Safe trajectory planning for single and multi agents

H. Piet-Lahanier\textsuperscript{1,*} helene.piet-lahanier at onera.fr

Sautos
- 24 October 2023 -

\textsuperscript{1}ONERA
Introduction

Safe Trajectory Planning

- Motivation
- Methods for single/multiple agent(s)
- Uncertainty
- Scalability?
Single autonomous vehicle or set of autonomous vehicles
Trajectory Planning

Motivation

1. Single autonomous vehicle or set of autonomous vehicles
2. Known initial locations
Trajectory Planning

Motivation

1. Single autonomous vehicle or set of autonomous vehicles
2. Known initial locations
3. Aiming to known target points
Trajectory Planning

Motivation

1. Single autonomous vehicle or set of autonomous vehicles
2. Known initial locations
3. Aiming to known target points
4. What is the trajectory (ies) to follow with the best performance criterion?
Trajectory Planning

Motivation: A simple situation

Diagram 1: Targeted zone for Vehicle 1 and Vehicle 2 with no obstacles.

Diagram 2: Targeted zone for Vehicle 1 and Vehicle 2 with Obstacle 1 and Obstacle 2.
Trajectory Planning
Motivation: A more complex situation
Trajectory Planning
Summarizing the issues

Issues in path planning with collision avoidance

- Producing feasible trajectories
- Accounting for the dynamics
- Complex environment
- Complex Maps/Terrains
- Finding shortest path or with cheapest cost
- Movable obstacles not initially known
- Accounting with multi...
A subject of interest

Numerous survey papers focusing on specific aspects (chronological order) for UAV domain

- A survey of motion planning algorithms from the perspective of autonomous UAV guidance [1]
- Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey [2]
- Overview of path-planning and obstacle avoidance algorithms for UAVs: A comparative study [3]
- Survey on computational-intelligence-based UAV path planning [4]
- A review: On path planning strategies for navigation of mobile robot [5]
Trajectory planning
A first taxonomy

Deterministic Graph Search

Function Optimization

Sample Based Graph Search (RRT)

Stochastic Optimization

Start Position

Goal Position

Start Position

Goal Position

Start Position

Goal Position

24/10/2023  H. Piet-Lahanier  SAUTOS
Graph Representation
- Iterative algorithms can handle most graph representation
- Grids used for simplicity of representation
- Representation may be selected for flexibility, adaptation, complexity
- Graphs $\mathcal{G} = S$, $E$ correspond to set of nodes $S$ connected by edges $E$

- Quadtree: Starting from rough graph, refining close to obstacles
- Voronoï: Edges built to be equally distant form obstacles (vertices at intersection)
- Visibility Graph: Environment with obstacles as 2D Polygons, Edges and vertices located on polygons boundary
- Voronoï: Edges built to be equally distant form obstacles (vertices at intersection)
Graph based Search
Deterministic Graph based Search

Graph Representation
- Iterative algorithms can handle most graph representation
- Grids used for simplicity of representation
- Representation may be selected for flexibility, adaptation, complexity
- Graphs $\mathcal{G} = S, E$ correspond to set of nodes $S$ connected by edges $E$
Graph search problem

- Known graph $G = S, E$
- Start vertex: $s_{\text{start}} \in S$
- Goal vertex: $s_{\text{goal}} \in S$
- Computation of costs along the edges: $g(s_i, s_j)$
- Heuristic: easy to compute approximation of cost $h: s \times s_{\text{goal}} \rightarrow \mathbb{R}$
- Result: Path $(s_{\text{start}}, s_i, s_{\text{goal}})$
Graph based Search
Deterministic Graph based Search

Graph search problem

- **g**: cost so far
  \[ g(s) = g(s_{prec}) + cost(s_{prec}, s) \]
- **h**: heuristic cost to go \( h(s, s_{goal}) \)
- **f**: evaluation of a tentative path
  \[ g(s) + h(s) \]
Method for deterministic graph based search

Initial algorithm Dijkstra’s Algorithm [7] deterministic search without heuristic
⇒ Development of heuristic-based algorithm \( A^* \) and variants (e. g. [8], [9], [10])

- Requires definition of admissible heuristics
- Heuristic function \( h(s, s_{goal}) \) is admissible: \( h(s, s_{goal}) \leq cost(s, s') + h(s', s_{goal}). \)
- Examples of heuristic: \( \Delta x \) and \( \Delta y \) difference of coordinates along \( x \) and \( y \) axes
  - Euclidian distance \( \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2} \)
  - Manhattan distance \( \Delta x + \Delta y + \Delta z \)
  - Max(\( \Delta x, \Delta y, \Delta z \))
  - Potentially weighted by \( w_{x,y,z} \)
Graph based Search
Deterministic Graph based Search

Iterative Search
What if graph evolves?
• Initial solution determined on known graph

Graph:
- $s_{start}$
- $s_{1}$
- $s_{2}$
- $s_{3}$
- $s_{4}$
- $s_{goal}$

Connections:
- $s_{start}$ to $s_{1}$ with weight 1
- $s_{1}$ to $s_{2}$ with weight 3
- $s_{1}$ to $s_{3}$ with weight 3
- $s_{2}$ to $s_{3}$ with weight 2
- $s_{3}$ to $s_{4}$ with weight 5
- $s_{4}$ to $s_{goal}$ with weight 1
Iterative Search

What if graph evolves?

- Initial solution determined on known graph
- Graph is modified (e.g. obstacles detected)
Graph based Search

Deterministic Graph based Search

Iterative Search
What if graph evolves?
- Initial solution determined on known graph
- Graph is modified (e.g. obstacles detected)
- Requires to recalculate path
Graph based Search
Deterministic Graph based Search

Iterative Search
What if graph evolves?
• Initial solution determined on known graph
• Graph is modified (e.g. obstacles detected)
• Requires to recalculate path
• Is it necessary to recompute the whole path?
Graph based Search
Deterministic Graph based Search

Incremental search methods

How can initial path can be reused? ⇒ Necessary to save information

- Initial optimal results
- Optimal path planned in terms of vertices
- Values of $g$ functions

⇒ Detection of inconsistent paths
- Perform updated graph search
- Check variation of edge costs
- Check removed or added vertices
- Find local consistency

⇒ Replace inconsistent paths by consistent paths
⇒ Reuse the parts of the graphs that were not affected
Graph based Search
Deterministic Incremental Search

Some algorithms for incremental Graph search

- General
  - Incremental All Pair Shortest Path (e. g. [11])
  - Lifelong Planning $A^*$ [12]
- Often used for vehicle path planning
  - $D^*$ Algorithm [13]
  - Improved version of $D^*$ : $D^*$ Lite [14]
- Integrating Temporal Logic
- Propositional Satisfiability and Temporal Planning (e. g. [15]).
  - Incremental Temporal Consistency (ITC) [16]
  - Space Filling Trees [17]
The $D^*$ Lite algorithm

Why use $D^*$ Lite?
- Efficient for robot navigation in partly unknown environment
- Easy to implement

What are the basic principles?
The $D^*$ Lite algorithm

Why use $D^*$ Lite?
- Efficient for robot navigation in partly unknown environment
- Easy to implement

What are the basic principles?

$D^*$ Lite

Based on Lifelong Planning $A^*$
- Use of functions $g$, $cost$, $h$ as $A^*$
- Additional function $rhs(s)$: one step prediction based on $g$ values:
\[
\min_{s' \in S_{pred}} g(s') + cost(s)
\]
Graph Based Search
Deterministic Incremental Search

\( D^* \text{ Lite} \)
- Classical determination of optimal path (same \( A^* \))
Graph Based Search
Deterministic Incremental Search

$D^*$ Lite

- Classical determination of optimal path (same $A^*$)
- Progression along optimal path: Graph is modified (e.g. obstacles detected)
**Graph Based Search**
**Deterministic Incremental Search**

**D* Lite**
- Classical determination of optimal path (same A*)
- Progression along optimal path: Graph is modified (e.g. obstacles detected)
- Computation of predicted $rhs(s)$

![Graph Diagram]

- States: $s_{start}$, $s_1$, $s_2$, $s_3$, $s_4$, $s_5$, $s_6$, $s_{goal}$
- Edges: $1$, $2$, $3$, $4$, $r$?
**Graph Based Search**

**Deterministic Incremental Search**

---

**$D^*$ Lite**

- Classical determination of optimal path (same $A^*$)
- Progression along optimal path: Graph is modified (e.g. obstacles detected)
- Computation of predicted $rhs(s)$
- Updating of the path

---

**Diagram:**

- Nodes: $s_{start}$, $s_1$, $s_2$, $s_3$, $s_4$, $s_5$, $s_6$, $s_{goal}$
- Edges with weights: $1, 2, 3, 4, 2$
Graph Based Search
Stochastic search

Two main approaches

- Probabilistic Road Map: Creation of a graph by random search, then definition of a path
  - Reactive Deformation Roadmap (local variations for attractivity or repulsion)
  - Flexible Anytime Dynamic PRM (anytime and adaptivity to local unknown environment)
- Rapid Random Tree
Graph Based Search
Stochastic search

Rapid Random Tree

- Starting Point and End Point
- From starting point: build tree by random selection of vertices and growing branches
- Check for newly created vertex: outside an obstacle or forbidden zone
- Edge construction: Verification of obstacle avoidance

Examples in [18], [19]
Graph Based Search
Stochastic search

**RRT* Algorithm**

Exploration tree $G$, Set of Vertices $V$, Set of edges $E$

- Initialize $(G, V, E)$ by considering starting point,
- Define cost between two points, e.g. distance, energy, time
- Random generation of $x_{rand} \notin Obst_j$ find $x_{near} \in G$, $x_{near} = \text{Argmin}(d(x_i, x_{rand}))$, $x_i \in G$ with $d$ to be defined
- Build the new vertex $x_{new}$ reachable from $x_{near}$ in the direction of $x_{rand}$ without obstacles collisions.
- Check with other vertices of $G$ if $x_{new}$ can be reached more economically from $x_{close} \in G$,
- If true, replace $x_{near}$ by $x_{close}$ add $x_{new}$ to $G$ and $[x_{close}, x_{new}]$ to $E$
Comparisons: zone with Identical characteristics
Similar cost Performances for $A^*$ and $RRT^*$ (faster) for example 1
Graph Based Search

Comparisons $A^*$ and $RRT^*$

Comparisons: zone with Identical characteristics
$A^*$ more efficient for cost performances in example 2
Trajectory planning
The optimization based approaches

- Deterministic Graph Search
- Function Optimization
- Stochastic Optimization
  - Sample Based Graph Search (RRT)
Deterministic Search for controlled trajectory

- Functional optimization
- Including dynamics
- Under constraints (feasible path)
- Set of inputs that minimizes a cost $J(p)$
- Satisfies $p \in \mathcal{P}$, $\mathcal{P}$ set of feasible paths
Optimal Control Problem

Determine $x(t)$ and $u(t)$

- Minimizing:
  $$g(x(0), x(t_f)) + \int_0^{t_f} c(x(t), u(t)) \, dt$$

- Subject to:
  $$\dot{x} = f(x(t), u(t)), \forall t \in [t, \, t_f]$$
  Dynamics

- Final constraints
  $$e(x(0), x(t_f)) = 0$$

- Current constraints,
  $$h(x(t), u(t)) \leq 0, \forall t \in [t, \, t_f]$$
Optimal Control Problem

Determine \( x(t) \) and \( u(t) \)

- Minimizing:
  \[
  g(x(0), x(t_f)) + \int_0^{t_f} c(x(t), u(t)) \, dt
  \]

- Subject to:
  \[
  \dot{x} = f(x(t), u(t)), \forall t \in [t, t_f]
  \]

  Dynamics

- Final constraints \( e(x(0), x(t_f)) = 0 \)

- Current constraints,
  \[
  h(x(t), u(t)) \leq 0, \forall \in [t, t_f]
  \]
**Trajectory optimization**

**Discrete formulation**

### Discretization on a limited time horizon

\[ \tilde{t} = [t_0, \ldots, t_N] \quad \tilde{x} = [x_0, \ldots, x_N] \text{ and } \tilde{u} = [u_0, \ldots, u_N] \]

- Minimizing: \[ g(x(0), x(t_f)) + \sum_{j=0}^{N} w_j c(x_j, u_j) \]
- Subject to: \[ \left| \sum_{j=0}^{N} D_{ij} x_j - f(x_i, u_i) \right| \leq \frac{1}{N^\delta} \]
- Final constraints: \[ |e(x(0), x_N)| \leq \frac{1}{N^\delta} \]
- Current constraints: \[ h(x_i, u_i) \leq 1 \frac{1}{N^\delta} \]
- Introduce bounds (actuation limits) and initial and final equalities: \[ x(0) = x_0 \text{ and } x(t_f) = x_N \]
- Search for optimal sequence (LP, NLP Mixed integer programming) and apply only first components

Examples: Cooperative trajectory [21], cooperative trajectory for search and track [22],
Trajectory optimization

Discrete formulation

Use of polynomial basis

Use of Berstein polynomials (in France Béziers Curves) [23]

• Select Berstein approximation

\[ x_N(t) = \sum_{j=0}^{N} c_j b_{j,N}(t), \quad u_N(t) = \sum_{j=0}^{N} c_{u,j} b_{j,N}(t) \]

• with polynomial basis

\[ b_{j,N}(t) = \frac{N!}{j!(N-j)!} t^j (t_f - t)^{N-j} \]

• Discretize time according to Berstein approximation: \( t_j = j \frac{t_f}{N}, \quad j = 0, \ldots, N \)

• Use property of BP for differentiation and convexity approximation

• Search for optimal coefficients by de Casteljau algorithm

⇒ Efficient determination of single or multi agent trajectories, see e.g. [24], [25], [26]
Trajectory optimization

Discrete formulation

Illustration
Trajectory optimization
Search by stochastic optimization

Various approaches

Bio-inspired methods [27]
- Ant colony [28]
- Firefly algorithm [29]

Monte-Carlo
- Particle Swarm Optimization [30]
- Genetic algorithm (Multi agent extension [31])
- Potential field algorithm [32] (efficiently combined with RRT)

⇒ increased use of RL or ML approaches
Multi-Agent Path Finding

Context

Objectives

- Defining a set of trajectories so that agents can safely rejoin their final positions
- Similar requirements in terms of performances, (length, energy spend)
- Dynamics constraints, obstacle avoidance
  → Additional Constraints: Mutual Avoidance
Main difficulties

- Combinatorial complexity: $n$ trajectories to design with interactions $NP$ hard
- Integration of mutual information: What does one know about evolution of neighbors
- Time description required: knowledge on dynamics, can the trajectory be held
- Potential adversarial decisions for criterion optimization
Multi-Agent Path Finding
Extensions of single-agent approaches

Direct extensions
- $RRT^*$ with potential fields [32]
- Short time horizon control laws [21], [22],
- Berstein polynomials [26]

Extensions of graph based methods
- Prioritized Planning
- Safe interval path planning : SIPP
- Conflict-based search for optimal multi-agent pathfinding : CBS
Multi-Agent Path Finding
Extensions of graph based method

Prioritized planning
- Use of Path planning method for single agent [33]
- Each agent receives a priority level for determining its path
- Plan must be performed avoiding conflicts with higher priority agents
- Definition of priority is made by a supervisor

Safe interval path planning
- Determination of path by $A^*$ in a dynamic environment [34]
- Division of the time space into intervals
- Path must insure safety and obstacles avoidance in the intervals
- Extension of reactive path planning methods
Multi-Agent Path Finding
Extensions of graph based method

Conflict-based search
• Determination of path with two-level optimization [35]
• Superior level for conflict identification
• Introducing new constraints
• Lower level search for optimal path by using $A^*$ type algorithm
• Extension of approach uses multi-objective [36]

Extensions of graph based methods
• CBS : Optimal search with conflict resolution, but heavy computation
• SIPP : Fast determination but conflicts are not ruled out
• Prioritizing : comparisons proposed in [37]
Uncertainty

How to account for uncertainty

- Multiple sources of uncertainty
- Obstacles
  - Obstacles locations
  - Shape of obstacles
- Other agents
  - Real locations
  - Intentions
  - Dynamics
- Agent
  - Real locations
  - Dynamics
Uncertainty

How to account for uncertainty

- Multiple sources of uncertainty
- Obstacles
  - Obstacles locations
  - Shape of obstacles
- Other agents
  - Real locations
  - Intentions
  - Dynamics
- Agent
  - Real locations
  - Dynamics

H. Piet-Lahanier
24/10/2023
Uncertainty

Risk-aware trajectories

- Integration of risk in the cost [38]
  - Reduction of propagated states by prediction of risks
  - Limitation of collision
- Determination of probabilistic boundaries [39]
  - Discrete type dynamics
  - Gaussian motion
  - Sensing uncertainty
- AI search
  - Use of RL [40], [41]
  - Determination of rewards
  - Construction of the learning bases
Graph based search

- Modelling as graph: integration of obstacles, reflect seeker information
- Integration of dynamics: adaptation of cost functions
- Dynamic changes: can be performed without reprocessing
- Uncertainty: more difficult for uncertainty on agents
- Scalability: Multi-agent in $RRT^*$, prioritized/conflict based
- Swarms: extension of SIPP and CBS to large fleet

Major issues: mostly defined for 2D, definition of costs integrating information

Optimal control

- Transformation of obstacles into constraints
- Integration of dynamics straightforwards
- Dynamic changes: adaptation of time horizon
- Uncertainty: propagation of probability with state dynamics
- Scalability: Multi-agent extensions with multi criterion (Pareto optimization)
- Swarms: trajectories designed with polynomial based approach,

Major issues: definition of costs integrating information, discretization, constraints
Merci de votre attention !
Des questions ?

www.onera.fr
Bibliographie

[1] C. Goerzen, Z. Kong et B. Mettler,
A survey of motion planning algorithms from the perspective of autonomous UAV guidance,

Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones : A survey,

[3] M. Radmanesh, M. Kumar, P. H. Guentert et M. Sarim,
Overview of path-planning and obstacle avoidance algorithms for UAVs : A comparative study,
Unmanned systems, 6 (02), pp. 95–118 (2018).
[4] Y. Zhao, Z. Zheng et Y. Liu,
\textit{Survey on computational-intelligence-based UAV path planning},

\textit{A review: On path planning strategies for navigation of mobile robot},

\textit{Recent advances in unmanned aerial vehicles real-time trajectory planning},

[7] E. W. Dijkstra et al.,
\textit{A note on two problems in connexion with graphs},

[8] A. Stentz,
\textit{The D* Algorithm for Real-Time Planning of Optimal Traverses.},
\textit{Theta*: Any-angle path planning on grids}, 

[10] A. Nash, S. Koenig et C. Tovey, 
\textit{Lazy Theta*: Any-angle path planning and path length analysis in 3D}, 
Dans \textit{Proceedings of the AAAI Conference on Artificial Intelligence}, vol. 24, 
pp. 147–154 (2010).

\textit{On dynamic shortest paths problems}, 

\textit{Lifelong planning A}, 
[13] D. Ferguson et A. Stentz,  
*Field D* : An interpolation-based path planner and replanner,  
Dans *Robotics Research : Results of the 12th International Symposium ISRR*,  

[14] S. Koenig et M. Likhachev,  
*D* lite,  

[15] D. N. Pham, J. Thornton et A. Sattar,  
*Modelling and solving temporal reasoning as propositional satisfiability*,  

[16] I.-h. Shu, R. T. Effinger, B. C. Williams et al.,  
*Enabling Fast Flexible Planning through Incremental Temporal Reasoning with Conflict Extraction*,  
[17] J. J. Kuffner et S. M. LaValle, 
Space-filling trees : A new perspective on incremental search for motion planning, 
Dans 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, 

[18] R. Pepy, M. Kieffer et E. Walter, 
Reliable robust path planning with application to mobile robots, 
International Journal of Applied Mathematics and Computer Science, 19 (3), 

R2-RRT*: Reliability-based robust mission planning of off-road autonomous ground 
vehicle under uncertain terrain environment, 
IEEE Transactions on Automation Science and Engineering, 19 (2), pp. 1030–1046 
(2021).

[20] E. Polak, 
Optimization : algorithms and consistent approximations, 


[29] B. Patle, A. Pandey, A. Jagadeesh et D. R. Parhi, 
*Path planning in uncertain environment by using firefly algorithm*, 

[30] S. Shao, Y. Peng, C. He et Y. Du, 
*Efficient path planning for UAV formation via comprehensively improved particle swarm optimization*, 
ISA transactions, 97, pp. 415–430 (2020).

[31] H. Shorakaei, M. Vahdani, B. Imani et A. Gholami, 
*Optimal cooperative path planning of unmanned aerial vehicles by a parallel genetic algorithm*, 

[32] F. Bayat, S. Najafinia et M. Aliyari, 
*Mobile robots path planning: Electrostatic potential field approach*, 
[33] B. Binder, F. Beck, F. König et M. Bader,
*Multi robot route planning (MRRP) : Extended spatial-temporal prioritized planning*,

[34] V. Narayanan, M. Phillips et M. Likhachev,
*Anytime safe interval path planning for dynamic environments*,

[35] G. Sharon, R. Stern, A. Felner et N. R. Sturtevant,
*Conflict-based search for optimal multi-agent pathfinding*,

[36] Z. Ren, S. Rathinam et H. Choset,
*Multi-objective conflict-based search for multi-agent path finding*,


[41] S. Aradi,
Survey of deep reinforcement learning for motion planning of autonomous vehicles,