DUC DARPA ~~Bots~~ DUC https://kpit-ad.udemy.com

1. finding a path
2. Searching for the safest manoeuvre
3. Determine the most feasible trajectory

Complexity Performance accuracy

→ evaluate → advantage/disadvantage
→ inherent limitation.
→ feasibility
→ optimality
→ handling of obstacle
→ Testing operational environments

(V2V, V2I).

Traffic flow boundry. zhang et al 2013

planning for memory for parallel running.

Existing Planning from - Robotics

applied on-road & offroad vehicle.

approach concerned with planning

Intelligent driving divided into four hierarchical

1) route planning
2) path planning
3) manoeuvre choice
4) Trajectory Planning (termed as control planning)

→ takes vehicle dynamics
→ obstacles,
→ Road geometry
→ Traffic interactions into account

Planning local on Road

vehicle position orientation

state of the vehicle at the moment.

state of a vehicle's are — position (x, y, z)
— orientation $(\theta_x, \theta_y, \theta_z)$
— linear velocity (v_x, v_y, v_z)
— angular velocity $(\omega_x, \omega_y, \omega_z)$

Holonomy :-

Cars - non-holonomic b/z their motion is constrained by non-integrable differential constraints due to the assumption that the wheels roll without slipping.

→ Actions are system inputs.

→ Configuration space or state space, planning is a computationally intensive task, demanding high memory utilization.

Path :- Path-Planning - geometric path

manoeuvre :- Regarding position & speed.

manoeuvre planning - address the problem of taking high level decision.

Trajectory :- . sequence of states visited.

Trajectory planning also called trajectory generation.

vehicles kinematic limits based on vehicle dynamics.

→ Parametrized by time as well as acceleration.

→ Trajectory planning at scheduled at regular intervals.

Come test at (VIS lab)

Trajectory error

ROS

Search space for Planning:

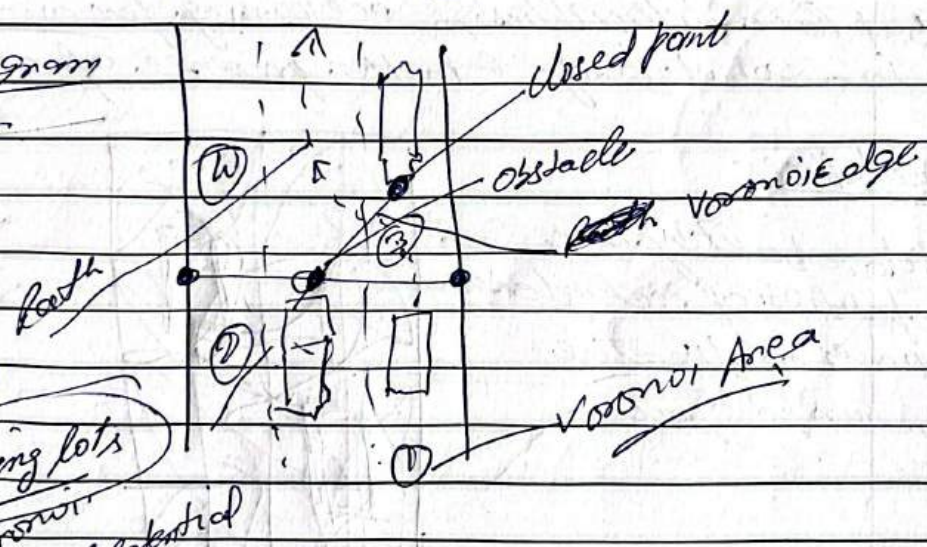
enable the query for a path.

→ reading from sensor & information obtained from digital map. it used to continuous of the environment into a digital representation of the road n/w which is essential space for planning.

Continuous Search: in continuous coordinate using road boundaries boundaries & positions of the obstacle.

→ Decomposition technique analyse the space with higher resolution & include Voronoi Diagram, occupancy grid, cost maps & lattices, Brivi corridor.

Voronoi Diagram



Comparison

mostly used in parking lots by Voronoi diagram potential fields → An obstacle avoidance algorithm derived from mobile robotics.

This combine approach reflects as Voronoi field, was developed to overcome the issue of conventional potential field approaches in narrow passages.

→ Not so suitable for on-road path planning.

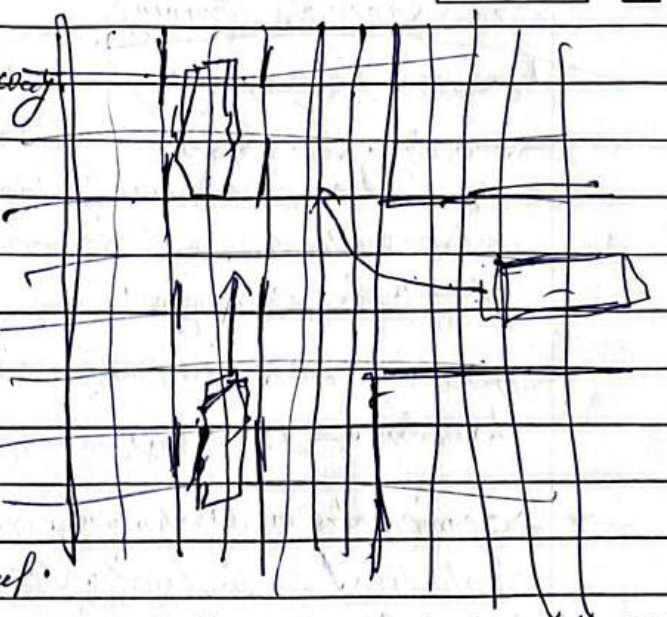
→ Voronoi edges along which a vehicle navigates, can potentially be discontinuous & unsuitable for non-holonomic vehicle.

Scanned with CamScanner

Occupancy Grids:

Occupancy & Cost map works in similar way.

It discretise the state space into grid & each cell of the grid is allocated with probability of the each cell being occupied by an obstacle or a cost proportional to the feasibility or risk of traversal.



→ Fast in finding a solution with low computation power but have difficulties in accounting for non linear dynamics in a robust way & in the presence of obstacle.

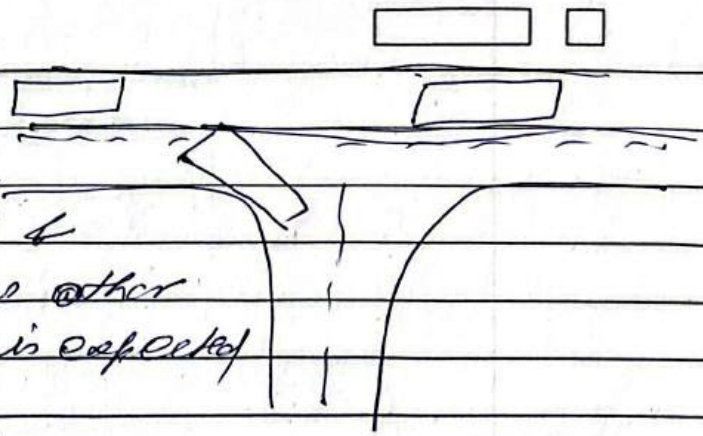
State lattices:

grid are built by the repetition of rectangles or squares to discretise continuous pt space.

→ The problem of planning then reduces to a boundary value problem of connecting the original state with required final state.

→ It overcome the limitation of grid based technique in efficiency without increasing computation power.

Driving Corridors:



represents continuous collision free space, bounded by road & lane boundaries as well as other obstacles, where a vehicle is expected to move.

SLAM — Simulation localisation & mapping technique.

→ It is constructed for each vehicle according to the chosen manoeuvre.

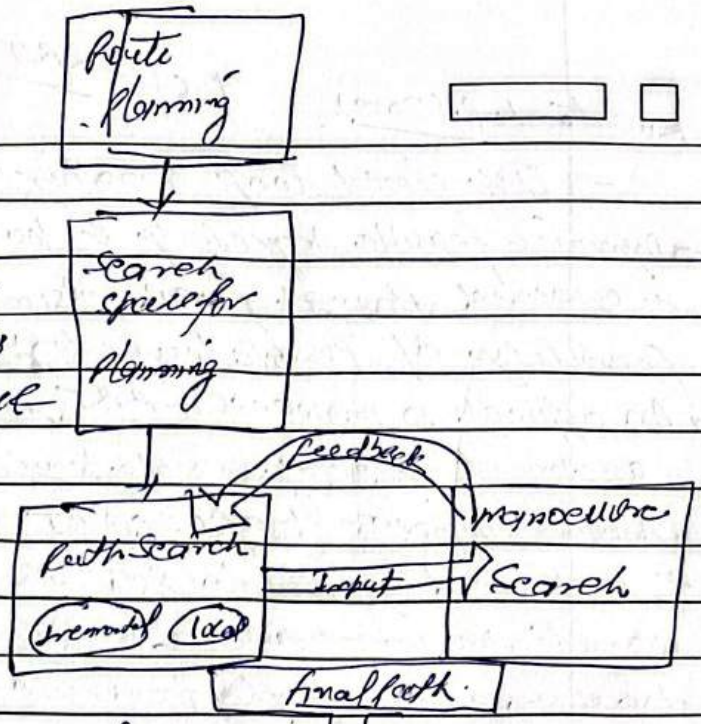
→ Drawback → Computation power is needed for planning for the entire range of coordinates regarding the road n/w, representation of roads or lanes may constrain the motion of the vehicle.

allure of for planning capability is not only for better results in all three level (path, manoeuvre & trajectory planning)

	Pro	Cons
Voronoi diagram	— Completeness — max distance from obstacle	limited to static environments. Discontinous edge.
occupancy grid/costmaps	— Fast discretization. — small computational power	— Problem with vehicle dynamics — error in the presence of obstacle
state lattice	— efficiency without increasing computional time — Pre-computation of edge is possible	— Problem with curvature — Restrict motion. — Difficulties in dealing with evasive manoeuvre
Driving corridors	— Continuous collision free space for the vehicle to move	— computational cost — constraints on motion.

Scanned with CamScanner

Planning technique



→ finding best path.

→ should follow ~~model to~~ motion model, way-points & traffic environment (static & dynamic obstacles).

→ Planning – Increment search & local one.

→ final path may change based best manoeuvre.

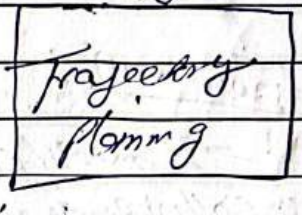
→ once path finished then final trajectory planning is generated.

It is three levels.

1) ~~a)~~ finding the best geometric path for the vehicle to follow.

2) find best manoeuvers to perform.

3) find best trajectory to follow – the optimisation of a geometric curve.



Increment search :- searching for best path.

search technique not fully specified in advance & these algorithm also re-use information from previous search to increased search speed.

1) Rapidly - exploring Random Trees (RRT)

2) Lattice Planners (LP)

RRT algorithm along with B-splines to produce smooth paths.

(randomisization parameters - acceleration, acceleration duration, sampling time & orientation angle.

Rasekh et al (2006)

8m x 8m simulation → laser range finders, monocular camera, IMU & GPS with these path find effectively.

Kuwata et al. (2009). (CL-RRT)

→ Use closed loop controller.

→ Dynamic feasible trajectory & traversability cost for each vertex is obtained through forward simulation which evaluate the feasibility of possible trajectory.

→ An approach to improve CL-RRT is ~~cost~~ combined with threat assessment for every possible trajectory using game theoretic aspect.

→ Another improvement of CL-RRT, in term of obstacle handling is implemented by Lindqvist (2014) where Gaussian process are used to estimate the intentions of the obstacle in order to safely steer these expansion of the tree, sub the trade-off the path optimality, real time constraints & intentions estimation in an area that needs further research.

Karaman & Frazzoli (2011).

Hamotte (2014) → It modify RRT with two extra lines are

I) Collision checking

II) expansion.

Rouyes (astrid (2013) → used advanced techniques

Dynamic system of vehicle & road network formed as kripke structure.

Lattice

~~State lattice~~ construct a discrete search space which ensures relevant state continuity.

~~reduce~~ computational cost as pre-computation is employed.

Local search : most popular – Search space contains a certain geometric curve & several lateral shifts of this curve.

Path generated with lateral shifts i) lateral shifts in the ~~collision~~ ~~Evading~~ action space of the vehicle

ii) lateral shifts in the state space.

Searching entire entire state space towards goal with an infinite horizon requiring substantial computation power, Partial motion planning (PMP).

→ PMP use short ~~term~~ time horizon.

Inevitable collision state (ICS).

→ If ICS find in path, an alternative path is searched and at each time step a nodes are expanded in an RRT fashion.

#3.3 manoeuvre planning & decision making.

manoeuvre planning moves away from searching a path or generate a trajectory; instead act as a brain, which refers the results of path search, interacts with other traffic participants & gives the approval for the geometric path before it is transformed to a feasible trajectory.

→ ~~Two~~ Planning in two categories.

i) Those that emphasis motion modelling & obstacle prediction.

ii) Those that are concerned with the decision making module of autonomous vehicle, based on the modelling of the traffic environment.

i) ~~most~~ motion models ~~according~~ are classified into physics-based, manoeuvre-based & interaction-aware models.

→ Crash can be predicted through collision prediction through estimated trajectories.

→ Grid based bayessian filter – to anticipate curve roads & infer the trajectories of other vehicles.

Scanned with CamScanner

Hybrid-state system → hidden markov model
→ finite state machines } to predict future state of traffic participants

at intersections ← this approach need extensive driving data & extensive data acquisition to train the models.

→ Ontology:

→ Partially observable markov decision processes (POMDPs)
→ 'make accurate prediction for other traffic participants.
→ Their technique can respond well with noisy sensors, allow long term predictions.

IIS

use multiple criteria decision making (MCDM) & Deterministic finite automata (DFA).
→ data comes from prior known data, sensors, vehicle comm.

Hanely & Campbell — hidden motion of dynamic obstacles & the presence of static ones.

Kala & Warwick

hidden & cost function based (HCB)
reference trajectory + static & dynamic obstacles → Input reduction 90% computational cost.

Nanathory:

Trajectory level: - after finding best path.
Trajectory must be generated that satisfies the motion model or state constraints & guarantees comfort.
→ Problem of generating trajectory is primary solved by selecting a geometric curve to assure smooth motion through road network.
→ Arc, clothoids, Nelson polynomials, polynomial spirals, spline curves, Bezier curves.

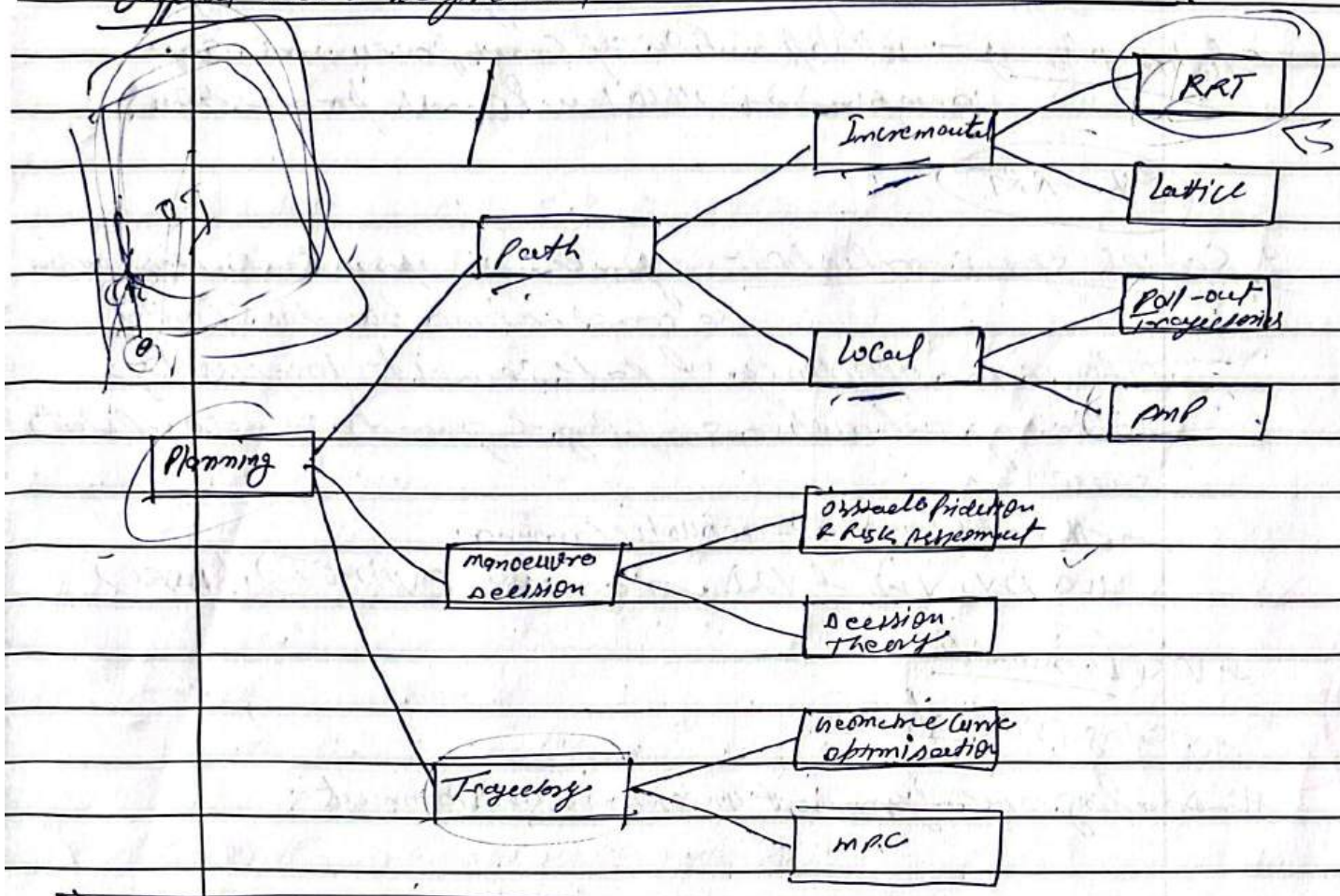
Scanned with CamScanner

Trajectory generation algorithm (Tiji) — ~~Second order~~ polynomial.

error between current state & final state.

Smoother & more controllable trajectory → used Bezier curves.

different order polynomial have been used in this area.



Constraints & limitations

1) Obstacle challenge: Need huge computational power requirement since the obstacles trajectories need to be calculated & check at each moment.

→ Obstacle represent as rectangle or circle.

→ Inability to see around corners & detect obstacles.

2. Vehicle dynamics.

3 Risk Indicators.

4. Sensing & Perception.

epoch (sensing cycle)

future research directions.

1 Risk assessment → misplaced perception & decision making
~~use of Bayesian Game theory~~ → have potential to assist real time handling of interaction such as at intersections, merging into traffic or overtaking with limited perception.
future planning — identification of crash precursors, gap acceptance & H/W level crash prediction.

(Risk - RRT)

2 Search space reduction:- prior search space constructed within the road or lane boundaries could enhance the efficiency of path search techniques by addressing exhaustive sampling & greedy searches of state space.
→ thermal camera & acoustic camera.
we V2V, V2I & V2D. → as V2X (V2H&T), (DSRC)

* Vehicle model.
* Testing & validation.
* Dealing with dynamic & uncertain environment.

holonomic, nonholonomic & kinodynamic → for 12 DOF

k-nearest neighbor

~~Closed loop~~ RRT

Rapidly exploring Random Tree (RRT)

→ Randomized data structure.

→ ~~use to handle nonholonomic constraints & high degree of freedom.~~

RRT popular Two → Randomized potential field algorithm
→ probabilistic roadmap algorithm.

existing problem with RRT → Although powerful for standard path planning, do not naturally extend to general nonholonomic planning problems.

→ Randomized potential field method depends heavily on the choice of a good heuristic potential function.

→ local planning steps must be efficient.

→ Tend to better performance analysis & consistency of behavior.

→ RRT can directly applied to nonholonomic & kinodynamic planning.

→ RRT do not require any connection to be made b/w pairs of configurations, while Roadmap requires tens thousands of connections.

→ RRT more efficient than Barrett's probabilistic roadmap for holonomic path planning.

→ heavily biased towards unexplored portions of the state space.

→ distribution of vertices in an RRT approach the sampling ~~distribution of vertices in an RRT~~ distribution, leading to consistent behaviour.

→ RRT probabilistically complete under very general cond^n.

→ always connected even though the number of edges is minimal.

→ RRT constructed by iteratively breaking large Voronoi regions into smaller ones.

→ RRT could replace the random walk stage in a randomized potential field approach.

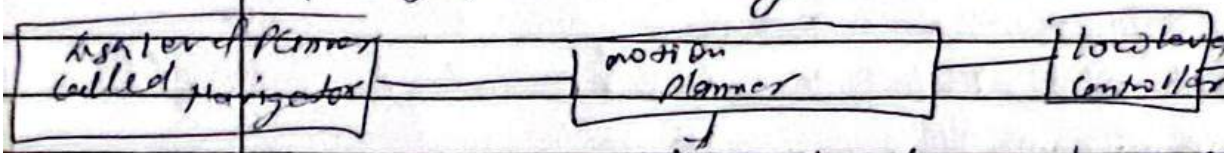
- There are many issues involved in ~~blasting the~~ [illegible] grand.

→ bias can be given that slightly prefer a goal state

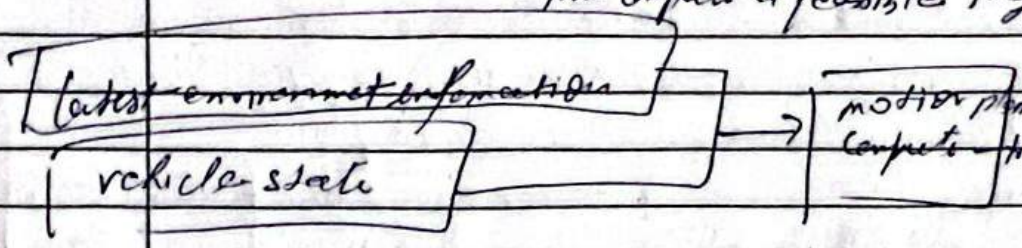
CL-RRT - dynamically feasible plan in real time.

DUC - DARPA Urban Challenge.

→ for planning, control & sensing.



- This compute a feasible trajectory



for motion planning we do build on the RRT algorithm.

It belongs to class of incremental sampling-based methods.

→ Unstable nature of the vehicle dynamics requires the addition of a path-tracking control-loop whose performance is generally hard to characterize.

→ Use of RRT robotic vehicle with complex, unstable dynamics & significant drift, while preserving safety in the face of uncertainty & limited sensing.

CL-RRT

1. Introduction
2. Problem formulation
3. Planning over closed-loop dynamics
4. Tree Expansion — A - sampling strategies
   B. feasibility check + Risk evaluation
   C. Node Connection strategy
      I) Discrete motion
      II) selection Criteria
   D. Cost-To-Go Estimate
   E. Safety as an invariant property
      I) Safe state
      II) Unsafe Node
5. ONLINE REPLANNING
   A. Committed part of the tree
   B. Lazy Revaluation
   C. Reprapagation
6. Implementation of the controller
   A. Steering controller
   B. Speed controller
   C. Accounting for the Prediction Error
7. Application Results
   A. vehicle model
   B. Race Result
      I) High speed behaviour on a curvy road
      II) Unsafe Nodes in the Dynamic environment
      III) Parking
      IV) Road Blockage
      V) Interaction with Virginia Tech's car
      VI) Reprapagation
8) Conclusion

Scanned with CamScanner

CL-RRT - Proposed algorithm is at the core of the Planning & Control Software.

Extension to the standard RRT are predominantly motivated by I) The need to generate dynamically feasible plans in real time.
II) Safety requirement
III) The constraints dictated by the uncertain operating (Urban) environment.

Introduction

Why this project is required

vehicles are expected to stay in the correct lane; maintain a safe speed at or under specified speed limit, correctly yield to other vehicle at intersections pass other vehicle when safe to do so, recognize blockage & execute U-turn when needed & park in an assigned space.

Focus on motion planning (motion planner).

motion planner computes a feasible trajectory to reach a goal point specified by the navigator.

Challenge in motion planning — I: Complex & unstable vehicle dynamics, with substantial drift.
II) Limited sensing capabilities, such as range & visibility in an uncertain, time-varying environment.
III) temporal & logical constraints on the vehicle's behaviors.

Scanned with CamScanner

(1, 4, 5)

approaches to address the motion planning problem

140 miles

7.8 miles completed

198 — 23

for motion planning chose RRT algorithm (9 – 11)

(Why RRT) i) Sampling based algorithm are applicable to very general dynamic model.

ii) Incremental nature of the algorithm lends itself easily to real time, on-line implementation, while retaining certain completeness guarantees.

iii) Sampling base of method do not require the explicit enumeration of constraints, but allow trajectory-wise checking of possible very complex constraints.

→ It can be stopped in intermediately by setting the control input to zero.

⇒ This approach enable on line use of RRT with complex, unstable dynamics & significant drift, while preserving safety in the face of uncertainty & limited sensing.

(ii) Problem formulation

motion planning problem:

vehicle has nonlinear dynamics

$\dot{x}(t) = f(x(t), u(t))$ $\quad x(0) = x_0$, — (1)

static in nature & inputs of the system.

$x_0 \rightarrow$ Initial state

$x(t) \in X_{free}(t)$ $\quad u(t) \in U$

avoidance constraints for moving obstacles.

$X_{goal} \subset R^n \rightarrow$ motion planning problem ~~given~~ is assumed to be given by a higher level route planner.

motion planning problem

$$t_{goal} + \int_0^t \psi(x_{ref}(t), r(t))\, dt.$$

(III) Planning over closed-loop dynamics

Randomized planning algorithm solve for the input to the vehicle $u(t)$ (9, 10, 12–14)
controller (15)

→ low level controller reference command $r(t) \in \mathbb{R}^{n_r}$
dimension vehicle state n_x would be large but r in lower dimension.

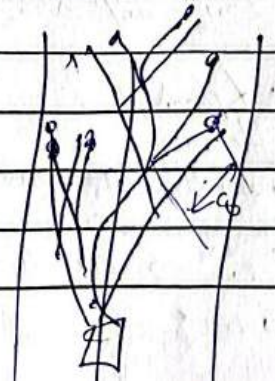
$$\boxed{n_r << n_x}$$

reference in 2D path:
→ (x_{ij}, v_{cmd}) → x_{ij} position + v_{cmd} of the reference path
v_{cmd} → desired speed.
→ CL-RRT run simulation using vehicle model & controller compute state trajector $x(t)$.
CL-RRT works with unstable dynamics like CAR
Most important → A single input to the closed-loop system can create a long trajectory.

(IV) Tree Expansion :-

→ At tree growing phase, the best trajectory is chosen for execution & the cycle repeats.

Algo 1 → How tree grows.



A. Sampling strategies:- → In Urban Sampling the space is in a random manner & result in large no. of waste samples → To improve efficiency [16 – 20]

$(S_x, S_y, V_{cmd})_i$ for $i \in \{1, 2, \ldots, n_{ref}\}$

(S_x, S_y) → generated by random ~~number~~ Sampling.

V_{cmd} → ~~sto speed~~ stopped state at the end. [15]

Sampling point $s = [S_x, S_y]^T$ → reference point (x_0, y_0, θ_0)

by $\begin{bmatrix} S_x \\ S_y \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + r \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix}$ with $\begin{cases} r = \sigma_r |n_r| + r_0, \\ \theta = \sigma_\theta n_\theta + \theta_0 \end{cases}$

n_r & n_θ → random variables with gaussian distribution σ_r, in radial direction

σ_θ → standard deviation in circumferential direction.

r_0 → offset w.r.t. (x_0, y_0).

→ maneuvers can generated by changing above parameters according to vehicle location.

1) Physical Environment as a Bias:-

Sampling start with randomly shifting the estimated lane centre points in the lateral direction, to fully exploit the lane shape.

→ At intersection, wider & relatively short Gaussian distribution used

σ_r → intersection set to distance to the goal

σ_θ → 0.4π

→ batch Sampling → It is useful when intended trajectory has some specific shape, such as arriving to the goal with a certain heading.

2. Logic of Environment as a Bias:- wider distribution is used $\sigma_\theta = 0.4\pi$.

(U-turn → three points: i) travel forward left.

ii) reverses right. iii) moves forward into the target lane with the correct heading.

B. Feasibility Check & Risk Evaluation:-
After the forward propagation, the resulting predicted trajectory is tested with obstacle & rules of the road.
→ static & moving obstacle
→ lane boundaries
→ standoff distance
drivability map → 2D grid. 20cm.
i) infeasible, ii) restricted (iii) drivable with penalty.
obstacle & lane boundaries.

C. Node Connection strategy:-

i) Distance metric: Attempt to connect sample to the closest node in the tree. Shorter paths have lower probability of collision.
→ RRT can quickly cover the free space without wasting many samples.
→ CL-RRT algorithm uses the length of the Dubins path b/w a node & the sample. (2) [22]
→ Euclidean group of dimension.

ii) Selection criteria: [13] [14]

D. Cost to Go Estimate:- Every time a new node q is added to the tree, CL-RRT computes estimates of the cost to go from q to the goal.
lower bound upper bound

CLB

$$CUB = \min_{c \in CLB} \{ c(e_c) + CUB(c) \}$$

Once a feasible trajectory found to the goal, the best sequence of nodes is selected based on the upper bound CUB.

RRT based planner

E. Safety as an invariant property.

i) Safe state [13][25]

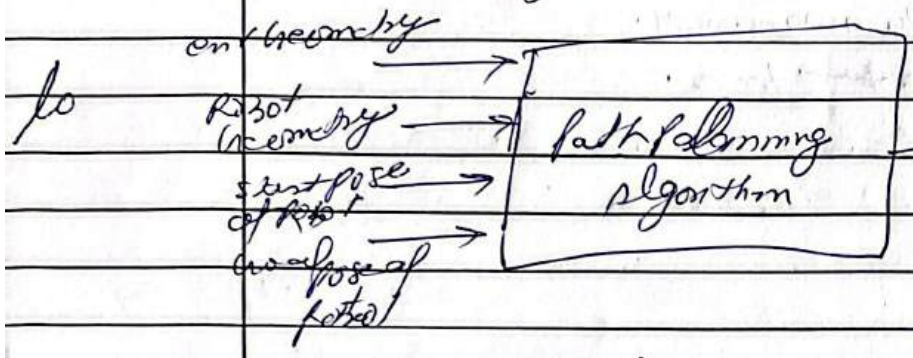
ii) Unsafe Mode.

V) ONLINE REPLANNING:- The tree needs to keep growing during the execution b/c of the constant update of the situation awareness.

A. Committed part of the ~~tree~~ tree.

Introduction: intro to planning & navigation.

Roadmap
→ Classical Path Planning
→ Sample Based & Probabilistic Planning
→ Navigation with Deep learning



Approach to Path planning
1. discrete or combinatorial) path planning
2. Sample-based path planning
3. Probabilistic path planning

4. Continuous representation → Discretization → Graph Search

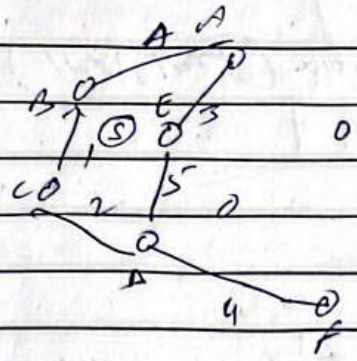
Minkowski Sum
3D Configuration space
Discretization
Roadmap
visibility Graph
Voronoi diagram
cell decomposition
potential field

Graph Search
Breadth-first Search
Depth-first Search
→ Uniform Cost Search

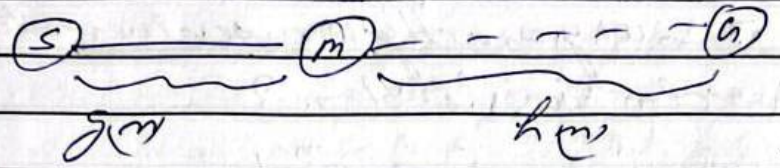
Uninformed

Informed Search

A* Search



or goal



$f(m) = g(m) + h(m)$

heuristic value of each node

node	g(m)	h(m)	f(m)
(E)(S)(A)	5 +	3	8
(E)(C)(L)	6 +	3	9
(E)+(B)	3 +	8	11
(E)+(A)	5 +	8	13

modeling the problem

0	1	0	0	0	0
0	1	0	0	0	0
0	1	0	0	0	0
0	1	0	0	0	0
0	1	0	0	0	0
0	0	0	1	1	0

[g, x, y] g → total cost of expanding

A* shortest Path

x_d = x goal − x cell position

y_d = y goal − y cell position

manhattan distance $d = |x_d| + |y_d|$

manhattan heuristic vector

9	8	7	6	5	4
8	7	6	5	4	3
7	6	5	4	3	2
6	5	4	3	2	1
5	4	3	2	1	0

Euclidean distance $d = \sqrt{x_d^2 + y_d^2}$

6	5	5	4	4	4
5	5	4	3	3	3
5	4	3	2	2	2
5	4	3	2	1	1
5	4	3	2	1	0

Scanned with CamScanner

Chebyshev distance $d = \max(|x_d|, |y_d|)$

5	4	4	4	4	4
5	4	3	3	3	3
5	4	3	2	2	2
5	4	3	2	1	1
5	4	3	2	1	0

(2) Sample Based → so why exactly we used direct planning for higher dimensional problem ?

→ ~~n-dimensional space → 3^n -1~~

3d → $3^3 - 1$ connected space

holonomic & non-holonomic

Sample Based planning ? (1) Probabilistic Roadmap method

(2) Rapidly Exploring Random Tree method

(1) multi-query planner

(2) single-query planner → much quicker than RPM

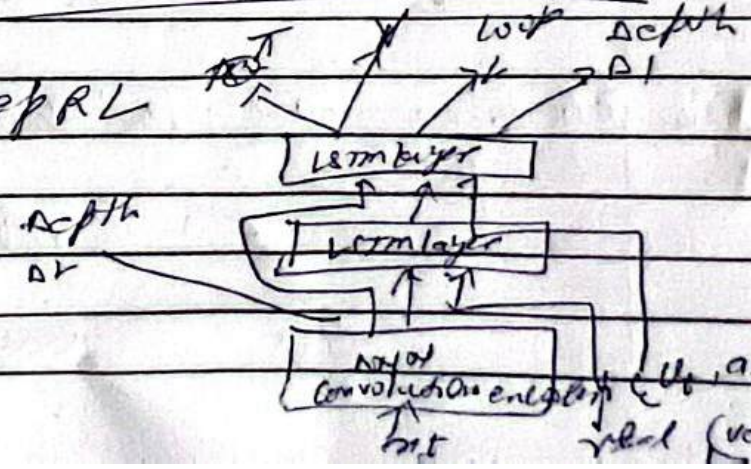
Path smoothing

(3) Introduction to probabilistic path planning

1) markov decision process

? Value Iteration Algorithm

Navigation with deep RL



ABC - Peter Corke

RRT

→ It search non convex.

→ for high dimensionality.

→ Constructed incrementally toward

→ Biased to grow large unsearched area. ~~Convex~~ convex hull

→ technique to generate open loop trajectories for nonlinear sys.

→ Considered as Monte Carlo method. to bias, into largest Voronoi regions of graph.

Algorithm

Algorithm ~~BuildRRT~~ BuildRRT
input: Initial Configuration q_{init}, no. of vertices in RRT K, incremental distance Δq)
output: RRT graph G.

```
G.init(q_init)
for k=1 to K
    q_rand = RAND_CONF() or RAND_FREE_CONF()
    q_near = NEAREST_VERTEX(q_rand, G)
    q_new = NEW_CONF(q_near, q_rand, Δq)
    G.add_vertex(q_new)
    G.add_edge(q_near, q_new)
    return G.
```

growth factor → tree now scale is limited by it.

Dijkstra's Algorithm → find shortest path.

1. start point
2. goal point
3. obstacle list
4. rand Area.
5. expand Dis = q
6. goal Sample Rate
7. max iter

Scanned with CamScanner

RRT* → slow or compare ↳ work in ellipsoid.

Tree Based Planning

→ Sample points add to tree to grow in entire space

→ It covers entire space quick efficient.

→ Probabilistic completeness. → If no. of node increase then probability to get solution becomes one.

→ ∞ not optimal.

→ Connection b/w nodes set only once.

→ used for look ahead in path planning where can see related like

→ CL-RRT ensure that agent will not deviate from the planned path ~~& other hand ensure~~ prune infeasible branches when the agent move on the tree.

It is suitable for multi query task that means generating path to multiple goals.

Potential Based Planning:-

Graph Based Path Planning → make grid of environment.

→ A*

$X \subseteq R^2$ or R^3

$X \subseteq R^2$

It is bounded.

X has obstacle which is dynamic

$X_{obst} \subseteq X$ – set of all obstacle.

free space

$X_{free} = X / X_{obs}$

Tree denoted by τ

each node inside representation

$x_i \in X_{free}$.

Current tree root x_0 → change move agent moves

Set of planned node – $(x_0, x_1 \ldots x_k)$ {k → user define limit for planning a path ahead from x_0

$x_{goal} \in X_{free}$

Cost $\rightarrow x_a \in X_{free}$ → It is position of agent

x_{goal} → should be a minimum length.

→ To find minimal length in dynamic → use cost-to-reach values

cost to reach value denoted by c_i which can be computed by Euclidean length of path from x_0 to x_i

→ restricted the tree to be real time means where to process minimum time to process the tree in Real time con. ~~evaluated tree~~ expansion tree & re-writing of path planning with expanded tree.

→ when tree growth & path does not find the used cost function $f_i = c_i + h_i$ → from x_0 to point close to goal

h_i → cost to go values.

Compute the value of using Euclidean method.

#

In multi query task each node in tree ~~or~~ space has potential to become goal while dynamic obstacle moving around & may block some paths.

→ Tree is really fast to in real time manner to react when to change in environment & return minimal ~~path length~~ path length to goal.

methods: Tree Algorithm - (Simple tree algorithm)

→ goal Sample rate → Controlling the direction of the tree growth factor

→ RRT growth can be biased by increasing the probability of sampling state from a specific area.

→ higher the probability the more greedy the tree grows towards the goal.

PSPACE - RRT

RRT* → for optimal solution.
about RRT to achieve the asymptotic

One of key drawback of RRT & RPM is that they often produce suboptimal solutions that 'wander' through state space

dtan2 (ran(1) - nearest.y, rand(1) - nearest.x)

atan2 → ~~mon means~~ atan(y/x) — radians

~~newNode = Deep copy return value b/w -π to π~~

sin θ
cos θ

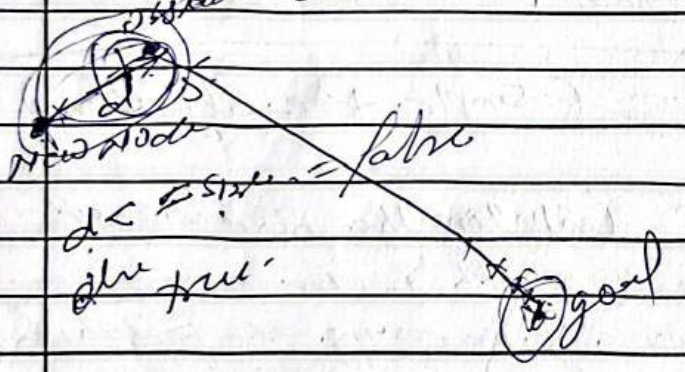
newNode = Deepcopy (nearestNode)

obstacle, x, y, size

dx = obstacle x - newNode x

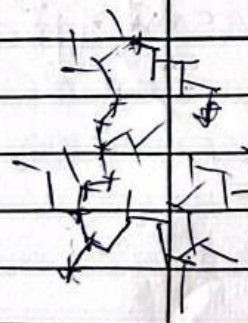
dy = obstacle y - newNode y

Euclidean distance d = sqrt(dx² + dy²)



d < size = false

d >= true



$x_1 y_1$

$y_i x_i$

d_1 d_2 d_3 d_4

$d = \sqrt{(x_i - y_i)^2 + (y_i - y_1)^2}$

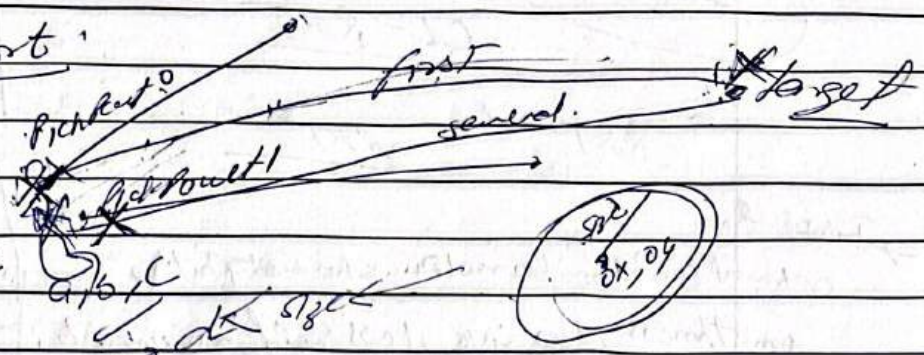
dx dy

length = $d_0 + d_1 + d_2 + d_3 + d_4$

(0, 15) lengths

$\leftrightarrow$ length → two uniform value

choose 4/w sort.



4/w Sort.

a, b, c

$x_1 y_1$

$x_2 y_2$

$a = y_2 - y_1$

$b = -(x_2 - x_1)$

$c = y_2 x (x_2 - x_1) - x_2 x (y_2 - y_1)$

$d = \frac{|a x + b y + c|}{\sqrt{a^2 + b^2}}$

$d = d \cdot abs(a x + b y + c)$

$d = d \cdot (a x + b y + c)$

if d

method

method (CL-RRT) → method needs to be wrote about RRT

→ method needs to be wrote about ~~CL-RRT~~ CL-RRT.

method - RRT.

optimized by CL-RRT

RRT - Smoothing

RRT - tree - Real path Planning

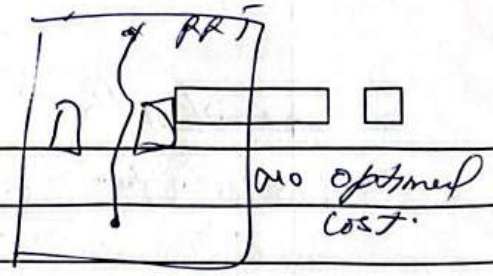
1. RRT - tree

2. RRT - tree expansion & Rewiring

CL-RRT.

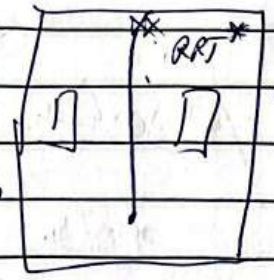
CL-RRT*

CL-RRT



RRT*

RRT*



near optimal cost.

→ Introduce

optimal could be based on shortest path, smoothness, low risk, less fuel requirements, maximum Area coverage, low energy consumption.

* Planner like Roadmap (RM), Potential field, cell decompositions are unable to deal with dynamic & complex high dimension problems.

→ Grid based Algorithms are expensive for a high dimension.

→ Evolutionary Algorithm are suitable for multiobjective problem (Algorithms are → PSO, Ant colony, Genetic

→ RRT* always coverage an optimal solution if adequate run time is provided.

→ RRT* has tremendous scalers to solve high dimensional complex problem.

Methodology

RRT* construct multiple short paths randomly organized as tree instead of one long path.

① Tree expansion

The process of selecting least cost parent & rewiring tree are two most promising features of RRT*.

→ The best parent selection & rewiring of tree improve the path quality. These features have an efficiency trade-off with path quality & make convergence slow as number of nodes in the tree increases.

When goal is found, a path connecting "init point" & goal is established. This path is improved as planner continues until a predefined number of iterations are executed or given time expires.

Imp. four basic steps to write RRT*. ① find near neighbors. ② Select best parent ③ check cost for rewiring ④ Rewrite tree.

Algorithm

```
T ← Initialize Tree():
T ← Insert Node(φ, Zinit, T)
  for i = 0 to i = N do
  Zrand ← Sample(i)
  Znearest ← Nearest(T, Zrand)
  ~~if obstacle free()~~
  (Znew, Unew) ← steer(Znearest, Zrand)
  if obstacle free(Znew) then
    Znear ← Near(T, Znew, |V|)
    Zmin ← chooseparent(Znear, Znearest, Znew)
    T ← InsertNode(Zmin, Znew, T)
    T ← Rewire(T, Znear, Zmin, Znew);
return T.
```

offline online (local)
sensor based
global
(HD map)

(i) Single Directional Holonomic RRT* Approaches.

① ~~Anytime variant of RRT*~~. The basic idea of Anytime variant of RRT* is to deal with the issue of large computational time by executing the planner for a predefined planning time. Once an initial path is obtained & stored, the rest of the time is used to improve initial solution.

→ Anytime RRT* involves two key features

I) Committed trajectory

II) Branch and - bound adaption.

RRT*-Smart ⇒ have two major features

I) Intelligent sampling

II) path optimization.

Scanned with CamScanner

(ii) Bidirectional Holonomic RRT* approaches:

→ It generates two trees simultaneously from start to goal state directing towards each other.

(c) Non-holonomic & Kinodynamic RRT* approaches:

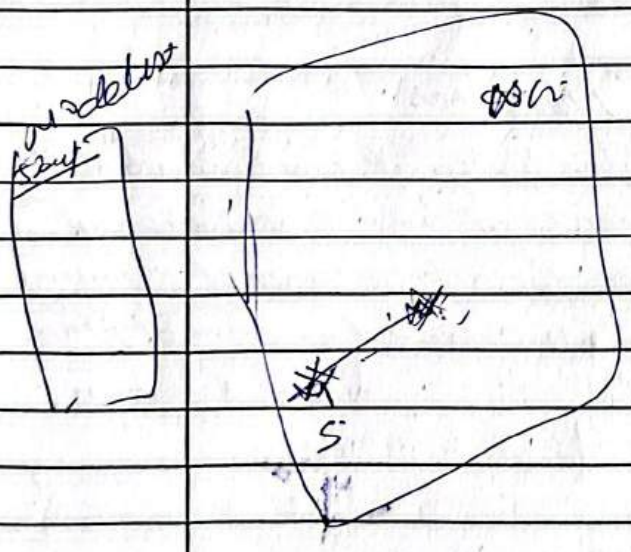
Non-holonomic path planning requires satisfying both internal constraints (Physical limitations of Robot) & external constraints (obstacles in environment).

→ motion planning problems for which velocity & acceleration bounds must be satisfied.

→ one extended approaches for kinodynamic RRT* called adaptive RRT∞

Code

~~steer~~ steer →



CL-RRT

→ Planning with closed loop prediction allows us to handle complex unstable dynamics & avoids the need to find computationally hard steering procedures.

→ This algorithm allows us to improve the solution quality by searching over alternative reference trajectories.

---

many algorithms have been studied in the past for robot motion planning problems, ~~not~~ real time motion planning for vehicle with complex dynamic & constraints has still been a challange.

→ Grid based searches algorithms such as A^*, Δ^* or E^* can efficiently search for a global plan but it is difficult to account for the non-linear dynamics ~~for~~ of the vehicle or tight maneuvers.

→ Each node in the closed loop RRT stores the states of the vehicle x & the corresponding controller input r.

Algorithm

1. repeat.
2. → measure the current vehicle state & environment.
3. → propogate the state by the computation time limit.
4. repeat
5. Generate a sample for the input to the controller.
6. Sort the node in the tree using heuristics.
7. for each sorted node do
8. form the controller input by drawing a line from the node to the sample, then propogate.
9. if the propogated portion is collision free then
10. add the sample to the tree. break.
11. end if.
12. end for.
13. until the time is reached.
14. Choose the best path & propogate from the current states.
15. if the repropogated trajectory is infeasible then.

Difficult

16. Remove the infeasible portion from the tree Ego-Line 14.
17. end if.
18. Send the best path to the Controller.
19. until the vehicle reaches the target.

→ Key difference b/w RRT & CL-RRT is that the sample generated is (x,y) point but is not taken in the space or the configuration space. Instead it is in the controller input space.

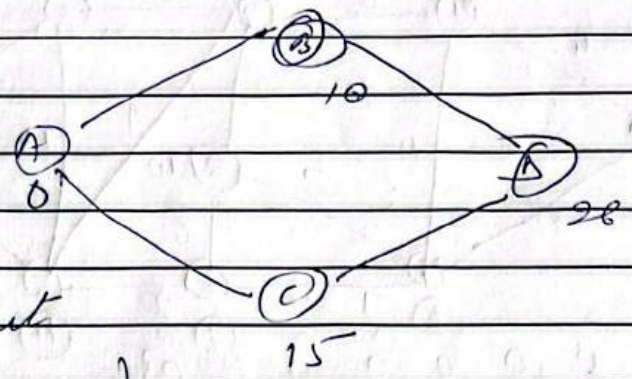
→ ~~three~~ three maintain two basic information.
I) One for the input to the controller.
II) other for the predicted trajectory of the vehicle. which is the output of the forward simulation.

When controller does not accurately track the reference path due to modeling errors, the planner could change the reference path so that the vehicle achieves the original desired path.

many ways to reduce prediction error.
I) Use of Dual for L1 rescheduling.
II) Space dependent speed command.
III) Repropagation.

Dijkstra

This is a Greedy Algorithm



Vertex	d (distance)	π (Parent)
A	0	NIL
B	∞	NIL
C	∞	NIL
D	∞	NIL

(1) $S = \phi$

(2) $Q = V[G]$

(3) while $Q != \phi$

u = extract-min Q

$S = S \cup \{u\}$

for each $v \in Adj[u]$ & not in Q

if $d[v] > d[u] + w$

$d[v] = d[u] + w$

$\pi[v] = u$

→ algorithm to find shortest path.

b/w any two parts vertices of a graph.

→ It uses greedy approach.

Assumptions

→ two sets → (1) included all vertex

(2) Not included vertex

Looking a distance + weight for adjacent vertex

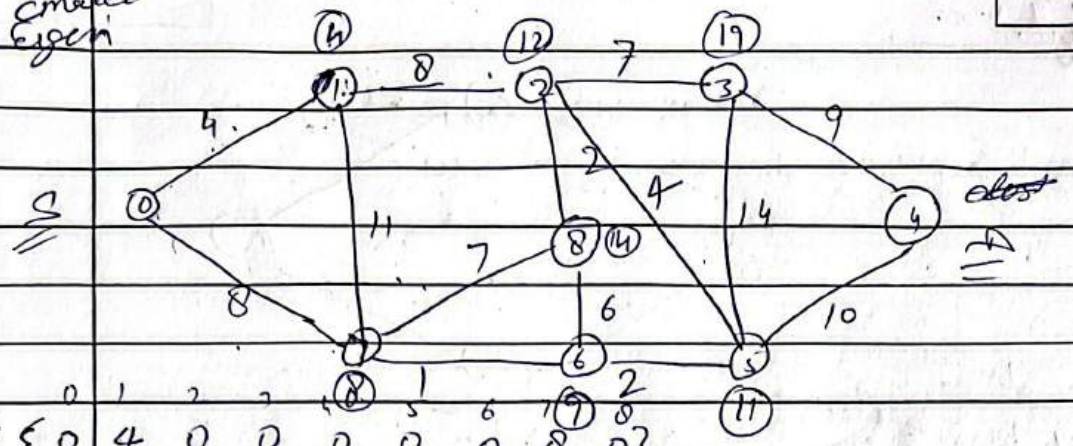
Scanned with CamScanner

Ompl → open motion planning library → It is based on sampling based motion planning

→ Boost

→ Smalice

→ Eigen



{{0 4 0 0 0 0 0 8 0},

{4 0 8 0 0 0 0 11 0},

{

Latombe's Books

Robot motion planning & control

1. introduction

→ grid based approach

→ notion of configuration space of a mechanical system.

→ larger robot has two controls (linear & angular velocities) while it moves in a 3-dimension configuration space.

→ constraints due to the obstacles are expressed directly in the manifold of configuration, nonholonomic constraints deals with the tangent space.

→ Obstacle avoidance adds a second level of difficulty. At this level we should take into account both the constraints due to the obstacles & the nonholonomic constraints linking the parameter derivatives.

→ Nonholonomic motion planning may be consider as the problem of planning open-loop controls; the problem of the feedback control is the purpose of Deluca Oriolo-Samson's chapter.

2. Controllability of mobile Robots:- → state precisely

(2.1) Controllabilities:-

n- dimensional manifold

$\mathcal{U}$- class of function of time & taking their value in some compact sub-domain $K \subset \mathbb{R}^2$.

The control system Σ considered differential system

such that $\dot{X} = f(X)u + g(X)$

control of the system.

i^{th} column of the matrix $f(x)$ is a vector field denoted by f_i. $g(x)$ is called drift.

→ An admissible trajectory is a solution of the differential system with given initial & final condition & u belong to $\mathcal{U}$

Σ is locally controllable from X if the set of points reachable from X by an admissable trajectory contains a neighborhood of X. It is small time controllable from X if the set of points reachable from X before a given time T contains a

neighborhood of X for any T.

→ small-time controllability clearly implies local controllability. (LARC) → Lie Algebra rank condition.

(2.2) mobile robot: from dynamics to kinematics:

modeling mobile robot with wheels on control system may be addressed with a differential geometric point of view by considering only the classical hypothesis of "Rolling without slipping" such modeling provides directly kinematic models of the robots.

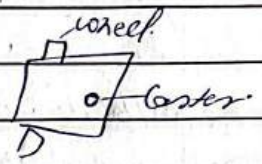
kth coordinates of $[f, g]$ is

$$[f,g][k] = \sum_{i=1}^{n} \left(g[i] \frac{\partial}{\partial x_i} f[k] - f[i] \frac{\partial}{\partial x_i} g[k] \right)$$

Let's consider two system: 1) A two driving wheel mobile robot

11) Car.

1) Two-driving wheel mobile robots:



→ acceleration of each being controlled by an independent motor.

→ stability of platform is ensured by casters.

→ The reference point of robot is the midpoint of the two wheels.

→ y coordinates with respect to a fixed frame are denoted by (x, y); the main direction of the vehicle is the direction θ of the two driving wheels.

→ l → distance b/w driving wheels & dynamic model.

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v}_1 \\ \dot{v}_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{2}(v_1+v_2)\cos\theta \\ \frac{1}{2}(v_1+v_2)\sin\theta \\ \frac{1}{l}(v_1-v_2) \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

v_1 & v_2 → respective wheel speeds.

→ This 5-dimensional system is not small-time controllable from any point (due to presence of drift & to the bounds on u_1 & u_2)

C^1

then
$$v = \frac{1}{2}(v_1 + v_2)$$
(omega) $$\omega = \frac{1}{b}(v_1 - v_2)$$
we get kinematic model which is expressed as the following 3dimensional system
$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix} v + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \omega$$

This bounds on v_1 & v_2 induce bounds v_{max} & ω_{max} on the new controls v & ω.

→ apply LARC evaluation show that it is small time controllable from everywhere.

note that v & ω should be C^1.

ii) Car-like robots:- Car has two control (i) accelerator
(ii) steering wheel.

→ Reference point with the coordinates (x, y) is the mid point of the rear wheels.

→ assume, the distance b/w rear & front axles is 1.

→ ω as the speed of the front wheels of the car &

(zeta) ζ is the angle b/w front wheels & the main direction θ of the car.

→ mechanical constraints imposes $|\zeta| \leq \zeta_{max}$ & consequently a minimum turning radius.

→ The dynamic model of the car is
$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\omega} \\ \dot{\zeta} \end{pmatrix} = \begin{pmatrix} \omega\cos\zeta\cos\theta \\ \omega\cos\zeta\sin\theta \\ \omega\sin\zeta \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ D \end{pmatrix} u_1 + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} u_2$$
with $|u_1| < u_{1max}$, $|u_2| \leq u_{2max}$.

The 5-dimensional system is not small time controllable from everywhere.

Scanned with CamScanner

→ first simplification is consider ω as a control.
now it gives 4-dimensional system:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\tau} \end{pmatrix} = \begin{pmatrix} \cos\tau \cos\theta \\ \cos\tau \sin\theta \\ \sin\tau \\ 0 \end{pmatrix} \omega + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} u_2$$

Apply LARC condition shows that the small time controllability from everywhere → ω should be $\mathbb{R}$ (!).

→ The mechanical constraint $|\tau| \leq \tau_{max} \leq \pi/2$ appears as an "obstacle" in $\mathbb{R}^2 \times (S^1)^2$.

→ Let's ~~consider~~ don't ~~consider~~ care the direction of the front wheel.
then $v = \omega \cos\tau$ & $\omega \sin\tau$
the get 3-dimensional control system:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix} v + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \omega$$

v & ω are C^1 & their values are bounded.

→ Guiding wheel robot in the feasible path
→ Car should have a curvature lesser than 1.

→ A last simplification consist in putting $|v| \equiv 1$ & even $v \equiv 1$.
eg.

→ On the shortest paths in the plane with bounded curvature such system will be called Reeds & Shepp's car & Dubin's car.

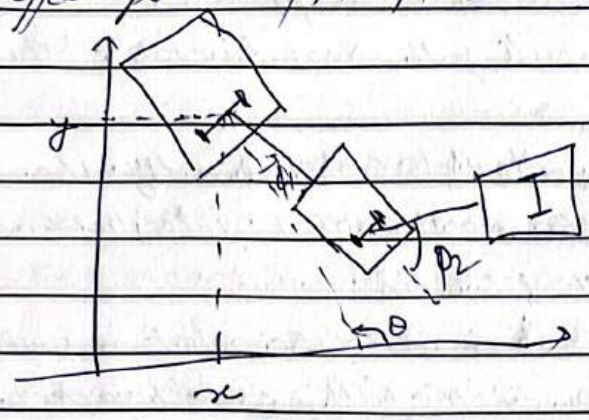
(I) → The admissible control domain of Reeds & Shepp's car is symmetric → LARC condition shows that it is small time control from everywhere.

(II) → Dubin's car is a system with drift → it is locally controllable but not small time controllable from everywhere.
To go from $(0,0,0)$ to $(1-\cos\varepsilon, \sin\varepsilon, 0)$ with dubin's car takes at least $2\pi - \varepsilon$ unity of time.

2.3 Kinematic model of mobile robot with trailers.

→ two-driving wheel mobile robot pulling two trailers. each trailer is hooked up at the middle point of the rear wheels of the previous one.

→ distance b/w reference point of the trailers is assumed to be 1.



kinematic model is

$$\dot{X} = f_1(X)v + f_2(X)\omega$$

where $X = (x, y, \theta, \psi_1, \psi_2)^T$

$f_1(X) = (\cos\theta, \sin\theta, 0, -\sin\psi_1, \sin\psi_2 - \cos\psi_1 \sin\psi_2)^T$

$f_2(X) = (0, 0, 1, 1, 0)^T$

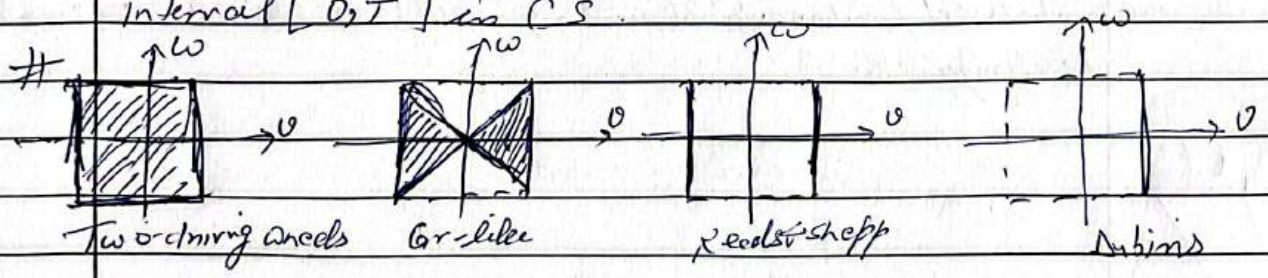
2.4 ~~Admissible path & trajectories~~

~~Constrained paths & trajectories~~

Let CS → Configuration space of mobile robot.

→ Inequal

a trajectory is a continuous function from some real interval $[0, T]$ in CS



Here appears

→ ~~Admin~~ Admissible paths for Dubins car are admissible for Reeds&Shepp's car; Admissible path for Reeds&Shepp's car are admissible for the car-like robot; ~~Admissible path for car-like robot are admissible for the two-driving wheel mobile robot~~

From path to trajectories:- The goal of nonholonomic motion planning is to provide collision-free admissible path in the configuration space of the mobile robot system.
Obstacle avoidance imposes a geometric point of view that dominates the various approaches address the problem.
The motion planner computes path which have to be transformed into trajectories.

→ A block box module allows to control directly the linear & angular velocities of the mobile robot. velocities & accelerations are of course submitted bounds.

* The more the kinematics model of the robot is simplified, the more the transformation of the path into a trajectory should be elaborated.

→ Let us consider for instance an elementary path consisting of an arc of a ~~circle~~ circle followed by a tangent straight line segment. Due to the discontinuity of the ~~arc~~ ~~arc of the circle~~ curvature of the path at the tangent point, a two driving-wheel mobile robot should stop at this point; this resulting motion is clearly not ~~sta~~ satisfactory. This critical point may be overcome by "smoothing" the path before computing the trajectory.

* On the other hand, some approaches address simultaneously the geometric constraints of obstacle avoidance, the kinematic & the dynamic one: this is so called "kinodynamic planning problem".

Combined from robot motion problems graph search & discretization techniques.

8. Path planning & small-time controllability: Path planning raises two problem: the first one addresses the existence of a collision-free admissible path while the second one addresses the computation of such a path. First is the decision problem while second one is the complete problem.

* Dubins' robot:- This is simplest example of system which is locally controllable & not small time controllable. For this system the decision problem is solved when a robot is reduced to a point.

* Reeds & Shepp's car like & two-driving wheel robots:- These systems are small time controllable.

* Mobile robot with trailers:- small time controllable, for both of them the decision problem is solved

Small time controllability:

Steering methods:- Steering method is an algorithm that solve the path planning problem without taking into account the geometric constraints on the state. Even in the absence of obstacle, computing an admissible path b/w two configurations of ~~the~~ a nonholonomic system is not an easy task.

* main approaches which have been applied to mobile robotics.

1) From vector field to effective paths:-

By Bellaiche-Jean-Risler's given combinatorial & geometric point of view of the path planning problem:-

- choose → x on a manifold & a vector field f defined around this point.

exactly one path $\gamma(t)$ starting at this point & following f.

so $\gamma(0) = x$

$\dot{\gamma}(t) = f(\gamma(t))$.

$\gamma(1) = \underset{\hookrightarrow \text{ a function on the manifold}}{\xi . x}$

$e^{f+g} \neq f+g$ for unit time

$[f,g] \neq 0$

$\alpha f + \beta g$

$\alpha f \neq \beta g$

Let say $f_1, \ldots, f_n$ → vector fields defined by neighborhood N of point X.

Two coordinate system called first & second normal coordinates.

$(\alpha_1, \ldots, \alpha_n) \mapsto e^{\alpha_1 f_1 + \cdots \alpha_n f_n} \cdot X$

This is associated to → $\{f_1, \ldots, f_n\}$.

Campbell-Baker-Hausdorff formula:

states precisely the difference b/w the two systems for small τ

$$e^{\tau f} \cdot e^{\tau g} = e^{\tau f + \tau g - \frac{1}{2}\tau^2 [f,g] + \tau^2 \epsilon(\tau)}$$

where $(\epsilon \tau) \to 0$ when $\tau \to 0$.

$$\tau^2 \epsilon(\tau) = \sum_{k=3}^{\infty} \tau^k c_k$$

This formula tell us → How a small time nonholonomic system can reach any point in a neighborhood of a starting point.

This for → explicitly computing admissible paths in a neighborhood of a point.

(4.2) nilpotent system & nilpotentization.

for mobile robot → ~~let~~ Lets consider the control system:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\varphi} \end{pmatrix} = \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \\ -\sin\varphi \end{pmatrix} u_1 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} u_2$$

$$= f_1 u_1 + f_2 u_2$$

θ → direction

φ → angle of the trailer ~~with~~ w.r.t mobile robot.

Coordinate vector field of $f_3 = [f_1, f_2]$

$f_4 = [f_1, [f_1, f_2]]$

$$f_3 = \begin{pmatrix} \sin\theta \\ -\cos\theta \\ 0 \\ \cos\varphi \end{pmatrix}, \quad f_4 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

easily check that $\{f_1, f_2, f_3, f_4\}$ is a basis of the tangent space at every point of the manifold $\mathbb{R}^2 \times S^1 \times S^1$

let

$$x_0 = (x_0, y_0, \theta_0, \varphi_0)$$

$$x = (x, y, \theta, \varphi)$$

$$\Delta x = x - x_0, \quad \Delta y = y - y_0, \quad \Delta\theta = \theta - \theta_0, \quad \Delta\varphi = \varphi - \varphi_0$$

$$y_1 = \cos\theta_0 \Delta x + \sin\theta_0 \Delta y$$

$$y_2 = \Delta\theta$$

$$y_3 = \sin\theta_0 \Delta x - \cos\theta_0 \Delta y$$

$$y_4 = \sin(\varphi_0 - \theta_0)\Delta x + \cos(\varphi_0 - \theta_0)\Delta y - \Delta\theta + \Delta\varphi.$$

4.3 steering chained form systems.

(4.4) Steering flat systems.

(4.5) steering with optimal control.

(5) Nonholonomic path planning for small time controllable system.

Lets consider mobile robot go from $(0,0,0)$ to (x,y,θ) in same configuration.

→ robot first execute a pure rotation to the configuration $(0, 0, \arctan\frac{y}{x})$ then it moves along a straight line segment to $(x, y, \arctan\frac{y}{x})$

(5.1) Traditional steering methods accounting for small-time controllability.

let d_{AS} be following distance on the configuration space CS

$$d_{AS}(x^1, x^2) = \sum_{i=1}^{n} |x_i^1 - x_i^2|$$

Scanned with CamScanner

Geometry of Nonholonomic Systems

(i) Symmetric control systems: an introduction.

(ii) Control systems & motion planning:

Regardless of regularity hypothesis, control systems may be introduced in the today.

(i) $\dot{x} \in V_x$

$V_x \to$ for every x, some subset of the tangent space T_xM, or in a parametrized way as $\Rightarrow \dot{x} = f(x,u)$

where for every x the map $u \mapsto f(x,u)$ has V_x as its image.

→ Condition of first kind occur as linear constraints on the velocities, such as rolling constraints, as well in free movement the classical object of study in mechanics, as in the case of systems propulsed by motors.

(ii) Second kind may represent the action of "actuators" used to move the state of the system in the configuration space.

Two actuators are represented by

$\dot{x} = f_1(x) \to$ ~~vec~~ Convect vector field f_1

$\dot{x} = f_2(x) \to$ " " " f_2

Configuration space — c-space

State —
state-space — it is set of all that contains all the possible configuration of the robot it can continuous or discrete.

→ Build a graph (roadmap)

Stage transform → ~~odom, base link~~

↓

static transformation make relation b/w odom & map

map → odom frame

/goal

↓

/Costmap → global planner → /Path

/Cost map-update → global planner

↑

/tf

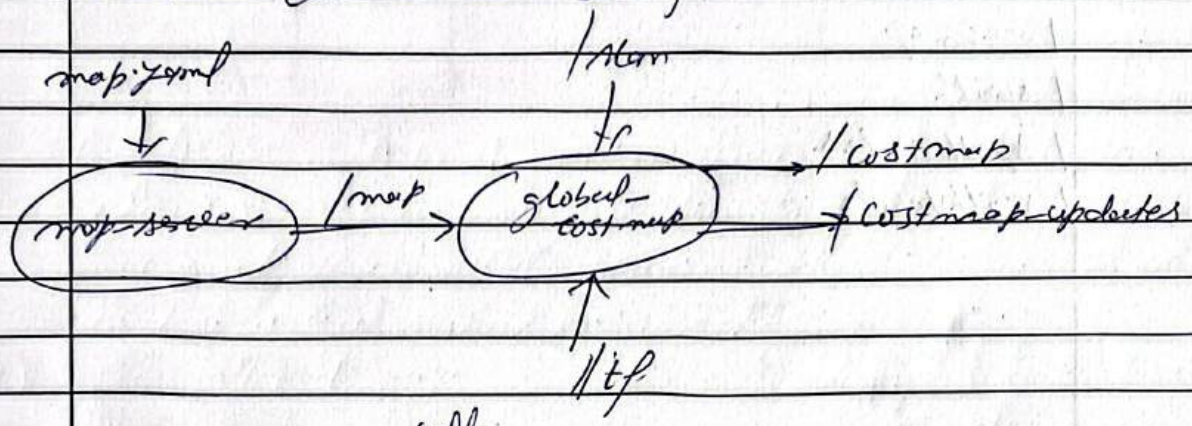
RVIZ → Publish → /move_base_simple/goal.

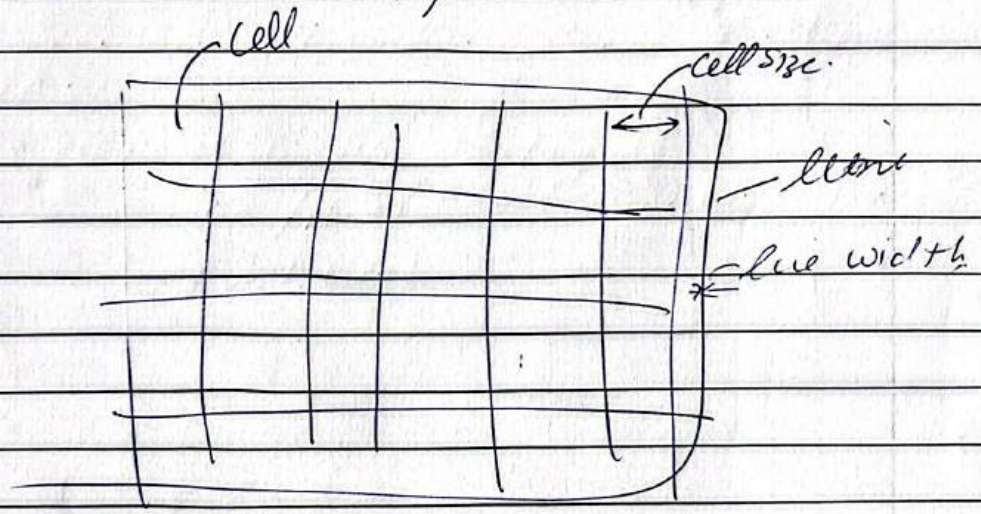
OCC-map — Cost map

map-server → Publish a map → occupancy grid (nav_msgs)
in the topic /map

0 → free
100 – occupied
-1 → unknown.

Cost-2d-model (global-costmap) from /map topic





grid

ROS TOPICS

/clicked_point →

/initialpose → initialpose intermal from RViz

/map → value from D to/co

/map_metadata → resolution, width=80, height=80, origin, position, orientation

/map_updates → localmap updates for specific robot/map msgs/occupancy grid

/move_base_simple/goal

/move_base_simple/start

/occ_map

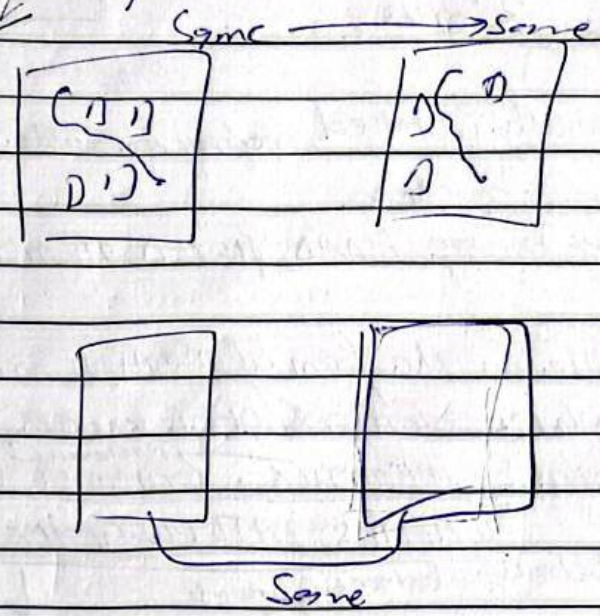
/spath

/spathNodes

/spathVehicle

Scanned with CamScanner

path found at $t = 0.5$
with us + $C_p = 14.72$



19. Cost
79 ms