Proceedings of ICAR 2003
The 11th International Conference on Advanced Robotics
Coimbra, Portugal, June 30 - July 3, 2003

Moving Safely but not Slowly - Reactively Adapting Paths for Better Trajectory Times

K. MadhavaKrishna R. Alami T. Simeon

LAAS-CNRS
7, avenue du Colonel-Roche
31077 Toulouse Cedex - France

Abstract: *In our previous effort [1] we had reported on a methodology for computing the maximum velocity profile for a planned trajectory of a mobile robot. The profile computed is indicative of the maximum velocity the robot can possess at a given location that enables it to come to a halt before collision with mobile objects that could appear from areas not seen by the robot's sensing apparatus. The profile also allowed the deformation of paths such that the overall trajectory time is reduced. An extension of the approach to an online scheme that modifies and adapts the path as well as velocities to changes in the environment such that both safety and execution time are not compromised is presented in this paper. Simulation and experimental results indicate the efficacy of this methodology.*

1 Introduction

In the emerging context of human friendly robots, it is of paramount importance that the robot navigation is performed in a way that avoids unnecessary interference with humans especially so if they are aged. Under this scenario it would be advisable to have robots that slow down while they approach turns and bends along their path, even if they (robots) do not intend to make a turn, by being pro-active in expectation of mobile objects (humans) to emerge from those turns and bends. This is explained through figure 1, where a safe robot would need to slow down, even if it does not intend to make a turn through the doorway in anticipation of objects to emerge through the doorway.

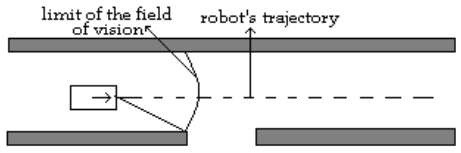

Figure 1: *It entails that a safe robot slow down while approaching the doorway*

Our previous effort[1] essentially confronted the following two problems.

1: On a path computed by a typical planner[6] further plan a velocity profile over the path that allows the robot to come to a halt before colliding with unexpected moving objects that could impinge on to its future trajectory from areas which are hidden to its sensory apparatus. The velocity profile is computed with the knowledge of robot dynamics (maximum possible acceleration, deceleration and velocity), sensor capacities (maximum range of laser) and environment dynamics (maximum speed a mobile could possess). Safe navigation is a natural consequent of such a scheme that would entail the robot to slow down in situations shown in figure 1 if required.

2: It need not always be the case the robot needs to compromise its speed in lieu of safety, hence further adapt the path to obtain a profile that reduces overall trajectory time if possible.

The present paper is directed towards the problem of the reactive stage where an initial trajectory planned for a particular environment is modified when new objects are introduced such that the basic philosophy of ensuring safety as well as reducing time-lengths of the path continue to be respected. Simulation and experimental results are presented to indicate the efficacy of the scheme.

Related work can be cited in the areas of modifying global plans using sensory data obtained during execution for overcoming uncertainty accumulated during motions [3] and those that try to bridge the gap between planning and uncertainty [7] or planning and control [5], [2]. The work of [9] involves considering robot's pose and velocities at the planning phase. A path is determined in the (x, y) space and a subgoal is chosen. A sequence of linear and angular velocities, (v, w), is furnished till the subgoal is reached. The resultant path is able to tackle better the problem of

turning into a narrow corridor or a room as the velocities are computed not for the next instant alone but till the inext sub-goal. It however does not mention reducing the time-length of path by modifying it and the dynamics of the environment does not seem to affect the computation of the velocity profile which are addressed in this effort. In [8] a policy search appoach is presented that projects a low dimensional intermediate plan to a higher dimensional space where the orientation and velocity are included. As a result better motion plans are generated that enable better execution of the plan by the robot. The current effort has similar ties to [8] at the planning level but also extends it to a suitable reactive level in the presence of new obstacles encountered at execution.

2 Computation of maximum velocity profile $v_\tau(s)$

While the methodology for computing the maximum velocity profile delineated here is essentially for a holonomous path, its extension to the non-holonomous case is not complex.

1. A holonomic path τ, consisting of a sequence of straight line segments ab, bc, cd is deformed into a sequence of straight lines and clothoids to ensure continuity of velocities at the bends [4].
2. The linear velocity along a clothoid is a constant and the maximum possible linear velocity considering robot dynamics alone is calculated for each of the clothoidal arc $a1b0, b1c0$ according to [4] and are represented as $v_c(a1), v_c(b1)$.
3. The straight segment $aa1$ is discretized into N equally spaced points, excluding the endpoints of the segment, viz a and $a1$. We denote the first such point as a_1 and the last of such points as a_N. The point of entry into the clothoid, viz. $a1$ is also denoted as a_{N+1}.
4. For each of the N points, a_i, the steps $4a$ to $4e$ are repeated.

4a Maximum possible velocity that a robot could have such that it can come to a halt before colliding with objects that enter into the robot's field of vision from the boundary is computed as [1]

$$v_{rb}(a_i) = -v_{ob} + \sqrt{v_{ob}^2 + 2d_m R_{vis}} \qquad (1)$$

Where $v_{rb}(a_i)$ is the velocity of robot due to objects on the boundary when the robot is at a_i, d_m is the maximum deceleration of the robot, v_{ob} is the maximum velocity possessed by a mobile object and R_{vis} represents the maximum range of the sensor.

4b Velocity of the robot due to stationary obstacles[1] inside the robot's field of vision that create shadows is computed as a solution to

$$v_{rsv}(a_i)^4 - 4(d_m d\cos\theta + v_{ob}^2)v_{rsv}(a_i)^2 + 4d_m^2 d^2 \geq 0 \qquad (2)$$

Here d represents the distance to the vertex of a stationary object that creates a shadow, θ is the angle made by theat vertex with the robot's motin direction and $v_{rsv}(a_i)$ is the velocity of the robot due to possible mobile objects that can emerge from behind those shadowed vertices. The minimum of all the velocities due to such vertices is found and denoted as $v_{rs}(a_i)$.

4c The maximum possible velocity of the robot at a_i due to environment is then computed as

$$v_{re}(a_i) = min(v_{rb}(a_i), v_{rs}(a_i)) \qquad (3)$$

4d Velocity of the robot at a_i due to its own dynamics is given by

$$v_{rd}(a_i) = \sqrt{v_r^2(a_{i-1}) + 2a_m s(a_i, a_{i-1})} \qquad (4)$$

The above equation is computed if $v_{re}(a_i) > v_r(a_{i-1})$. Here $s(a_i, a_{i-1})$ represents the distance between the points a_i and a_{i-1} and a_m represents the maximum acceleration of the robot.

4e The eventual velocity at a_i is given by

$$v_r(a_i) = min(v_{rd}(a_i), v_{re}(a_i), v_{rm}) \qquad (5)$$

Here v_{rm} represents the maximum robot velocity permissible due to servo motor constants.

5. The velocity at the endpoint $a1$ is computed as $v_r(a1) = min(v_r(a1), v_c(a1))$ and this would be the linear velocity with which the robot would traverse the clothoid.
6. Steps $6a$ and $6b$ are performed by going backwards on each of the N points from a_N to a_1

6a $if v_r(a_i) > v_r(a_{i+1})$ then modified maximum possible velocity at a_i is computed as

$$v_{rd}(a_i) = \sqrt{v_r^2(a_{i+1}) + 2d_m s(a_i, a_{i+1})} \qquad (6)$$

6b Finally the maximum safe velocity at a_i is given as $v_r(a_i) = min(v_r(a_i), v_{rd}(a_i))$

7. Repeat steps 3 to 6 for all the remaining straight segments to obtain the maximal velocity profile over a given trajectory τ as $v_\tau(s) = \{v_r(a), v_r(a_1), ..., v_r(a_N), v_r(a1), v_r(b_1), ..., v_r(d)\}$

3 From Plan to Execution

The velocity profile, $v_\tau(s)$, is a sequence of maximum velocities calculated at discretized locations

along the trajectory $\tau(s)$. The computation of the velocity profile at planning and execution stages are not at the same locations in general. During execution it is computationally expensive to compute the profile for the entire remaining trajectory, hence the profile is computed for the next finite distance, given by, $d_{safe} = d_{max} + nd_{samp}$, where $d_{max} = v_{rm}^2/(2 * d_m)$, represents the distance required by the robot to come to a halt while it moves with the maximum permissible velocity afforded by motor constants. And $d_{samp} = v_{rm}t_{samp}$ is the maximum possible distance that the robot can move between two successive samples (time instants) of transmitting motion commands, where time between two samples is t_{samp}. The main issue here is what should be the distance over which the velocity profile needs to be computed during execution such that it is safe. A velocity command is **not** considered safe if it is lesser than the current velocity and **not** attainable within the next sample. The velocity is constrained by environment as well as robot's own dynamics and hence the role of either in creating such an unattainable velocity is probed below.

Effect of Environment Mobile objects that can emerge from corners in a head-on direction cause the greatest change in velocity over two samples. Figure 2 shows one such situation, where the rectangular object casts a shadow and is susceptible to hide mobiles. Let the current velocity of the robot at a due to the object be v_1. Let the velocity at a distance, s, from a, at b (fig 2) due to the object be v_2.

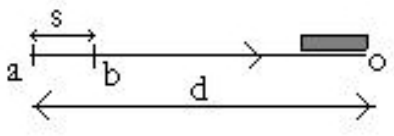

Figure 2: *The effect of the rectangular object that could hide possible mobiles on robot's velocity at locations a and b*

The velocities at a and b are given by

$$v_1(a) = -v_{ob} + \sqrt{v_{ob}^2 + 2d_m d} \tag{7}$$

$$v_2(b) = -v_{ob} + \sqrt{v_{ob}^2 + 2d_m(d-s)} \tag{8}$$

Hence

$$v_1^2 - v_2^2 = 2d_m s + 2v_{ob}(\sqrt{v_{ob}^2 + 2d_m(d-s)} - \sqrt{v_{ob}^2 + 2d_m d}) \tag{9}$$

Evidently the second term on the right hand side of equation 9 is negative, since the second square root term is less positive than the first. Hence $v_1^2 - v_2^2 \leq 2d_m s$. Therefore the velocity at b, v_2 can be attained from the velocity at a, v_1 under maximum deceleration, d_m, irrespective of the maximum velocity of the mobile or the robot's own motor constraints. This was intutively expected since the robot's velocity at any location is the maximum possible velocity that guarantees immobility before collision, its velocity at a subsequent location permitted by the environment would be greater than or equal to the velocity at the same location obtained under maximum deceleration from the previous location. In other words for safeness of velocity going purely by environmental considerations it would suffice to calculate the velocity, for the next sampling distance alone, for without loss of generality, $d = d_{samp}$.

Effect of robot's dynamics The robot needs to respect the velocity constraints imposed while nearing the clothoidal arcs and eventually while coming to the target. The robot can reach zero velocity from its maximum velocity over a distance of d_{max}, computed before. Hence $d_{max} + d_{samp}$ represents the safe distance over which the velocities need to be computed.

3.1 Online path adaptation for better trajectory time

During navigation the robot in general comes across objects hitherto not a part of the map. The robot reacts to these new objects in line with the basic philosophy of safe as well as time reduced paths. The adaptation proceeds by finding locations over a finite portion of the future trajectory where drops in velocity occur and pushing the trajectory away from those vertices of the objects that caused these drops to areas in free space where higher velocities are possible. A search is made through the newly found locations of higher velocities for a time reduced path.

Generalized Procedure The generalized procedure for adapting the path in presence of new objects is delineated through figure 3

1. On the trajectory segment that is currently traversed, AB in figure 3, enumerate the vertices of objects that reduce the velocity of the robot.
2. The positions are found on AB where the influence of vertices is likely to be maximal.
3. These positions are pushed by distances $d_p = k(v_l - v_r)$, where v_l and v_r are the velocities at

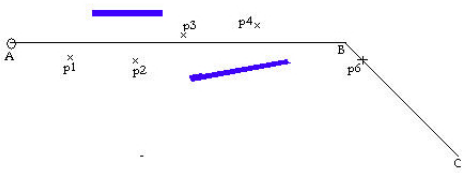

Figure 3: *A trajectory in presence of new objects. The points marked with crosses represent locations through which a path is searched for better time-length*

that location on the path due to the most influential vertices on the left and right of the path. These new locations are denoted as $p1, p2, p3, p4$ (fig 3) and maintained as a list provided the velocity at the new locations is more when compared with the original ones. $p6$ is the farthest point on the robot's trajectory visible from its current location at A

4. On this set of locations $A, p1, p2, p3, p4, p5, p6$ starting from the current location at A, find a trajectory sequence shorter in time than the current sequence of $A, B, p6$ if it exists.
5. The steps 1 to 4 are repeated until the robot reaches the target.

It should be noted that if a collision with an object is detected, a collision free location is first found that connects the current location with another location on the original trajectory and this new collision free path is further adapted for a time-reduced path if it exists.

4 Simulation and Experimental Results

4.1 Velocity profiles during plan and execution

In this section the velocity profiles obtained during planning and execution stage are compared in the absence of any new objects during execution. Figures 5 and 6 show the execution by the Nomad $XR4000$ (fig 4) of paths computed by a standard planner. Figure 5 corresponds to the original path computed by the planner and figure 6 is its time reduced counterpart.

The velocity profiles during execution of the two paths are shown in figure 7. Some of the bigger drops in the unreduced profile are absent in the reduced profile as the robot avoids turning close to the obstacles that form the bends. The path of figure 6 got executed in $12.9s$ while the path in figure 5 was executed in $13.98s$. The figures are meant as illustrations of

Figure 4: *The Nomad XR4000 used in our experiments*

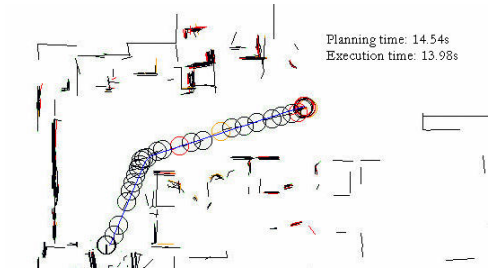

Figure 5: *Execution of the original planned path by the Nomad*

the theme that trajectories deformed to shorter time-lengths at planning stage are also executed in shorter time during implementation than their unreduced versions.

4.2 Online adaptation of paths for better trajectory time

This section presents results of the algorithm in the presence of newly added objects that affect the velocities of the robot in real-time. Figure 8 shows a path

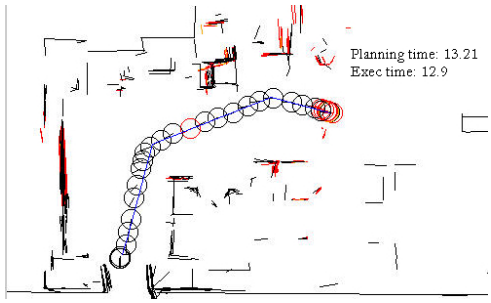

Figure 6: *Execution of the time-reduced path by the Nomad*

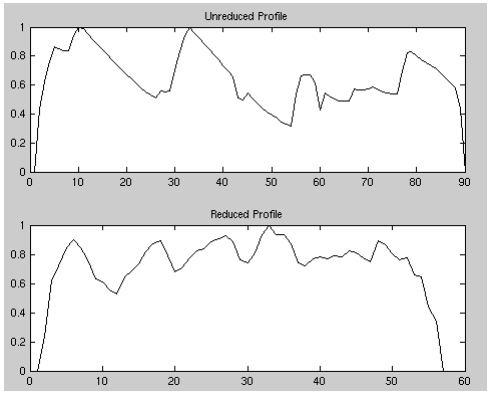

Figure 7: *The top profile corresponds to the path executed in figure 5 and the bottom to figure 6*

where the robot collision avoids the two new segments $S1$ and $S2$ intersecting the original planned trajectory but does not adapt its path for better time. The velocity profile for the same is shown in 9. Figure 10 is the counterpart of figure 11 where the robot adapts its path to a better time-length reactively. The big dips in the velocity profile of figure 9 are filtered in figure 11 considerably as the robot avoids the obstacles with larger separation. The time reduced execution tallied to $10.9s$ while the unreduced version was executed in $12.5s$. The trajectory time at planning was $7.9s$. The above graphs are those obtained in simulation.

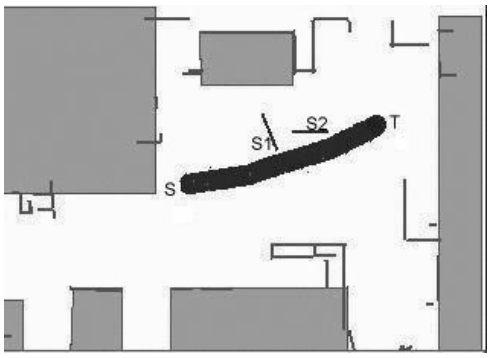

Figure 8: *A simulated execution in the presence of two new segments $S1$ and $S2$ along with the corresponding velocity profile. The path is not adapted to better time-length. Start and goal locations marked as S and T.*

Figure 12 shows the executed path by the $XR4000$ Nomadic robot after adaptation to better time-length. The obstacles in the original map are shown by black lines, while the segments perceived by the sick laser are shown in lighter shades of gray. Some of these segments get mapped to the ones in the map and the

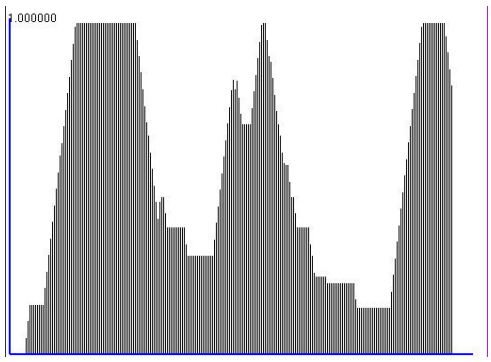

Figure 9: *The velocity profile for the execution of figure 8*

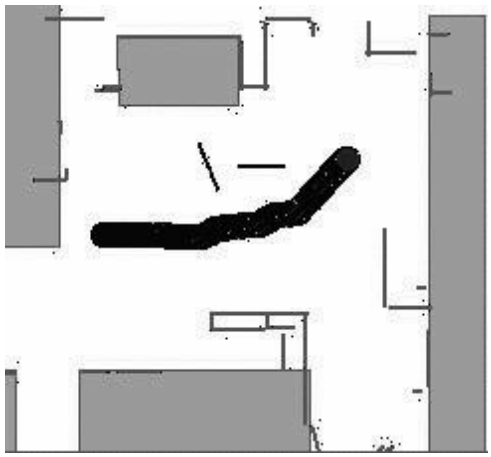

Figure 10: *Path of figure 8 adapted to better time-length*

others are considered new segments. The segments of concern are those that form a box shaped object marked as B. The vertex d exerts the maximum in-

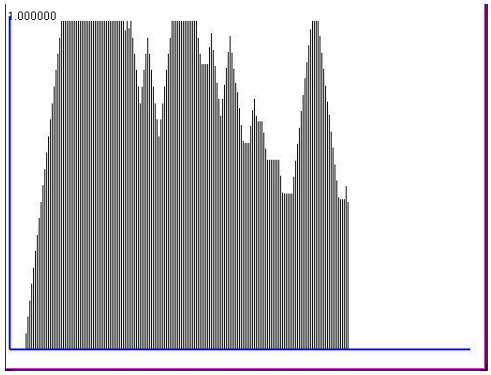

Figure 11: *The velocity profile for the execution of figure 10*

fluence forcing the robot to adapt its trajectory to a lesser time. The unadapted trajectory is shown in the same figure as a straight line connecting the initial and final locations, S and T.

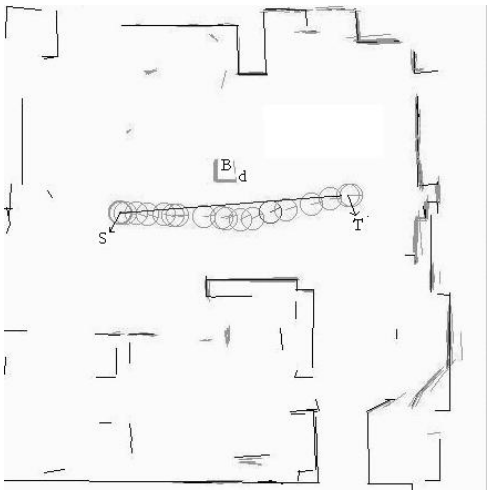

Figure 12: *Time reduced path executed by the Nomad XR4000. Increasing linear and angular separation from vertex d facilitates a higher speed.*

The velocity profile for the same is shown in figure 13. The reduced time-length tallied to $9.6s$ while the execution time in the absence of path adaptation was $10.5s$ with the original planning time being $8.8s$.

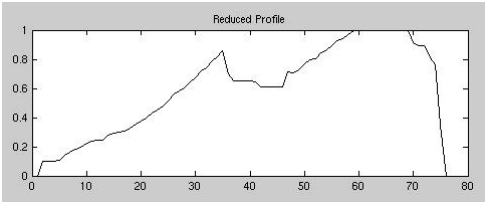

Figure 13: *Velocity profile for the path executed by the Nomad in figure 12*

5 Conclusions and Scope

A real-time methodology for computing velocity profile and modifying paths for better time-lengths based on the computed velocity profile has been presented. Simulation and experimental results at real-time corroborate with our earlier results obtained at the planning stage (that by keeping away from vertices of objects that could hide mobiles the robot could move at higher velocities and obtain better time-lengths) and thus the efficacy of overall strategy is vindicated. The minimum distance over which the velocities need to be computed on the remaining trajectory during real-time such that the computed velocities are safe is theoretically established. This avoids repetetive computation of velocities over the entire remaining trajectory for every motion command, thereby reducing computational intensity and facilitating for real-time implementation. The methodology could be useful in the context of personal robots moving in areas where interference with mobile humans especially aged ones are generally expected. Immediate scope of the work lies in validating the methodology in presence of mobiles such that the robot continues to avoid the mobiles without halting and continuing to respect safety considerations.

References

[1] R. Alami, T. Siméon, and K.M. Krishna. On the influence of sensor capacities and environment dynamics onto collision-free motion plans. *IEEE/RSJ International Conference on Intelligent Robots and Systems, EPFL, Swizerland*, 2002.

[2] J.C. Alvarez, A. Skhel, and V. Lumelsky. Accounting for mobile robot dynamics in sensor-based motion planning: experimental results. *IEEE International Conference on Robotics and Automation, Leuven (Belgium)*, 1998.

[3] B. Bouily, T. Simeon, and R. Alami. A numerical technique for planning motion strategies of a mobile robot in presence of uncertainty. *IEEE International Conference on Robotics and Automation, Nagoya (Japan)*, 1995.

[4] S. Fleury, P. Soueres, and J.P. Laumond. Primitives for smoothing mobile robot trajectories. *IEEE Transactions on Robotics and Automation*, 11(3):441–448, 1995.

[5] M. Khatib, B. Bouily, T. Simeon, and R. Chatila. Indoor navigation with uncertainty using sensor-based motions. *IEEE International Conference on Robotics and Automation, Albuquerque (USA)*, 1997.

[6] J.C. Latombe. *Robot Motion Planning.* Kluwer Academic, 1991.

[7] A. Lazanas and J.C. Latombe. Motion planning with uncertainty: a landmark approach. *Artificial Intelligence*, pages 287–315, 1995.

[8] N. Roy and S. Thrun. Motion planning through policy search. *IEEE/RSJ International Conference on Intelligent Robots and Systems, EPFL, Swizerland*, pages 2419–2425, 2002.

[9] C. Strachniss and W. Burgard. An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments. *IEEE/RSJ International Conference on Intelligent Robots and Systems, EPFL, Swizerland*, 2002.