Regulated Pure Pursuit Algorithm for Robot Path Tracking

Steve Macenski
R&D Innovations
Samsung Research
s.macenski@samsung.com

Francisco Martín
Intelligent Robotics Lab
Rey Juan Carlos University
francisco.rico@urjc.es

***Abstract*—The accelerated deployment of service and industrial robot assets has spawned a number of robotic algorithm variations to better handle the real-world conditions on the ground. Several different local trajectory planning techniques have been deployed on practical robot systems successfully - including the Dynamic Window Approach (DWA), Pure Pursuit, and Model Predictive Control (MPC). While DWA and some formulations of MPC can progress along paths and deviate in the presence of dynamic obstacles, the use of pure path tracking algorithms is still very commonplace. Decades after its introduction, Pure Pursuit and its variants continues to be one of the most commonly utilized classes of local trajectory planners. However, few approaches have been proposed with schema for variable linear velocities - most Pure Pursuit work has assumed a constant translational velocity or fails to address the point at all. This paper presents a new variant of the Pure Pursuit algorithm designed with additional heuristics to regulate linear velocities, built atop the existing Adaptive variant. The *Regulated Pure Pursuit algorithm* makes incremental improvements on state of the art by adjusting linear velocities to significantly reduce tracking overshoot with particular focus on safety in constrained and partially observable spaces commonly negotiated by service robots.**

We present experiments with the Regulated Pure Pursuit algorithm on industrial-grade service robots in a campus setting. This work was built to be production-grade and is freely available at https://github.com/ros-planning/navigation2 for fast evaluation. It is included with the new and improved ROS 2 Navigation Stack (Nav2).

***Index Terms*—Service Robots; Mobile Robots; Motion and Path Planning**

I. Introduction

Many different path tracking and local trajectory planning algorithms have been proposed to aid in creating robust robot navigation systems. There is a wide diversity of approaches to this fundamental problem, although a few common techniques are commonly employed on modern mobile robots. Dynamic Window Approach (DWA), Pure Pursuit, and Model Predictive Control (MPC) are by far the most commonly deployed path trackers on service and industrial robots. They all have a strong heritage for reliably tracking paths in a wide range of environmental conditions.

DWA and MPC are often, but not always, formulated as multi-objective trajectory generation problems to maximize criteria such as avoiding dynamic obstacle collisions on top of path tracking. This has made them particularly well suited for many robotics applications where dynamic robot behaviors are rewarded. A great deal of academic work has been conducted on these methodologies allowing them to reach the point of maturity where they can be found on many commercially available robots today.

However, there still exist many applications of deployed robot systems where the multi-objective trajectory generation problem can be considered a downside. Notably, service and industrial robots have begun to be developed and deployed at an accelerated pace. The variety of problems and industries they support demand a variety of different behaviors - including pure path tracking. Surveyed among high-end research and service robot navigation systems, pure path tracking continues to be a common theme in many practical robot environments, which was surprising to the authors. Among single-objective path trackers, a simple and reliable method continues to be exploited decades after its initial development: Pure Pursuit.

Pure Pursuit uses simple geometry to find the curvature of a path required to drive a robot towards a given point on the path. The algorithm itself does not place any restrictions on the translational velocities during operation, however it also lacks any schema for adjusting them. Near-universally, implementations use a fixed speed. It also has a known issue of overshooting in the presence of sharp changes in path curvature at speed due to imperfect actuators and vehicle dynamics. Many variations of Pure Pursuit exist, however most address the more obvious area of the selection of lookahead points which largely aids in stabilizing convergence behaviors towards the path at a larger range of velocities. The Pure Pursuit algorithm was not developed with service and industrial robots in mind, which have additional safety requirements making it further unrealistic to move at a fixed velocity and literature lacks any formal description of a solution to regulate translational velocities.

This work proposes an incremental improvement on the Pure Pursuit path tracking algorithm by describing a method of adjusting translational velocities to improve safety and operability in a broad range of common service robot applications. We improve the Adaptive Pure Pursuit algorithm by additionally regulating linear velocities by applying heuristics around path curvature and proximity to obstacles - two of the most common events requiring conscientious navigation behaviors in dynamic environments. This *Regulated Pure Pursuit* algorithm addresses one of Pure Pursuit's known

issues; it slows a robot proportional to path curvature during a sharp turn, largely solving the overshoot issue in practical application. This has the additional impact of slowing on turns into partially observable dynamic environments (aisles, hallways, intersections) to perform safer service robot operations when also paired with our algorithm's collision detection methodology reducing the likelihood and impact of potential collisions. The Regulated Pure Pursuit algorithm also contains a regulation heuristic for linear velocity in close proximity to obstacles such as people and fixed infrastructure to reduce likelihood of collision in constrained indoor environments.

This work is freely available with a high quality implementation available for evaluation and integration in the Navigation2 (Nav2) mobile robot navigation system. This work is well documented, tested, and is in use on several robots deployed today.

II. Related Work

The Pure Pursuit algorithm has been used for path tracking for over 30 years on autonomous cars and mobile robots and is still a mainstay in many autonomous systems. It uses a geometric approach to drive a robot towards a local-goal point on a given path, dubbed the lookahead point at a given linear velocity. As the vehicle moves towards this lookahead point, a new point is selected on the path to drive towards. The continual update of the robot pose and lookahead "carrot" creates a solution to the canonical path tracking problem [1].

The essential steps are as follows, outlined in [2]:

1) Find the path pose closest to the robot's position
2) Select a path point to use as the lookahead carrot
3) Compute the curvature required to drive the robot to the lookahead carrot
4) Send commands to robot base controller
5) Repeat at desired update rate

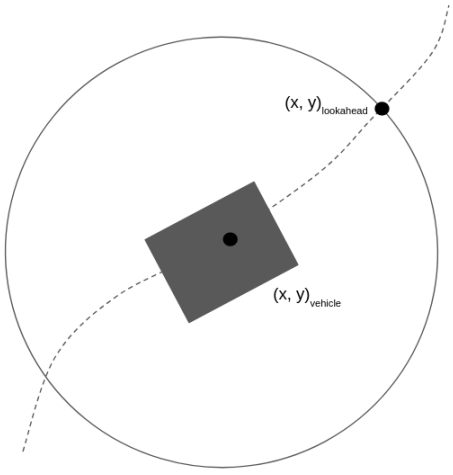

Fig. 1: Geometry of Finding the Lookahead Point.

Figure 1 and Figure 2 shows visual geometric representations of how Pure Pursuit functions. Figure 1 displays how the Pure Pursuit algorithm finds the lookahead point. First, it determines the closest point on the path to the vehicle. Using a given lookahead distance, it searches for the first point forward on the path that is at least that distance away. With a known lookahead point and a robot position, we can determine curvature of the circle (recall, $R = 1/\kappa$) using simple geometry. If the path can be represented in vehicle base coordinates, where the robot position is the origin, then the curvature can be represented as

$$\kappa = 2y/L^2 \tag{1}$$

Where κ is the path curvature required to drive the robot from its starting position to the lookahead carrot, y is the lateral coordinate of the lookahead point, and L is the actual distance between the robot and the lookahead point. Figure 2 shows this visually, where L is represented geometrically as the circle's chord.

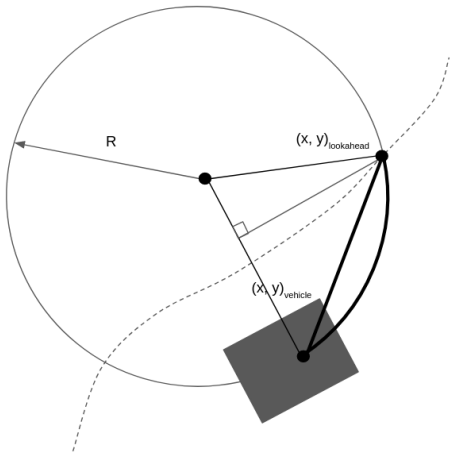

Fig. 2: Geometry of Finding the Path Curvature.

Since the Pure Pursuit algorithm is geometrically derived, the primary parameters of the Pure Pursuit path tracker are simply the translational velocity of travel and the distance along the path used to select the lookahead point. In the standard formulation, this lookahead point is a distance from the robot tuned to achieve an acceptable trade-off between oscillations centered around the path (shorter distances) and slower convergence (longer distances). There exists a broad range of admissible lookahead distances [1].

While geometrically simple, there are several known downsides of this approach. In high curvature situations, Pure Pursuit is known to have overshoot behaviors resulting in significant path deviations, even in a well tuned system [1]. This is typically not a major concern for autonomous driving applications which naturally has a minimum turning radius limit, but a more substantive issue for smaller-scale applications like industrial and consumer robots utilizing differential and omni-directional drive systems. It also does not specify any methods regarding translational velocities during execution or a relationship between the lookahead distance

and velocity. While this can be beneficial as it allows for a great deal of flexibility with different linear velocity profiles, in practice, without described methods, nearly all known variants use a constant translational velocity profile. Finally, it fails to specify any collision detection - which is of special importance due to the known overshoot behavior in when following high curvature paths.

Over time, many variants of this algorithm have been proposed to increase path tracking stability by varying computations of the lookahead point. MIT's entry into the DARP Urban Challenge implemented the Pure Pursuit algorithm for lane following with the most common variation: varying the lookahead distance changes proportionally to the translational velocity [3]. This variation allows a vehicle to travel at a broader range of velocities and maintain stability of the pure pursuit algorithm. For velocities in the operating range, a mapping of lookahead distances is required such that they have a lookahead distance with an acceptable trade-off between oscillation and slower convergence to the path. A common formulation for this is $L = v_c l_t$, where L is the lookahead distance, v_c is the translational velocity, and l_t is a lookahead gain representing the time to project v_c forward by [4]. This is also referred to as the "Adaptive" Pure Pursuit algorithm [5].

Another variation that is particularly beneficial in urban environments is interpolating between path via-points to determine the closest robot pose to the path and the lookahead point. Small variations are present in the actual lookahead distance utilized to compute angular velocities without interpolation between via-points. These small variations can create discontinuous jumps in the angular velocity output while traveling on a constant curvature path [4]. By interpolating between waypoints on a path, the pure pursuit algorithm can use exact values of the desired lookahead distance and smooth out the rate of change between iterations of the algorithm.

Beyond path tracking, Pure Pursuit has also been adapted for other tasks such as wall and person following [6]. Using the same ideas, it has been shown that the geometry of the algorithm can be adapted for a variety of non-path following applications.

However, none of the variants found address translational velocity behaviors or specifies preemptive collision detection to avoid catastrophic failure. Several variations on Adaptive Pure Pursuit have been proposed aimed at stabilizing path tracking, but none have targeted a solution to fix the overshoot found on particularly high curvature paths. This problem disproportionately impacts the most common robot platform types that can perform sharp turning maneuvers: differential and omni-directional drives.

This paper proposes a new variant on Pure Pursuit, Regulated Pure Pursuit, targeting these problems. It builds on the existing variants with heuristics for regulating translational velocities focusing on modern service, consumer, and industrial robotics needs.

III. Regulated Pure Pursuit Algorithm

IV. Implementation

V. Experiments and Analysis

VI. Limitations

VII. Conclusion

Acknowledgements

Other contributors to this work include Shrijit Singh and Ramon Wijnands.

References

[1] Coulter, R., "Implementation of the Pure Pursuit Path Tracking Algorithm". Carnegie Mellon University, Pittsburgh, Pennsylvania, Jan 1990.

[2] Samuel, M., Hussein, M., Mohamad, M., "A Review of some Pure-Pursuit based Path Tracking Techniques for Control of Autonomous Vehicle". International Journal of Computer Applications, 2006.

[3] Campbell, S., "Steering control of an autonomous ground vehicle with application to the DARPA Urban Challenge". University of Notre Dame, 2005.

[4] H. Ohta, N. Akai, E. Takeuchi, S. Kato and M. Edahiro, "Pure Pursuit Revisited: Field Testing of Autonomous Vehicles in Urban Areas", Cyber-Physical Systems Networks and Applications (CPSNA) 2016 IEEE 4th International Conference on, pp. 7-12, 2016.

[5] H. Ohta, N. Akai, E. Takeuchi, S. Kato and M. Edahiro, "Adaptive Pure Pursuit Model for Autonomous Vehicle Path Tracking", International Journal of Science, Vol. 4, No. 3, 2017.

[6] Morales, J., Martínez, J.L., Martínez, M.A. et al. "Pure-Pursuit Reactive Path Tracking for Nonholonomic Mobile Robots with a 2D Laser Scanner". EURASIP J. Adv. Signal Process. 2009, 935237 (2009).