Robot Path Planning with Differential Constraints.

Monica Patel

***Abstract*— Even the simplest robot model is subject to differential constraint. In robotics most problems involve differential constraints that arise from kinematics and dynamics of the robot. In order to plan a collision free path for robot which it can successfully follow its important to consider these constraints while planning. This project studies a very popular randomized path planning technique called RRT-Planning and apply it for a simple robot model with differential constraints. In order to deal with differential constraints a new sampling technique is studied in the report which is based on feasibility sets. Report also discusses in details theory of reachability sets and discusses reachability sets for simple car motion. Properties of RRTs and how the differential constraint problem affect those properties are also discussed in the report. Finally report also discusses the scope of improvement in the studied algorithms.**

I. INTRODUCTION

In the field of mobile robotics, autonomous aerial, underwater and ground robot have gained considerable popularity. Most of these robots are required to explore the unexplored territory. In order to do this agent must be able to plan a collision free path through the given environment. As the field progresses the robot system are becoming more complex, having higher degrees of freedom. Therefore a desirable property of a planner is to be able to adapt itself for higher dimensional planning without increasing the computation time significantly. One of the sampling based planning strategy has been proven particularly affective for such systems. RRTs [6] have been used for broad range of planning problems ranging from simple robots to high dimensional manipulators.

This report studies limitation of RRT in the system with differential constraints, and discusses a new adaptive sampling based algorithm to deal with such problem. In order to visualize and study the working of this algorithm a low dimensional system of a car model is chosen. The task of planner is to generate collision free trajectory from start location of car in environment to go goal location. While doing this planner should take into account differential constraints imposed by dynamics of the car.

II. RELATED WORK

There is a lot work on modification of RRT to make it suitable to account for differential constraints. Like one obvious solution is to choose better distance metric according to system requirement, like optimal cost to go [7]. Or rather than designing the distance metric making system learn one [8]. Optimal kinodynamic motion planning approach incorporates differential constraints in RRT* [3]. LQR-RRT* finds optimal plans in domains with complex or under actuated dynamics without requiring domain specific design choices [4]. The dynamic domain RRT [9] deals with futile oversampling by altering the size of Voronoi regions for those nodes closer to obstacle. Many other planners are proposed like Poli-RRT*, A*-RRT*, Theta*-RRT [10] [11] [12]. While all these algorithms deals with different aspects of RRT, they introduce significant change the RRT algorithm. The method studied in the report is very similar to RRT algorithm and fairly simple to implement to deal with differential constraints of system.

III. NAVIGATION TASK AND ROBOT MODEL

For the purpose of experiment for the project a simple car model is considered. For planning and navigation robot agent is assumed to have complete knowledge of the world. The world of robot consist of regular shaped obstacles which it has to avoid while navigating from start location to goal location.

If we assume a car as rigid body moving in 2D plane its configuration space will be $C = R^2 x S^1$. The simplest car dynamics can be described by following system of equations.

$$\dot{x} = u_s sin\theta, \; \dot{y} = u_s cos\theta, \; \dot{\theta} = \frac{u_s}{L} tan(u_\phi); \qquad (1)$$

Where u_s and u_ϕ are components of two dimensional action vector u = (u_s, u_ϕ). For specifying U, set of actions of the form u = (u_s, u_ϕ), lets suppose $u_s \in R$ is possible. As we know a car cannot slide side ways because back wheels would have to slide instead of roll. The interval $[\frac{-\pi}{2}, \frac{\pi}{2}]$ is sufficient for steering angle, u_ϕ. If we assume that car is moving with safe speed in order to navigate a bond such as $|u_s| \leq 1$ can be placed on speed. Therefore to summarize the specification of U, U $= [-1, 1] \times (-\phi_{max}, \phi_{max})$.

IV. ROBOT PATH PLANNING WITH RRT

Steve M. LaVelle introduced a randomized data structure that was designed for broad class of path planning problems [2]. Working of vanilla RRT is fairly simple. Given a start location, $x_{init} \in X$ and goal location

$x_{goal} \in X$ where X is state space of the robot, the planner incrementally builds sets of trajectory through X. While building these trajectories planner should avoid the fixed obstacle region $X_{obs} \in X$. At each iteration of the algorithm a random point from obstacle free region of state space is selected, $x_{rand} \in X_{free}$ where $X_{free} = X \setminus X_{obs}$. Then algorithm finds the closest node in the tree to this sample, x_{near}. The closest point is determined using the predefined distance metric. The the planner extend from x_{near} towards x_{rand} using fixed step size determined by input of system and this new node is added to the tree.

V. PROBELM OF DIFFERENTIAL CONSTRAINTS

The pattern in which RRT expands its nodes depends upon two factors - sampling distribution, and the distance metric, which makes RRT algorithm highly sensitive to distance metric. Most commonly the uniform sampling distribution and Euclidean distance metric is used for expansion. But when the system is subject to differential constraints (kinematic or dynamic) Euclidean metric ca be poor representation of true distance.

Formally defining the problem of differential constraint, Let X denote the state space that applies to Kinodynamic system and let C represent the general configuration space. We want to find time-varying control inputs u(t) in U for t in [0,T] that drives the system from initial state x-start to x-goal in finite time T. The resulting trajectory must be collision free and satisfy differential constraints according to the state transition function

$$\dot{x} = f(x,u) \tag{2}$$

VI. REACHABILITY- GUIDED RRT

We saw that the vanilla- RRT is very sensitive to the distance metric and that when a system is subject to differential constraints Euclidean distance is not very good representation of nearest neighbor. Algorithm introduced by Alexander Shkolnik, Matthew Walter and Russ Tedrake called Reachability Guided Sampling for RRT (RG-RRT) [1] reduces this sensitivity of the vanilla RRT planner on distance metric. RG-RRT ensures that any node added to the tree must make progress towards given sample. The main idea is that when a tree grows in system with differential constraint its often not able to expand in certain region. RG-RRT implements an adaptive sampling technique which constantly changes the region where tree can expand.

A. RG-RRT Algorithm

Algorithm 1 T ← RG_RRT(x_{init})

```
T ← InitialiseTree()
T ← InsertNode(x_init,T)
For k = 1 to k = K do
    x_rand ← RandomState()
    (x_near, x^r_near) ← NearestState(x_rand,T)
    While x_near =  do
        x_rand ← RandomState()
        (x_near, x^r_near) ← NearestState(x_rand,T)
    End While
    u ← solveInput(x_near, x^r_near, x_rand, T)
    x_new ← NewState(x_near,u)
    T ← InsertNode(x_new,T)
End For
return T
```

The algorithm is very similar to the vanilla RRT, except for, when the distance metric selects the nearest node in the tree to expand towards the random point, it not only looks for points in the tree but also at all the points in *Reachable Sets* of all the nodes in the tree. If the closest Reachable point is closer to the sample than the closest node of the tree, then both this reachable point, x^r_{near}, and its corresponding parent node, x_{near}, are returned. Otherwise, if the closest node of the tree is nearer to the sample than any Reachable point, the function returns an empty point pair, in which case the RG-RRT throws this sample away, and draws a new sample from the state space and repeats the process. Meaning the RG-RRT allows samples only from Voronoi regions for which the differential constraints permit the expansion of the node towards the sample. The notion of reachable set is explained in following subsection.

B. Reachable Sets

Definition: For a state $x_0 \in X$ and a finite local integration time step Δt its *Reachable set*, $R_{\Delta t}(x_0)$ is defined to be the set of all points that can be achieved from x_0 in finite time Δt according to differential constraint equation of the system and set of control set U.
Recalling the car model from previous section which can only drive forward and delta t is small enough that max steering angle is $\frac{\pi}{2}$ for an arbitrary configuration, the time-limited reachable set appears as shown in Figure 1

In RG-RRT even if samples were drawn from complete state-space by throwing away the sample where reachable points were further away than nearest point in node algorithm made sure that only the subset of

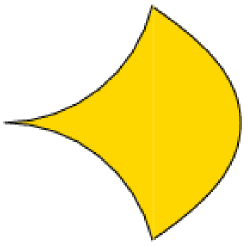

Fig. 1: Reachability Set for car model

the state space was used for expansion. Thus algorithm identifies not suitable regions for expansion, since it was impossible to grow tree into them in Δt time under differential constraints. Which vanilla RRT fails to identify.

VII. PROPERTIES OF RRT AND EFFECTS DUE TO REACHABILITY GUIDED SAMPLING

RRT is probabilistically complete under the condition that state space is randomly sampled. Since in RG-RRT state space is still randomly sampled using uniform distribution this property is intact. RG RRT also preserves other desirable properties of RRT like biased exploration towards unexplored region of state space, simplicity of the method and property of remaining connected even with minimal edges.

VIII. RESULTS OF EXPERIMENT

As we can see from Figure 2 vanilla RRT expands in every direction without considering that the system cannot steer more than 90 degrees. Figure 3 shows working of RG-RRT and we can see from expansion pattern that tree expands only the region where differential constraints are satisfies. Good example of RG-RRT following the differential constraints can be seen from Figure 4 where there is branch closer to goal but its not able to expand since its not permissible by the differential constraints of the system but rather a branch from farther way expands more quickly towards goal. From Figure 5 we can see that RG-RRT does not necessarily gives the shortest path to goal. But the first path that it finds.

IX. CONCLUSION AND DISCUSSION

- RG-RRT is probabilistic sampling based algorithm, which has drawback of running indefinitely if the path from start goal does not exists.
- From results we saw that RG RRT does not give the shortest path, which can be implementing by

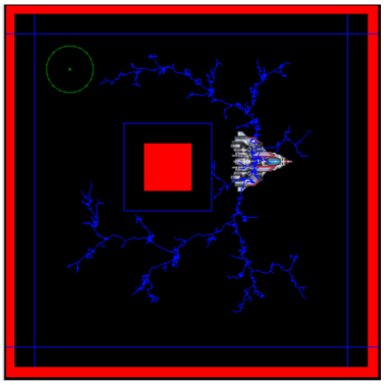

Fig. 2: Working of vanilla RRT

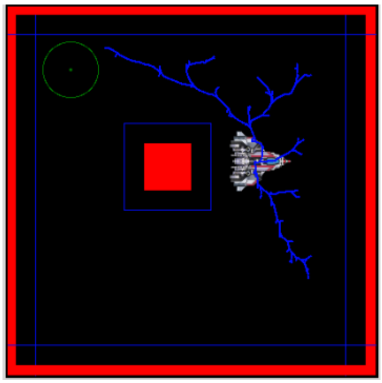

Fig. 3: Working of RG-RRT

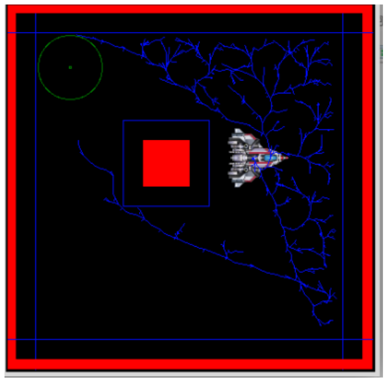

Fig. 4: Differential Constraints followed

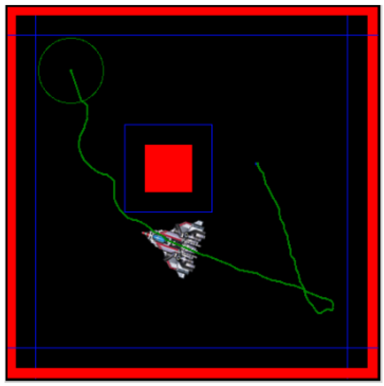

Fig. 5: Not the shortest Path

adapting reachability based adaptive sampling in RRT* or RRG algorithms.

- Main working of RG-RRT depends on finding the reachability set of the system. Which for simple models like car is easy to obtain or visualize geometrically, but for complex system it is a challenge. There is a machine learning approach proposed by Ross E Allen et al [5] which analyze reachability set of system in real time. Similar methods can be adapted to find reachability set of complex systems.

REFERENCES

[1] Alexander Shkolnik, Matthew Walter, and Russ Tedrake, Reachability-Guided Sampling for Planning Under Differential Constraints.

[2] Steven. M. LaVelle, Planning algorithm.

[3] Sertac Karaman, Emilio Frazzoli Optimal Kinodynamic Motion Planning using Incremental Sampling-Based methods.

[4] Alejandro Perez, Robert Platt Jr., George Konidaris, Leslie Kaelbling and Tomas Lozano-Perez, $LQR-RRT*$: Optimal Sampling-Based Motion Planning with Automatically Derived Extension Heuristics.

[5] Ross E. Allen, Ashley A. Clark, Joseph A. Starek, A machine learning approach for real-time reachability analysis.

[6] Steven. M. LaVelle, Rapidly Exploring Random Trees: A new tool for path planning.

[7]] E. Frazzoli, M. A. Dahleh, and E. Feron, Real-Time Motion Planning for Agile Autonomous Vehicles, Journal of Guidance, Control, and Dynamics, vol. 25, no. 1, pp. 116129, JanuaryFebruary 2002.

[8] P. Cheng, Sampling-based motion planning with differential constraints, Ph.D. dissertation, University of Illinois, Urbana-Champaign, Urbana, IL, August 2005.

[9] A. Yershova, L. Jaillet, T. Simeon, and S. M. LaValle, Dynamic domain RRTs: Efficient exploration by controlling the sampling domain, in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Barcelona, April 2005, pp. 38563861.

[10] Matteo Ragaglia, Maria Prandini, Luca Bascetta, Poli-RRT*: Optimal RRT-based planning for constrained and feedback linearisable vehicle dynamics

[11] M. Brunner, B. Bruggemann, and D. Schulz, Hierarchical rough terrain motion planning using an optimal sampling based method, in Int. Conf. on Robotics and Automation (ICRA), Karlsruhe, Germany, 2013.

[12] Luigi Palmieri, Sven Koenig, Kai O. Arras, RRT-Based Nonholonomic Motion Planning Using Any-Angle Path Biasing.