PEA: Perception and Energy Aware Path Planning in 3-D Environments

Anshika
Department of Computer Science and Engineering
Indian Institute of Technology Ropar
Punjab, India
2021csb1069@iitrpr.ac.in

***Abstract*—This paper presents a novel method for real-time path planning in a dynamic environment. A real-time sampling-based approach is utilized. The approach is based on variants of the RRT* algorithm. The proposed method focuses on energy-aware approach in unmanned vehicles and vehicles working on batteries. We analyze and compare the proposed algorithm for UAVs and autonomous vehicles operating on conventional fuel as well as Li-ion batteries based on the parameters of time, expanded nodes, and energy awareness. In several instances, UAVs are limited by several metrics that include flight distance/duration because of obstacle collisions, the shortage of onboard batteries, or positioning errors owing to onboard sensors and loss of signal from global navigation satellite system (GNSS). The proposed method, Perception and Energy-Aware RRT* (PEA-RRT*) is demonstrated in navigating a dynamic environment with moving obstacles. The method finds paths to new targets considerably faster when compared to other perception and energy-aware algorithms, and that too in a dynamic environment.**

***Index Terms*—Real-time path planning, RRT*, dynamic environment, energy-aware, perception-aware**

I. Introduction

Path planning in dynamic environments is a challenging problem encountered by truly autonomous mobile robots. State-of-the-art real-time path planning algorithms basically have to balance the trade-off between shorter search times and the goodness of the path.
The path planning problem gets even more complex in autonomous vehicles or unmanned vehicles where robots have to directly interact with human life. Conventional algorithms do not perform satisfactorily in real-world texture-less areas. Perception-aware algorithms are capable of avoiding texture-less regions and areas with sufficient texture. Not only texture, but overall energy is also crucial in deciding the overall goodness of an algorithm.
The unconstrained use of algorithms reduces a fuel-operated unmanned vehicle's potential, as not all of the robot's resources are required at all times. For example, in autonomous navigation, performing localization with all sensors always switched on is wasteful, as most robots do not need or use all sensors for most of their operating time (e.g., redundant or situational sensors) (Lee and Song 2004) [7].
In this paper, an approach is proposed where unmanned vehicles work efficiently pertaining to perception awareness, energy awareness, and dynamic real-world environments.

Identify applicable funding agency here. If none, delete this.

II. Related Work

In this section, firstly, relevant literature regarding sampling-based algorithms for path planning is reviewed, followed by the link between active perception and autonomous, unmanned vehicles. Then relevant planning strategies attempting to save energy are discussed and reviewed.

A. Real-time path planning using RRT* variants

Rapidly-exploring Random Trees combine random sampling of configuration space with biased sampling around the goal configuration to efficiently provide solutions to high-dimensional spaces. However, RRTs are asymptotically not optimal, and when connections are set once between the nodes, no rewiring is done. [5]
RRT* allows rewiring, and the output path advances to the shortest path as number of expanded nodes increases. Though asymptotically more efficient than RRT, it is still inefficient in large environments. RRT-Connect [6] involves no differential constraints, it is based on Connect heuristic attempting to move over a large distance and growth of RRTs from both the start and the goal. Dynamic Rapidly-exploring Random Trees is another variant of RRT which is basically as replanning algorithm for repairing Rapidly Exploring Random Trees when changes are made to the configuration space. In this paper an algorithm combining the efficient features of variants of RRT and RRT* for efficient planning in dynamic environments.

B. Perception-aware planning problems

The motion and path followed by an autonomous vehicle have a significant impact on state estimation algorithms. Perception-aware planning solutions aim to achieve path-planning and state estimation with regard to the unmanned vehicle's state uncertainty during planning. This leads to the need for a fully autonomous system that can deal with real-world uncertainties.
Early works describe the importance of active vision sensors. The perception awareness problem can be viewed as Partially Observable Markov Decision Processes that is a framework

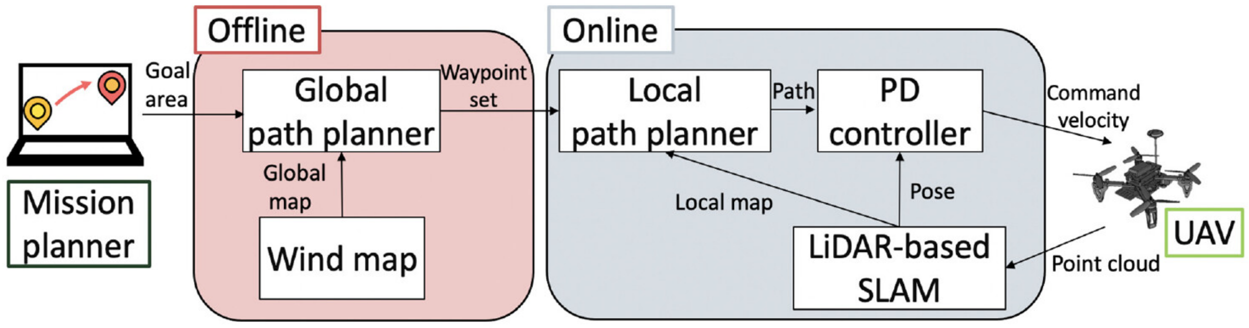

Fig. 1. Schematic overview of autonomous navigation systems for UAVs. [2]

to model problems of planning under uncertainty. Approaches based on Belief-space planning and receding-horizon planning strategies have emerged recently.
Currently, the perception-aware receding-horizon approach evaluates trajectories in terms of landmark concentration. Further work has been done on incorporating semantics to evaluate landmarks and candidate areas for navigation [2]

C. Energy-aware Planning domain

The advances in energy efficiency in path-planning domain deal with various methods to minimize energy consumption through algorithms without considering hardware-based actions like enabling and disabling switches. One of the widely used strategies includes avoiding regions that require more effort.The approach in (Datouo et al. 2017) achieves a 7.5% reduction in battery energy draw, while (Mejri et al. 2017) achieve a reduction of 46% .
In comparison to the work proposed in this paper, none of the related studies provide energy-consumption awareness along with perception awareness in dynamic environment. [4]

III. Problem Description

The problem considered in this paper is to reach a pre-defined goal while considering perception quality and energy awareness . Let D denote the dynamic environment which is our work space. D is a subset of R^3. D contains dynamic obstacles denoted by D_{dyobs} and static obstacles denoted by D_{stobs}. D_{free} denoted the free space i.e. D- $\{D_{dyobs}$ U $D_{stobs}\}$. The search tree T propagates in D_{free}. The set of planned nodes is represented by (x_0, x_1, ... ,x_m) where m is the pre-decided limit.
It is assumed that in D, some areas exist such that they are more suitable for localization than others and our objective is to take those areas into consideration as much as possible while planning the path. The aim is to generate smooth, collision-free trajectories in a receding horizon fashion while considering the constraints of keyframe-based Visual-Inertial Odometry (VIO) systems.
So, real-time path-planning problem in dynamic environments as follows: We want to find a path, i.e. (x_0, x_1, ... ,x_m) such that this path should lie in D_{free} and in order to find x_m, the heuristic used is the PEA-heuristic that attempts to move over a longer distance while being aware of the surroundings and the energy consumption. For the UAV, we assume that UAV is equipped with 3D LiDAR sensor on its bottom which assures 360 degree field of view. In case of UAVs, we assume that final path only includes spatiotemporal wind information in GNSS-denied scenario. Most of the information regarding the environment is unavailable until the Unmanned Vehices actually flies or operates and the onboard sensors acquire that information. In UAVs case that can include unpredicted wind conditions, destructed buildings, etc. So, in this research, the usage of wind information is in global planning whereas local planning is based on 3D geometrical information as in R. Takemura et al [1] .
Owing to the limitation of the LiDAR's scanning range, the local planning is connducted based on the current pose and local map at each step, that is, the problem is solved in a receding horizon manner as in [8]

IV. Methodology

In this section, the methodology employed for the optimal algorithm PEA-RRT* is defined.

A. Modelling the Environment

Firstly the environment where we are going to test our algorithm is set up. As previously mentioned, PEA-RRT* is tested in two types of environments and agents.
The first system is UAV-system where the agent is UAV and the environment is three-dimensional aerial environment.
The second system where the algorithm is tested is the autonomous vehicle system. Here the agent is the autonomous vehicle and environment is the road where it travels.
The primary difference between both of these systems is the sensor types and their ranges that are used to model the environment and the agent.
UAV agent is modelled with LIDAR sensor with a 360 degree field of view.
Similarly, autonomous vehicle that travels on ground, also uses LIDAR sensor with different ranges and field of views. .

B. Comparison of RRT* variants

Though RRT* itself does not prove to be so optimal for UAVs and Unmanned vehicles, its variants prove to be

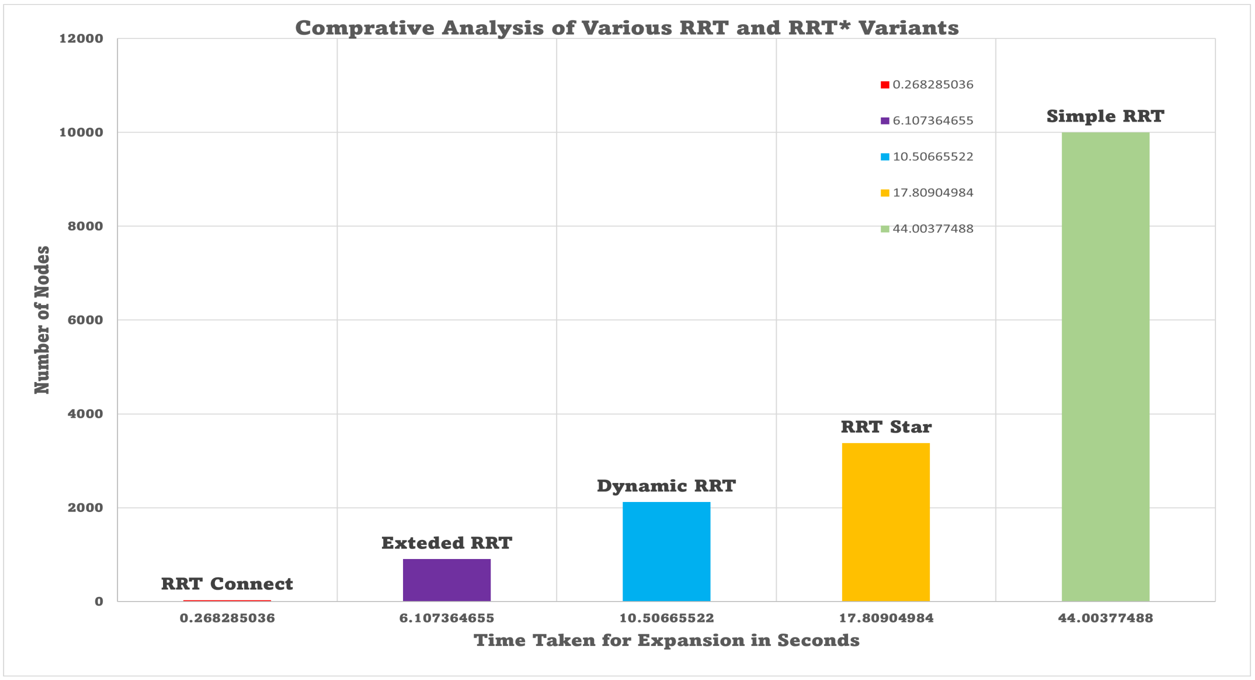

Fig. 2. Comparative analysis of various RRT and RRT* Algorithms

very optimal. A detailed analysis of these variants in 3-D environment has been done. These algorithms prove to be optimal than RRT* but the real challenge is its application in real world situations. Real world environments are complex and there are a lot of factors that have to be considered while path planning. Sometimes, the even the shortest path may not be the optimal one due to plenty of reasons that may include road texture-friction, fuel availability,etc. Fig 2. shows the results of simulations run in order to analyse the RRT* variants [1]

With precise knowledge of the problem and the detailed analysis of state-of-the-art algorithms, PEA-RRT* is proposed that aims to combine the benefits of the state-of-the-art algorithms while also handling complex real world issues.

V. Proposed Algorithm: PEA-RRT*

The proposed algorithm is built on top of RRT* algorithm. This variant of RRT* is implemented on D (as defined in Problem Description) containing obstacles and free space. The algorithm, as expected runs to find a search tree in D-{D_{dyobs} U D_{stobs}}. PEA-RRT* proceeds to work in bi-directional manner. As the start and goal node is predefined, PEA-RRT* starts expanding the tree from both the start and goal. This leads to not only optimization in time but also enhances the perception quality, thereby proving energetically efficient and total cost is also optimized as can be seen from comparison in table 3 as shown. The algorithm uses the cost function of any node n_i like in the traditional A* algorithm:

$$f(n_i) = g(n_i) + h(n_i) \tag{1}$$

where $g(n_i)$ is the cumulative energy consumption from the start node to the node n_i and $h(n_i)$ is the predicted energy

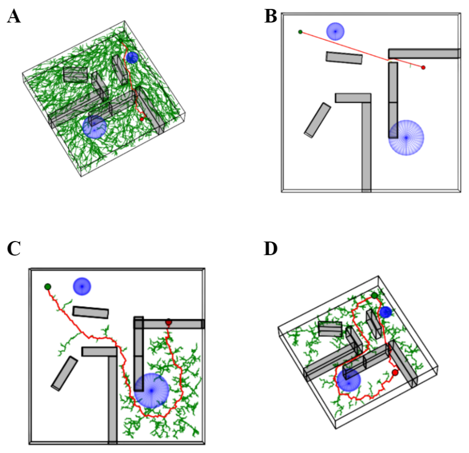

Fig. 3. Comparative analysis of various RRT and RRT* Algorithms: A. RRT* nodes expanded; B. RRT Connect nodes expanded; C. Dynamic RRT nodes expanded; D. RRT extended nodes expanded

consumption from n_i to the goal node. $g(n_i)$ and $h(n_i)$ are defined as follows, extending the model of Thibbotuwawa et al: [3]

$g(n_i)$ is defined in the following manner :

$$\begin{aligned} g(n_i) &= g(n_s, n_i) \equiv E(V_i) \\ &= \int_{t_s}^{t_i} \left(\frac{1}{2} C_D \mathrm{AD} (V_i)^3 + \frac{m^2 g^2}{D b^2 V_i} \right) \mathrm{dt}, \end{aligned} \tag{2}$$

whereas $h(n_{\mathrm{i}})$ is defined as follows:

$$\begin{aligned} h\left(n_i\right) &\equiv \min E\left(V_i\right)=E\left(V^*\right) \\ &=\int_{t_i}^{t_g}\left(\frac{1}{2} C_D \mathrm{AD}\left(V^*\right)^3+\frac{m^2 g^2}{D b^2 V^*}\right) \mathrm{dt}, \end{aligned} \tag{3}$$

D=density of air,
C_{D} = drag coefficient of UAV,
V_{i}= airspeed of UAV at n_{i},
A, b, m, g are cross-sectional area, the width,the mass of the UAV and gravitational acceleration. t_{s} and t_{g} are departure and arrival time at start and goal node respectively.
V^* is the airspeed such that the energy consumption from a node to the goal node is minimized.

$$V^* \equiv \arg \min_E \left(V_i\right)=\sqrt[4]{\frac{2 m^2 g^2}{3 C_D A D^2 b^2}}. \tag{4}$$

The proposed algorithm PEA-RRT* inputs the start and end goal and proceed to find a common fag and node in order to reach an optimal path. PEA-RRT* is defined as follows:

```
Algorithm: PEA-RRT*
Require: Map, UAV Specifications
Input: Start Position (start), Goal Position (goal)

1.  Initialize start and goal trees:
2.     $tree_{start}$ = InitTree(start)
3.     $tree_{goal}$ = InitTree(goal)
4.  goal_id ← -1
5.  while true do
6.     Expand the start tree:
7.        $tree_{start}$, goal_id ← PEA-plan($tree_{start}$, goal)
8.     Expand the goal tree:
9.        $tree_{goal}$, goal_id_goal ← PEA-Plan($tree_{goal}$, start)
10.    Check for the intersection of trees:
11.       common_node, common_flag ← CheckIntersection(
                                       $tree_{start}$, $tree_{goal}$)
12.    If common_flag is true then
13.       Construct the final path:
14.          final_path ← ConstructFinalPath( $tree_{start}$, $tree_{goal}$
                 ,goal_id, goal_id_goal, common_node)
15.       Optimize the final path:
16.          final_path ← OptimizeFinalPath(final_path)
17.       Return final_path
18. end while
```

The *CheckIntersection* function used within the PEA-RRT* algorithm. It's utility in PEA-RRT* is to check where the two expanding trees(one from start and the other from goal) intersect. It returns the common node where the intersect. It is defined as follows:

```
Function: CheckIntersection
Require: Nodes tree and $tree_{goal}$ from the two trees
Output: common_node, common_flag

  common_flag = false
  common_node = [ ]
  for i in 1 to size(tree, 1) do
    for j in 1 to size($tree_{goal}$, 1) do
      dist ← EuclideanDistance(tree(i, :), $tree_{goal}$(j, :))
      if dist < goal_diff then
        if SatisfiesConstraints(tree(i, :))
              and SatisfiesConstraints($tree_{goal}$(j, :)) then
          common_flag ← true
          common_node ← tree(i, 1:2)
          return common_node, common_flag
        end if
      end if
    end for
  end for
```

Lastly the PEA-plan algorithm is defined in the following manner:

```
Function: PEA-plan
Require: Wind speed at the current node $V_{wind}$
Input: $current_{tree}$, goal, wind_map, $\mathcal{V}_{th}$, $\Delta xth +$, $\Delta xth - \mathcal{V}$,
obstacle_map
Output: updated_tree, goal_id

  goal_id ← -1
  while true
    $current_{node}$, $current_{tree}$ ← ExtractMinCostNode( $current_{tree}$)
    if norm( $current_{node}$.position - goal) < $\delta$x
      goal_id ← $current_{node}$.id;
      return
    end
    $\mathcal{V}$i = getAirSpeed(Vwind, $\mathcal{V}$)
    Imax = getMaxErrDir( $current_{node}$)
    nq, dq ← getDispLimits( $current_{node}$, max, p)
    $dx_{max}$ ← norm( $current_{node}$.position - nq.position)
    $dt_{max}$ ← ||$current_{node}$.time - nq.time||
    $\Delta x$, $\Delta t$ ← getDxDt($dx_{max}$, $dt_{max}$,$\mathcal{V}$i)

    $\mathcal{N}$ = constGraph($\Delta x$, $\Delta t$, $current_{node}$);
    for j in 1:length($\mathcal{N}$)
      nj ← N(j);
      if isObstacle(nj, obstacle_map) || norm(getWindSpeed(
                                     nj, wind_map)) > $\mathcal{V}_{th}$
        continue
      end

      if isInGraph(nj, $current_{tree}$)
        cost_nj = calculateCost( $current_{node}$, nj)
        if cost_nj < (g( $current_{node}$,nj) + h(nj))
          $current_{tree}$ ← addEdge( $current_{tree}$, $current_{node}$,nj)
          $current_{tree}$ ← updateOpenSet( $current_{tree}$, $current_{node}$)
        else
          $current_{tree}$ ← addEdge( $current_{tree}$, $current_{node}$,nj)
          $current_{tree}$ ← addNode( $current_{tree}$, nj)
        end
      end
    end
  end
end
```

VI. Simulations and Results

In this section, results of simulations are discussed. The algorithm is tested on the environment as defined above [1]. Originally RRT* is tested on it by simulations by R. Takemaura et al.
The testing environment is set up it Matlab and tested for both thr original RRT* and the proposed PEA-RRT* for UAVs. Table 1 shows the comparison between the original and improved algortihms. The code used for the simulations produces the following results:

TABLE I
Simulation results for PEA-RRT*

Algorithm Metrics	Algorithm Names	
	RRT*	PEA-RRT*
Max eigen of FIM	298.0749	298.0749
Perception quality	1589.9971	1781.7417
Energy consumption	24.2415 [kJ]	24.1832 [kJ]
Overall Cost	133.8483	126.9698

From the analysis of this table, it can be seen that with the PEA-RRT* algorithm, the perception quality enhanced, energy consumption decreased because of better perception and optimized path. This led to an overall decrease in the cost of the flight. It is important to notice that the flight time remains almost the same.

Conclusion

In this paper, a perception-energy-aware path-planning algorithm is proposed for goal reaching tasks. This approach pushes the boundaries of the state of the art by not only keeping the planning limited to the planning algorithm but also considering energy awareness and perception quality as optimization parameters within the planning algorithm itself . It proposes better navigation through areas with harder perceptual conditions. We use a variant of RRT* that expands in bidirectional manner and uses optimization in terms of energy and perception. It is observed that this planning algorithm decreases the overall cost and can prove to be efficient in Unmanned vehicles and Unmanned Aerial Vehicles. Future directions can include testing the algorithm for other autonomous vehicles and incorporating the uncertainty of labels in semantic segmentation and feature classifiers when deployed in aerial vehicles.

References

[1] R. Takemura, N. Aoki, and G. Ishigami, "Energy-and-perception-aware planning and navigation framework for unmanned aerial vehicles," *Advances in Mechanical Engineering*, vol. 15, no. 4, 2023,

[2] L. Bartolomei, L. Teixeira, and M. Chli, "Perception-aware Path Planning for UAVs using Semantic Segmentation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 25-29, 2020, Las Vegas, NV, USA (Virtual), Affiliation: Vision For Robotics Lab, ETH Zürich, Switzerland.

[3] A. Thibbotuwawa, P. Nielsen, Z. Banaszak, et al., "Energy consumption in unmanned aerial vehicles: A review of energy consumption models and their relation to the UAV routing," in *Information Systems Architecture and Technology: Proceedings of 39th International Conference on Information Systems Architecture and Technology – ISAT 2018 - Part II.*

[4] R. Maidana, R. Granada, D. Jurak, M. Magnaguagno, F. Meneguzzi, and A. Amory, "Energy-Aware Path Planning for Autonomous Mobile Robot Navigation," School of Technology - Pontifical Catholic University of Rio Grande do Sul – Brazil,@pucrs.br, 2023

[5] Steven M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Technical Report, Computer Science Department, Iowa State University,* (TR 98-11), October 1998.

[6] J. Kuffner and S. LaValle, "RRT-Connect: An Efficient Approach to Single-Query Path Planning," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2, 2000, pp. 995-1001,

[7] Sik-Yum Lee and Xin-Yuan Song, "Evaluation of the Bayesian and Maximum Likelihood Approaches in Analyzing Structural Equation Models with Small Sample Sizes," *Multivariate Behavioral Research*, vol. 39, no. 4, pp. 653-686, 2004.

[8] Y. Tang, Y. Miao, A. Barnawi, et al., "A joint global and local path planning optimization for UAV task scheduling towards crowd air monitoring," *Computers & Networks*, vol. 193, pp. 1–15, 2021.