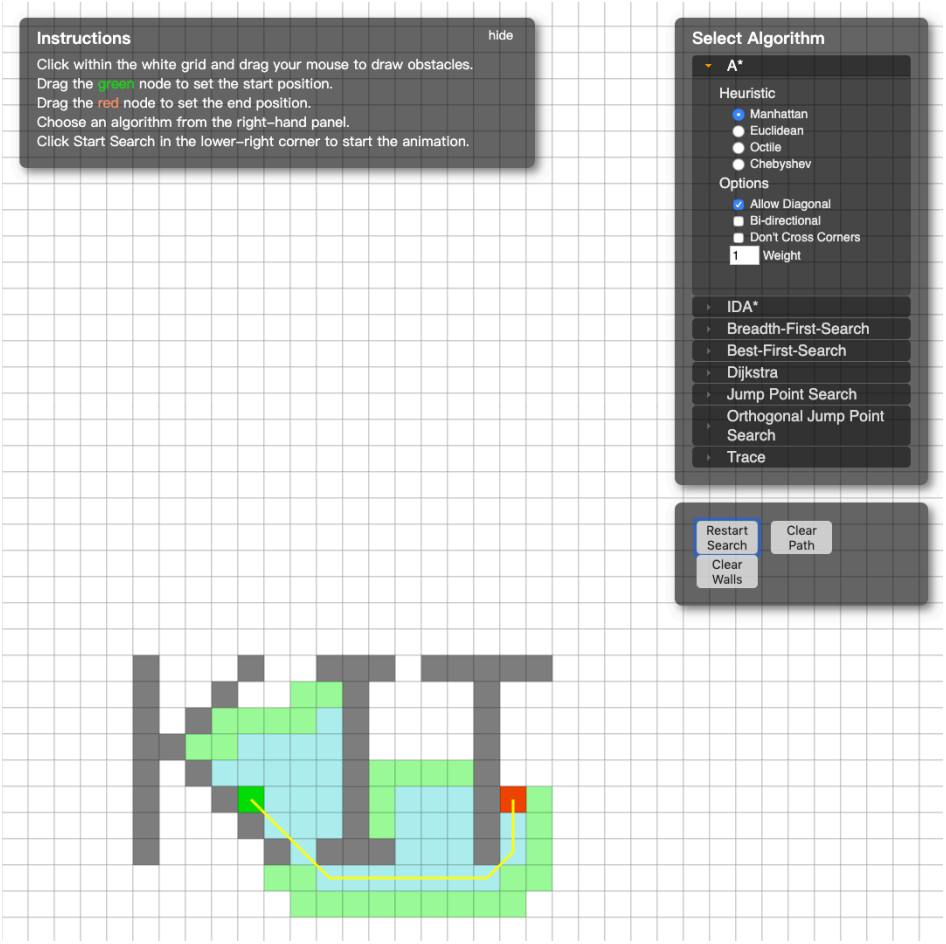
(e) RRT [40] (f) Line&Circle [41] (g) Clothoid [42] (h) Polynomial [43] (i) Bezier [44] (j) Spline [45]

Fig. 2. Planning algorithms as presented in the literature. a) Representation of a global path by the Dijkstra algorithm in [29]. b) Trajectory optimization from [38], taking into account a vehicle in the other lane. c) Lattices and motion primitives as presented in [39]. d) Hybrid A* as implemented in the DARPA Challenge by Junior [36]. e) RRT* as showed in [40]. f) Optimal path to turn the vehicle around as proven in [41]. g) Planning a turn for the Audi TTS from Stanford [42]. h) Different motion states, planned with polynomial curves as presented in [43]. i) Evaluation of several Bézier curve in a turn, as showed in [44]. j) Spline behavior when a knot changes place, as presented in [45].

Planning Approaches

1.2 How to solve a Motion Planning problem?

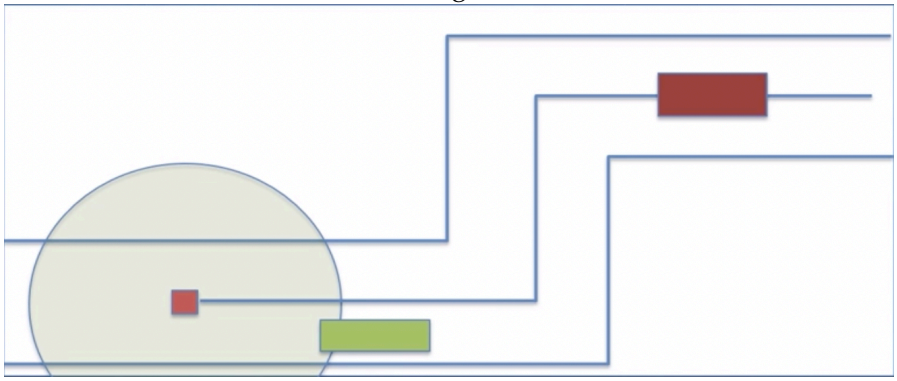
- Simplify the problem: Path Finding Problem
  - Ignore speed
  - Fixed environment
  - classic methods:
    - Non-Informative search methods
      - BFS & DFS
      - Dijkstra
    - Informative search methods
      - A*: Rank unsearched nodes based on combination of current cost plus heuristic cost
  - A simple shortest path example



copyright at: http://qiao.github.io/PathFinding.js/visual/

1.2.1 The challenge of motion planning in autonomous driving system

1. Partially Observed Dynamic Environment
    - when it observes new map information(such as previously unknown obstacles), it adds the information to its map and, if necessary, replans a new shortest path from its current coordinates to the given goal coordinates. It repeats the process until it reaches the goal coordinates or determines that the goal coordinates cannot be reached. - *From wikipedia*



- use incremental search to slove this: Partially observed information needs local planning and refinement
  - D*/D* lite: Search by partial environment(Algorithm used by Apollo moon landing car)
  - Unable to guarantee global optimal solution

Stentz Anthony,* "Optimal and Efficient Path Planning for Partially-Known Environments"*, 1994

2. How to find a reliable path under parially observed space?
   - Continuous representation -> Discretization -> Graph searching(blind, best-first, A*)
   - Roadmap Methods
     - Visibility graph: *Introduced in the Shakey project at SRI in the late 60s. Can produce shortest paths in 2-D configuration spaces*
     - Voronoi field
   - PRM
     - Construction Phase
       - Random Samples in configuration space
       - Nearest Neighbor Connections
     - Search Phase
       - Dijkstra or A*'s shortest path
   - RRT: An incremental search version of PRM
     - Build a tree T of configurations, starting at X_{start}
     - Extend
       - Sample a configuration X_{rand} from C at random
       - Find the node X_{near} in T that is closest to X_{rand}
       - Extend a short path from X_{near} toward X_{rand}
     - RRT's Problem
       - Path is not smooth enough for vehicle maneuver
       - Opimal trajectory is not the one with shortest path
       - Refinement: Incremental random search with consideration of vehicle dynamic
   - Lattice Sampling: Vehicle moving states are discretized into lattice
     - Pros: Suitable for on road motion planning
     - Cons: Discretizing both spatial and temporal parameters, search space grows exponentially

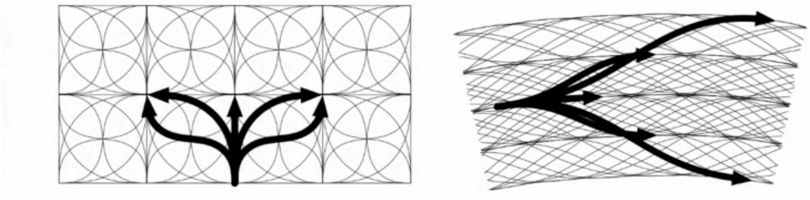
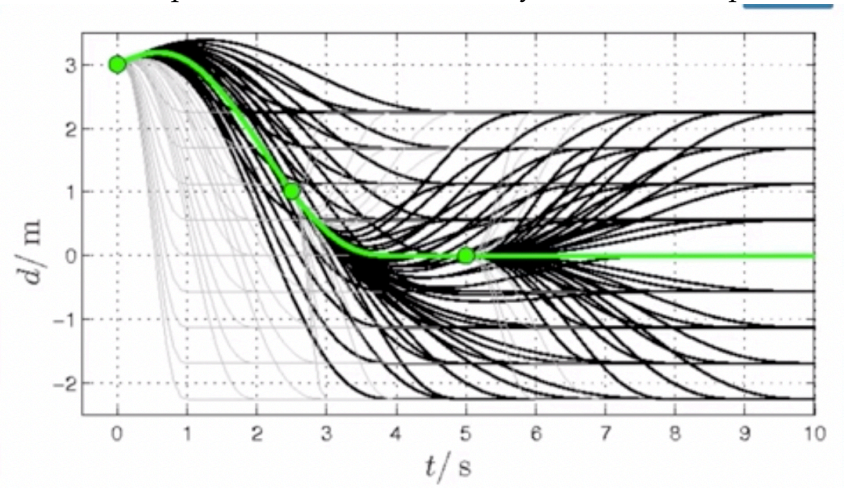


Fig. 1. Left: regular state lattice in an unstructured environment. The same five paths are rigidly transformed to create a graph of kinematically feasible actions. Right: state lattice conformed to a structured environment. Each path must be optimized to fit its endpoints.

McNaughton, Matthew, et al. "Motion planning for autonomous driving with a conformal spatiotemporal lattice." Robotics and Automation(ICRA). 2011 IEEE International Conference.

- Lattice with longitudinal and lateral decomposition
  - Pros: Search in 3D lattice space
  - Cons: Path Speed behavior is limited by the lattice shape in one planning cycle



Werling, Moritz, et al. "Optimal trajectory generation for dynamic street scenarios in a frenet frame." Robotic and Automation(ICRA), 2010 IEEE International Conference.

- Optimization in lattice - Path speed iterative methods

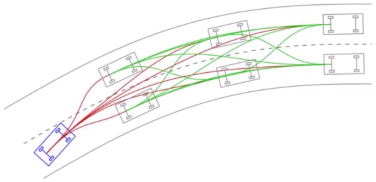


Fig. 2: Path set. The blue vehicle represents the current vehicle pose, and grey vehicles represent sampled endpoints. The red paths are quartic curvature polynomials and the green paths are cubic ones.

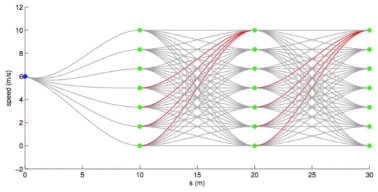
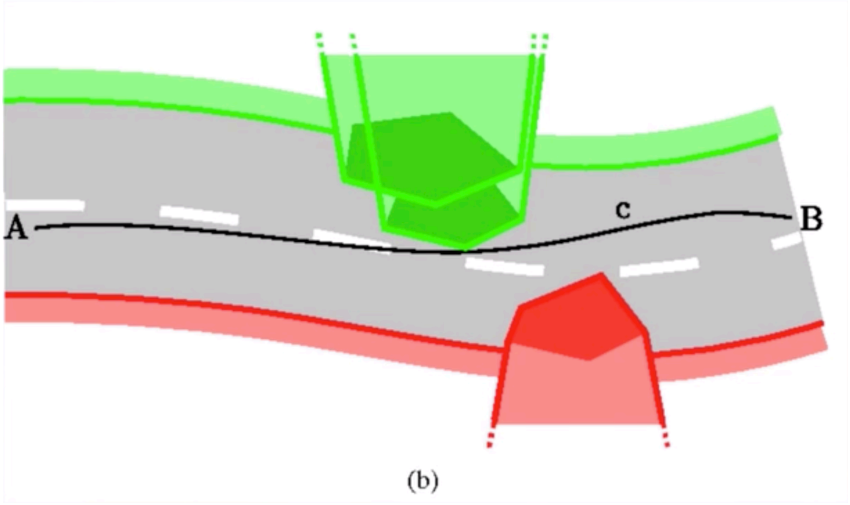


Fig. 4: Speed set. The green points are sampled speeds, the red curves are beyond the acceleration limit, and the grey curves are valid ones.

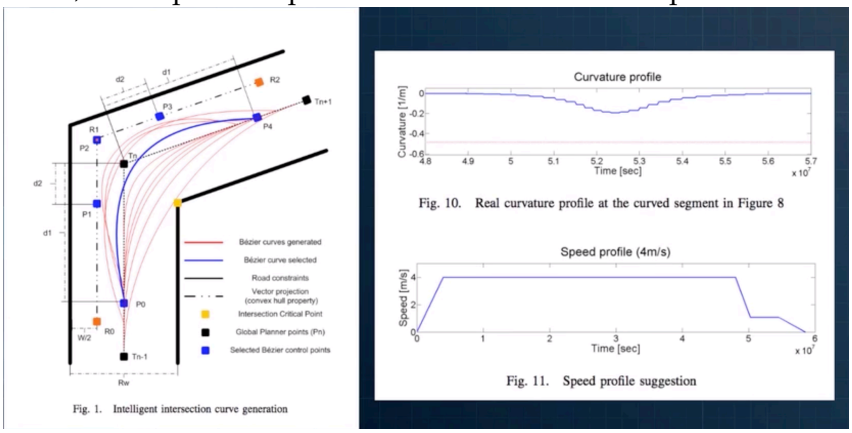
Xu. Webda. et al. "A real-time motion planner with trajectory optimization for autonomous vehicles." Robotic and Automation(ICRA). 2012 IEEE International Conference

- Quadratic Programming Approach
  - Pros: Finite differencing to model a trajectory, local continuity
  - Cons: Not smooth enough, need decision to find a tunnel



Ziegler, Julius, et al. "Trajectory planning for Bertha–A local, continuous method." Intelligent Vehicles Symposium Proceedings, 2014 IEEE

- Spline approaches
  - Pros: Use bezier curves to generate smooth path
  - Cons: need rule based decisions to-walk through obstacle; Speed profiles are too simple and not scalable



Bautista, David Gonzalez, et al. "Continuous curvature planning with obstacle avoidance capabilities in urban scenarios." 2014 IEEE.

3. Autonomous driving car cannot move like we planning on the graph
   - Types of Path constraints
     - Local constraints: *e.g., avoid collision with obstacles*
     - Differential constraints: *e.g., bounded curvature*
     - Global constraints: *e.g., combnation of smoothness and length*
4. How to construct the traffic regulations that our computer can understand?
   - **#TODO**
5. Real time calculation
   - Normal driver reaction time: 300~500 ms
   - Drunk driver: 1000 ms
   - Autonomous car planning module: **~100 ms**(using C++)

Motion planning in Autonomous Driving

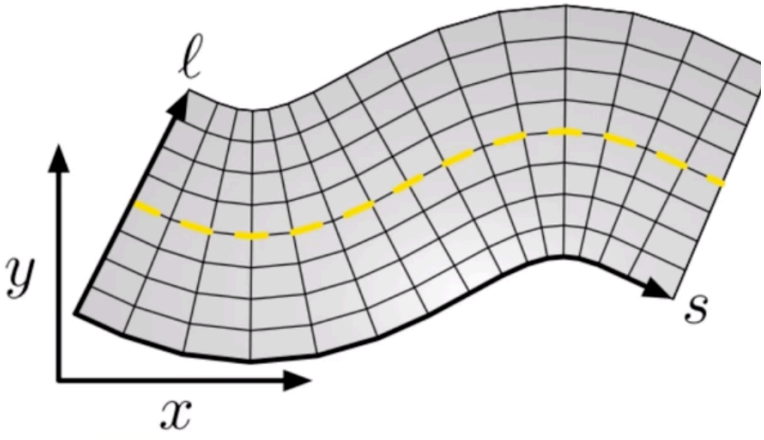
- Lattice search is simpler and more suitable for autonomous driving on **structured road**
- Searching on the 3 Dimensional Lattice is still time consuming; Path Speed iterative search vs. direct search both have pros and cons
- Dynamic programming based search method depends on the state dimension
- Quadratic programming based method need decision to determine a feasible convex tunnel
- Splines are more smooth than direct finite differencing and grid trajectory, but splines need decisions as well.

1.2.2 Motion planning with environment

- From point-mass model to bicycle model
- Understand the requirement from vehicle control perspective
- Smooth driving path and speed profile; importance performance index
- smooth SL to XY Frenet Frame mapping techniques
- Obstacle mapping

SL Coordinate Frame and XY Coordinate Frame

- Centerline of the road is used to construct SL frame
  - S: the direction along centerline
  - L: the direction orthogonal to the centerline
- SL Coordinate to XY Coordinate
  - $X = X_r - l * sin(\theta)$
  - $Y = Y_r + l * cos(\theta)$



  - $k = \frac{1}{1+\frac{1}{k_r}}$

Obstacle relations

- Bounding box: is the rectangle with minimal area that includes the obstavle polygons
- Hyperplane separation theorem

– Sufficient and necessary condition: if two convex polygons are separated, there exist a direction that is orthogonal to one of the polygon edges which can separate two convex polygons