An Integrated Approach to Real-Time Mobile Robot Control in Partially Known Indoor Environments

Article · January 2005

DOI: 10.1109/IECON.2005.1569176 · Source: OAI

3 authors:

Marija Seder
University of Zagreb
32 PUBLICATIONS **388** CITATIONS
SEE PROFILE

Kristijan Macek
Varian Medical Systems
32 PUBLICATIONS **408** CITATIONS
SEE PROFILE

Ivan Petrovic
University of Zagreb
215 PUBLICATIONS **1,964** CITATIONS
SEE PROFILE

An integrated approach to real-time mobile robot control in partially known indoor environments

Marija Seder, Kristijan Maček, Ivan Petrović

University of Zagreb, Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
E-mail: {marija.seder, kristijan.macek, ivan.petrovic}@fer.hr

Abstract—**In this paper we present a motion control method for mobile robots in partially known indoor environments based on integration of focussed D* search algorithm and dynamic window local obstacle avoidance algorithm. While focussed D* generates global path, dynamic window local obstacle avoider generates the admissible robot trajectories that ensure safe robot motion. A simple and efficient procedure to the selection of appropriate motion commands based upon alignment of acquired trajectories and global geometric path is proposed. The initial a priori knowledge is used about the environment in the form of the occupancy grid map that is incrementally updated in runtime. The algorithms are verified both by simulation and experiments with a Pioneer 2DX mobile robot equiped with laser range finder.**

I. INTRODUCTION

Indoor mobile robots should perform goal-directed tasks in cramped and unknown environments. Both global path planning and local reactive obstacle avoidance algorithms must be implemented in order to make a mobile robot with this capability. While a global path planning algorithm calculates optimal path to a specified goal, a reactive local obstacle avoidance module takes into account the unknown and changing characteristics of the environment based on the local sensory information.

Some of the most popular reactive collision avoidance techniques can be divided into directional and velocity space based approaches. The directional approaches generate a direction of the robot to head in. Potential field method [1], Vector Field Histogram [2] and Nearness Diagram algorithm [3] are typical representatives of this category. Whilst these approaches are efficient in obtaining an efficient direction command they do not take robot dynamics directly into account. Velocity space approaches take robot's kinematic and dynamic constraints directly into account by performing a search in space of translational and rotational velocities. It is typically assumed that the robot moves along circular arcs. Curvature-Velocity method [4], Lane-Curvature method [5], and Dynamic Window (DW) approach [6] are among typical methods where an objective function is evaluated according to local obstacle configuration and goal directedness.

Global path planning is well studied by Latombe in [7] where techniques such as cell-decomposition and road map are examined. A simple numerical navigation function NF1 also mentioned by Latombe was successfully implemented in the Global Dynamic Window approach [8]. To enhance the smoothness of globally obtained paths, elastic bands were introduced in [9]. The A* graph global search algorithm [10] gives a complete and optimal path in static environments. It was improved in [11] and additionally in [12] for more efficient on-line searching of dynamic environments. These dynamic A* algorithms are named as D* and focussed D* (FD*) algorithms, respectively.

The integration method proposed in [13] for the integration of A* and DW algorithm is here adapted for the use of FD* algorithm instead of A* algorithm. Thus, reliably real-time control of mobile robot in dynamic environments even with narrow passages is ensured.

II. PATH PLANNING ALGORITHM

Graph based search algorithms are the most commonly used algorithms for path planning of mobile robots. Among them the most popular one is A*, which finds complete and optimal path in static environments [10]. In a dynamic environment the global path must be completely replanned at each servo tick, which may cause A* algorithm to perform poorly since it does not use search information from previous iterations. Therefore, minimum path criterion may not be optimal in the sense of minimum time. Therefore, D* graph search algorithm should be used [11], which allows updating of only those nodes along the path that actually change due to sensor measurements. Its computation effort is further improved in [12], where introduction of a heuristic function for the remaining cost of the path was proposed, giving the so called focused D* algorithm.

The FD* graph search algorithm is based on a path cost function g, which represents the total cost from the current node of the search to the goal node, and a heuristic function h, which estimates but never overestimates the cheapest solution for achieving the current node from the start node in the (x, y) grid map search space. Such a heuristic function is called admissible and optimistic. The total cost function $f = g + h$ determines order of expanding nodes in state space. When following any path from the start node, value of the f-funciton never decreases, which is true if heuristic exhibits monotonicity.

The most often used heuristic function is Euclidian distance from the start node to the current node. However, the Euclidian distance is computationally inefficient since calculation of the square root function for each node expansion demands floating point arithmetics. In order to alleviate this problem a heuristic that uses integer arithmetics is proposed. This is possible in occupancy grid maps because they enable transition costs to be described as integer multiplies. For example, if each cell of a grid is regarded as a node of the path graph, 10 can be used for straight transition and 14 for diagonal transition. We used the heuristic:

$$\begin{aligned}
a &= \max(\,|x_S - x_N|, |y_S - y_N\,|\,), \\
b &= \min(\,|x_S - x_N|, |y_S - y_N\,|\,), \\
h(N) &= 14b + 10(a - b).
\end{aligned} \tag{1}$$

where (x_N, y_N) are the coordinates of the current node N and (x_S, y_S) of the start node S. This heuristic exhibits monotonicity because it fulfilles the triangular inequality property.

FD* search fans out from the goal node, expanding neighbour nodes within the contours of increasing f-value untill the start node is reached or all possible free neighbours (free of obstacle) are exhaused upon which the algorithm declares no path found. Initial search by FD* algorithm sets pointer from each state in the searched area to the next state and optimal paths to the goal from every state in the expanded area of the environment are computed simply following the pointers. Further on-line execution of the algorithm relies on sensory information from the vicinity of the robot environment. Any discrepancy that is discovered from the earlier information about the environment initiates algorithm execution. New path is then determined redirecting the pointers localy. The number of expanded nodes is minimal and consequently the time of execution.

III. Dynamic window obstacle avoidance

The dynamic window approach is a velocity space based local reactive avoidance technique where search for commands controlling the robot is carried out directly in the space of velocities. Trajectory of a robot can be described by a sequence of circular and straight line arcs as given in the original paper [6].

The search space is reduced by the kinematic and dynamic constraints of the robot to a certain span of velocities around the current velocity vector (v_c, ω_c) that can be reached within the next time interval. The dynamic window V_d containing the possible reachable velocities is defined as [6]:

$$V_d = \left\{ (v, \omega) \mid \begin{array}{l} v \in [v_c - \dot{v}_b \Delta t, v_c + \dot{v}_a \Delta t] \wedge \\ \omega \in [\omega_c - \dot{\omega}_b \Delta t, \omega_c + \dot{\omega}_a \Delta t] \end{array} \right\}, \tag{2}$$

where accelerations $\dot{v}_a$ and $\dot{\omega}_a$ are maximal translational and rotational accelerations exertable by the motors and $\dot{v}_b$ and $\dot{\omega}_b$ are maximal translational and rotational breakage deccelerations.

A velocity touples (v, ω) is considered safe if the robot is able to stop along the trajectory defined by (v, ω) before hitting any object that may be encountered along that path. The set V_a of admissible velocities can be determined according to:

$$V_a = \left\{ v, \omega \leq \sqrt{2\rho_{min}(v, \omega)\dot{v}_b} \wedge \sqrt{2\rho_{min}(v, \omega)\dot{\omega}_b} \right\}, \tag{3}$$

where the term $\rho_{min}(v, \omega)$ represents the distance to the closest obstacle on the corresponding curvature.

If we denote the initial search space of velocities that are limited by the maximum translational and rotational velocity value as V_s the resulting search space can be described as the intersection of the restricted areas:

$$V_r = V_s \cap V_a \cap V_d. \tag{4}$$

Search space V_r can be expressed as the Cartesian product of two search spaces:

$$V_r = V_{rv} \times V_{r\omega}, \tag{5}$$

where V_{rv} is translational velocity search space and $V_{r\omega}$ rotational velocity search space. We have chosen dimension of V_{rv} to be 5 and dimension of $V_{r\omega}$ to be 7, therefore, number of velocity touples is 35. In this work, dimensions are chosen such that computational cost is acceptable. Each velocity touple (v, ω) uniquely determines a circular trajectory whose radius is calculated as $r = \dfrac{v}{\omega}$ and a sequence of such velocity vectors forms a curvature that is approximated by a sequence of circular arcs. Higher dimension of $V_{r\omega}$ than of V_{rv} is chosen with the purporse of densier spatial coverage with the possible trajectories (see Fig. 1). Translation velocities influent on the lengths of trajectories in given time and since in navigation higher velocities are prefered, the dimension of V_{rv} is less signifficant. The dynamic window is centred around the current velocity (v_c, ω_c) and the extensions of it depend on the accelerations that can be exerted. All curvatures outside the dynamic window cannot be reached within the next time interval and thus are not considered for the obstacle avoidance.

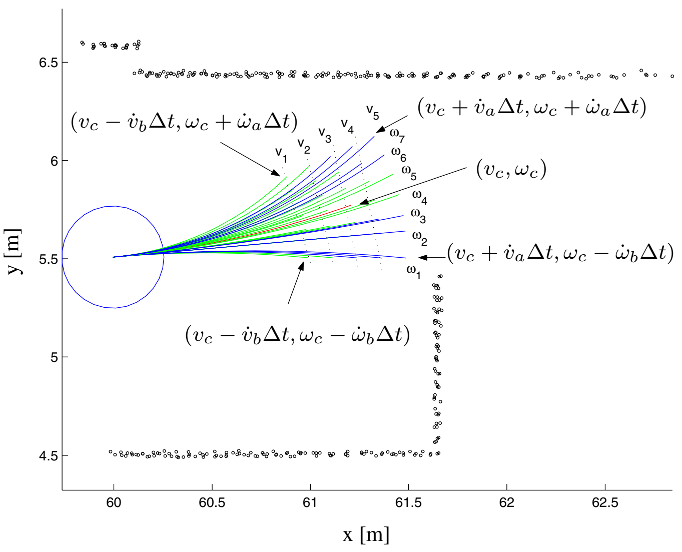

Fig. 1. A snapshot of possible robot trajectories

In order to generate a trajectory to a given goal point for the next n time intervals it has to be determined which velocities (v_i, ω_i) for each of the n intervals must be executed. Therefore, the search space for these vectors is exponential in the number of the considered intervals. To make the search for velocities feasible and appropriate for fast reactive response, the dynamic window approach considers exclusively the first time interval to choose the optimal velocity vector and assumes that the velocities in the remaining $n - 1$ time intervals are constant. The reduced search space is two-dimensional over the discrete set of velocity touple (v, ω) and thus feasible in polynomial time search sense. The search is repeated after each time interval and the velocities stay automatically constant if no new commands are given. A snapshot of possible robot trajectories at given time and local obstacle configuration determined by resulting velocity space V_r assuming for a next n time intervals to be constant is depicted in Fig. 1.

The local obstacle configuration can be taken into account by calculating the distance until collision along a certain circular trajectory. Alternatively, the time until collision may be used, which takes velocity v of the robot more directly into account [14]. The clearance objective measure ϑ_{clear} describes how close is a chosen trajectory to potential obstacles and is considered only if the basic admissability condition for the paricular trajectory is fulfilled (i.e. the breakage distance is smaller than the closest object on the trajectory):

$$\vartheta_{clear}(v,\omega) = \begin{cases} 0 & : \ t_{col} \leq T_b \\ \frac{t_{col}-T_b}{T_{max}-T_b} & : \ T_b < t_{col} < T_{max}, \\ 1 & : \ t_{col} > T_{max} \end{cases} \quad (6)$$

where $T_b(v,\omega) = max(\frac{v}{\dot{v}_b}, \frac{\omega}{\dot{\omega}_b})$ is the breakage time along a certain trajectory determined by (v,ω) and $t_{col}(v,\omega) = \frac{\rho_{min}(v,\omega)}{v}$, is the time until collision with the closest obstacle on the trajectory. It has to be noted that the collision calculation is not based on the contact between an object and the robot contour itself but rather between an object and an enlarged safety contour margin around the robot. In [6] it is called speed dependent side clearance SC_v and is used for the translational velocity. This side clearance grows linearly with v and the effect of it is that at higher speeds the free-space areas (i.e. corridors, passages between objects) appear narrower to the robot. $T_{max} = \frac{s_l}{v_{max}}$ is the admissible collision time and it represents the temporal limit above which a trajectory is considered void of obstacles. Its value depends on the maximum translational velocity of the robot v_{max} and the look-ahead distance of the sensors s_l used.

The velocity maximizing a certain objective function $\Gamma(v,\omega)$ is chosen from the remaining set of velocities V_r. In [6] it is expressed as weighted sum of objective measures for clearance ϑ_{clear}, target heading ϑ_{head} and linear velocity ϑ_{vel}:

$$\Gamma(v,\omega) = \lambda_{clear}\vartheta_{clear} + \lambda_{head}\vartheta_{head} + \lambda_{vel}\vartheta_{vel}. \quad (7)$$

Linear velocity measure ϑ_{vel} is chosen such that higher linear velocities are preferred, and target heading measure ϑ_{head} considers achieving heading towards global goal position. As was reported in [4] and [6], this approach is susceptible to local minima without further global path information. If connectivity of free-space toward goal position is considered, local minima can be avoided. A successful approach is reported in [15] among others. In the Global Dynamic Window approach described in [8] the objective function has four terms:

$$\Gamma(v,\omega) = \lambda_{clear}\vartheta_{clear} + \lambda_{head}\vartheta_{head} + \lambda_{vel}\vartheta_{vel} + NF1. \quad (8)$$

The $NF1$ navigation function [7] provides the global path information. This wave-propagation technique generates a channel of free-space connecting the start and goal location. At any point in the channel the gradient information on reduction of distance to the goal can be obtained.

Although the local minima in Eq. (8) is avoided, problem of determing weighting factors λ_{clear}, λ_{head} and λ_{vel} that influence the net performance of the robot motion significantly remains unsolved.

As opposed to $NF1$ navigation function, the graph based search algorithms provide a single cell path from start to goal. Therefore, a certain measure of alignment for the trajectories generated by dynamic window module to the global geometric path can be provided.

Our approach may be compared to the Reduced Dynamic Window Approach given in [16], where the translation velocities on different A* geometric path curvatures were considered. However, in our implementation no predefined set of velocities is used a-priori.

IV. Integration of Dynamic Window and Path Planning Module

In [13] a criterion is introduced that makes comparison of the current possible robot trajectories to the geometric global path. Both target heading $\vartheta_{head}(v,\omega)$ and linear velocity measure $\vartheta_{vel}(v,\omega)$ are incorporated in a single path alignment measure $\vartheta_{path}(v,\omega)$. The global objective function now accounts only for clearance $\vartheta_{clear}(v,\omega)$ and path alignment $\vartheta_{path}(v,\omega)$ measure:

$$\Gamma(v,\omega) = \lambda\vartheta_{clear} + (1-\lambda)\vartheta_{path}. \quad (9)$$

Path following criterion relates possible robot trajectories with the local geometric path configuration and thus improves the restrictions on the translational velocity and also determines the local target heading direction. This restriction of the linear velocity is the key element since the velocity is adaptive according to the local path configuration.

In this work the so called ***effective path*** is introduced. Essentially, it is the straight line segment connecting the current robot position and the reference point on the path. Its orientation determines the current reference orientation of the robot in relation to the local path configuration affecting the rotational velocity of the robot ω_{ref} and its length determines what the optimal translational velocity v_{ref} should be. Reference point on the path has assigned constant time T_{max} (admissible collision time) as a fixed traveled time from the current robot position to the reference point.

The nominal reference point position is at the point immediately before the second path direction change (path direction changes are multiples of 45°), as can be seen in Fig. 2. This assumption is based on the fact that detecting the first path direction change which altered the path direction for $\pm 45^\circ$ starting from the robot position is not enough to determine the tendency of the path change thereafter. The second path direction change then determines whether the path direction changed back to its original direction or continued changing which signifies a stronger curvature change and therefore gives a good local curve tendency information.

Due to dynamic constraints of the robot, a distance between the reference point position on global path and robot current position $R(v_c)$ is upperbounded by a maximum translational velocity $v_c + \dot{v}_a\Delta t$, that can be accessed in the next time interval Δt for the current robot velocity vector (v_c,ω_c), and the time look-ahead T_{max} (see Fig. 3):

$$R_{max}(v_c) = (v_c + \dot{v}_a\Delta t)T_{max}. \quad (10)$$

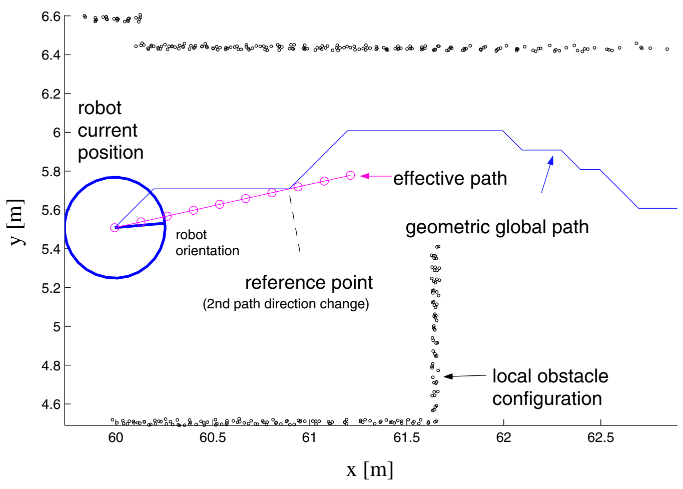

Fig. 2. Determing the nominal reference point position on the global geometric path

Therefore, its length is adaptively increased as the robot speeds up or vice-versa.

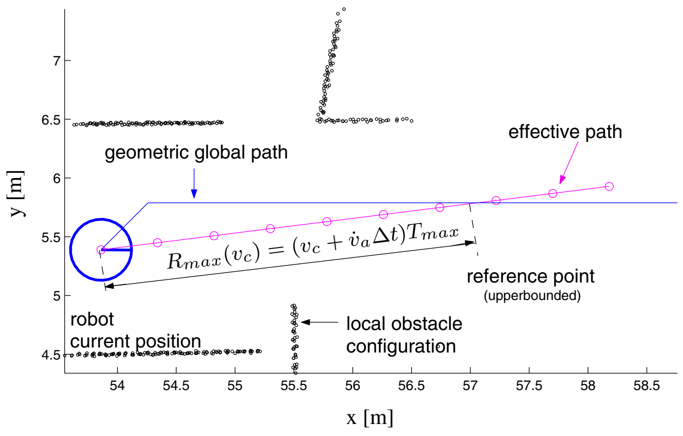

Fig. 3. Determing the upperbounded reference point position on the global geometric path

If the path direction change is detected in the immediate vicinity of the robot (for instance, the grid-cell next to the robot position), its direction change impact on the robot is not significant. Therefore, a distance between the reference point position on global path and robot current position is lowerbounded with R_{min} that filters out the insignificant path direction changes that would slow down the robot but would not change its global position much (see Fig. 4). It is effectively set according to the maximal breakage time $T_{b_{max}} = \frac{v_{max}}{\dot{v}_b}$ so the robot could stop along the minimal effective path length if it previously achieved maximal translation velocity:

$$R_{min} = v_{max} T_{b_{max}} - \frac{1}{2}\dot{v}_b T_{b_{max}}^2 = \frac{1}{2}\dot{v}_b T_{b_{max}}^2. \tag{11}$$

Therefore, a minimal reference translation velocity would be $v_{min} = \frac{R_{min}}{T_{max}}$.

Path alignement measure $\vartheta_{path}(k)$ for a robot trajectory is defined according to the following expression [13]:

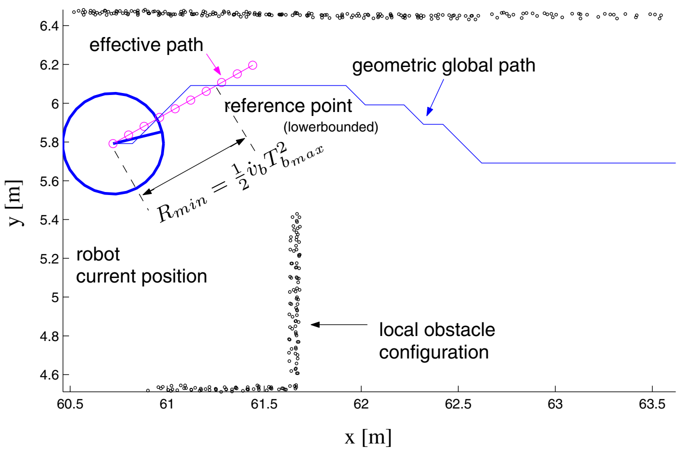

Fig. 4. Determining the lowerbounded reference point position on the global geometric path

$$\vartheta_{path}(v, \omega) = 1 - \frac{\sum_{i=1}^{N_t} \sum_{j=1}^{N_p} j d_{ij} - D_{min}}{D_{max} - D_{min}}. \tag{12}$$

A trajectory generated by the dynamic window, which is a circular arc, is represented by a discrete set of points N_t. Trajectory length L_t is set according to the velocity touple (v, ω) which characterize current trajectory and the time look-ahead T_{max} beyond which all trajectories are considered clear of obstacles:

$$L_t(v, \omega) = v T_{max}. \tag{13}$$

N_p is a set of points on the effective path and d_{ij} is the Euclidian distance between the i-th point on the trajectory and j-th point on the effective path. The number of points on both curve is fixed. When choosing these numbers one must consider computation complexity in one hand and a quality of approximation of the curves in another hand. In our work, we have assumed number of $N_p = 10$ points to be valid aproximation of straight line segment and number of $N_t = 30$ points to be valid approximation of circular trajectories (see Fig. 5). When the

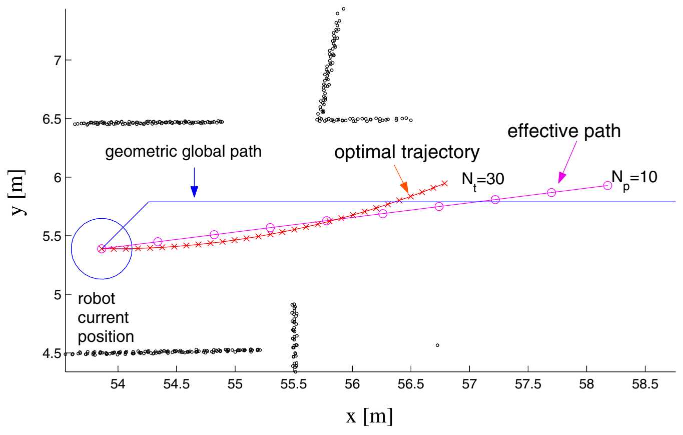

Fig. 5. The effective path and robot trajectory comparison

number of points is fixed, resolution of the curves depends only on the velocities: higher velocities mean more rarely distributed points. The limit values $D_{min} = \min_{(v,\omega)} \left\{ \sum_{i=1}^{N_t} \sum_{j=1}^{N_p} j d_{ij} \right\}$ and $D_{max} = \max_{(v,\omega)} \left\{ \sum_{i=1}^{N_t} \sum_{j=1}^{N_p} j d_{ij} \right\}$ are used for normalization along all possible robot trajectories. The index factor j that weights the distance contributions is introduced to penalize more the external points at the end of the trajectories which define the trajectory deviation from the global path more than the starting points on different trajectories, since they all start from the same robot position. Essentially, the Eq. (12) gives a measure of side area between a trajectory and the effective path. However, it does not represent directly the surface between a trajectory and the effective path. This is chosen due to the fact that criterion based on the surface may be ill defined for the cases where a trajectory and the effective path are almost perpendicular to each other.

Notice that according to Eq. (12) the trajectory that would receive the best path alignment, if the heading was the same as that of the effective path, would be the one that has $\frac{2}{3}$ of the effective path length. That conclusion comes from the fact that the center of gravity of the weighted sum

$$D_{i=1} = \sum_{j=1}^{N_p} j d_{ij} = d_{i1} + 2d_{i2} + ... + N_p d_{iN_p} \tag{14}$$

is

$$c_g = \frac{2}{3}(N_p - 1)\Delta d, \tag{15}$$

where $\Delta d = d_{i(j+1)} - d_{ij}$ is considered constant since points on the effective path are evenly distributed and $(N_p - 1)\Delta d$ presents total length of the effective path. Therefore, to encourage the maximum translational velocity, once the reference point position is found (and therefore effective path orientaion fixed) the effective path lenght is de-facto enlarged to $\frac{3}{2}$ of its size:

$$L_{e_{max}(v_c)} = \frac{3}{2}(v_c + \dot{v}_a \Delta t)T_{max}. \tag{16}$$

The net effect is that the maximum translational velocity receives the maximum path alignement if the robot heading is aligned with the effective path orientation. This case is depicted

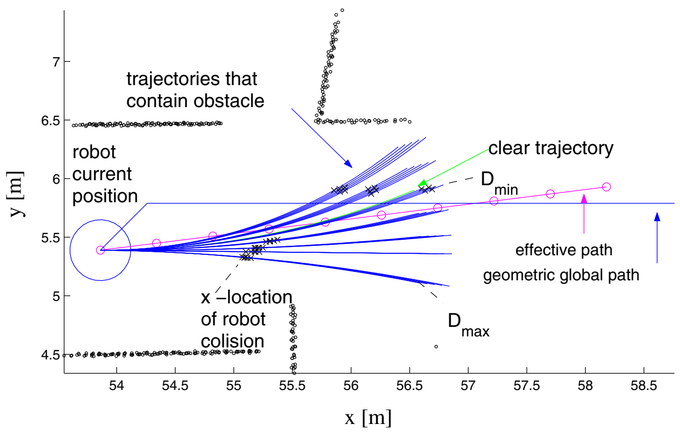

Fig. 6. Dynamic Window trajectories and effective path: The maximum velocity case

in Fig. 6. Traversing the straight line path robot is able to speed up to the maximum velocity. All possible trajectories are almost of the same length and that is $\frac{2}{3}$ of the effective path. The location on trajectories that contains obstacle where robot would colide with an obstacle is denoted by x. Each mark includes contact between an obstacle and an enlarged safety contour margin around the robot which size depends on robot current velocity (in [6] it is called speed dependent side clearance, SC_v). Trajectory that gives maximal side area with effective path is noted with limit value D_{max} and the minimal side area is noted with D_{min}.

V. Experimental Results

Our motion control method has been implemented and tested on mobile robot Pioneer 2DX. Robot is equipped with SICK laser range finder that is used to detect dynamic obstacles and to update occupancy grid map information. Obstacles considered by the dynamic window module were represented as point object. The cells in grid map were C-space enlarged prior to path search algorithm to account for robot dimensions. Partially known information about the state of the environment is given a-priori. The experiments were taken at our department.

Experimental run is presented in Fig. 7. Thicker lines of

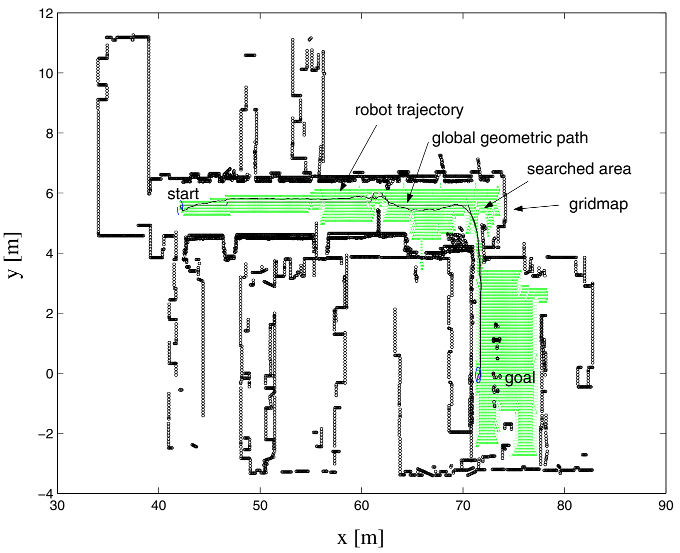

Fig. 7. Experimental run with an average velocity of 0.41m/s

the walls correspond to the on-line detected obstacles which are incrementally incorporated into grid map. The zigzag curvature represents geometric global path computed off-line from the start point to a particular goal point, which is set by a human operator. The smooth curvature represents the true robot trajectory. Robot was able to follow the planned optimal path well.

The experiments were taken with different values of scalar $\lambda \in [0.3, 0.9]$ where no greater diferences in robot traverse were noticed. Recommended value of scalar λ is 0.5.

Velocity profiles are represented in Figs. 8 and 9. Translation velocity is limited to $v_{max} = 0.6$ m/s and rotational velocity to $\omega_{max} = 1.75$ rad/s. Accelerations are limited to $\dot{v}_a = 0.5$ m/s^2 and $\dot{\omega}_a = 0.87$ rad/s^2 and breakage decelerations are limited to $\dot{v}_b = 0.5$ m/s^2 and $\dot{\omega}_b = 0.87$ rad/s^2. In the presented experi-

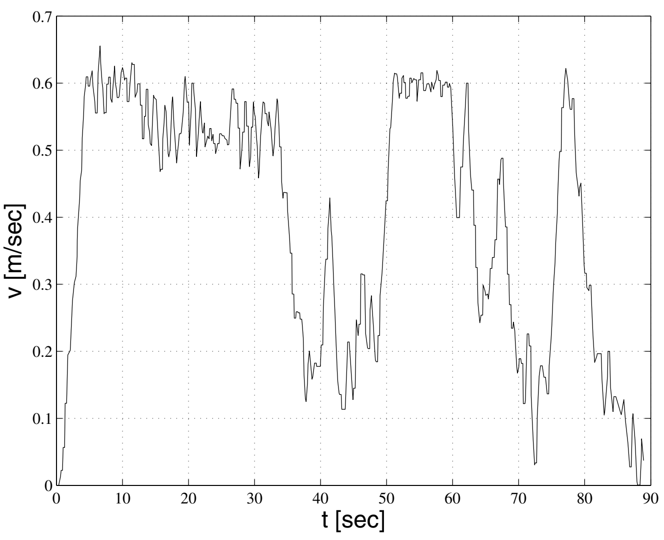

Fig. 8. Transversal velocity profile

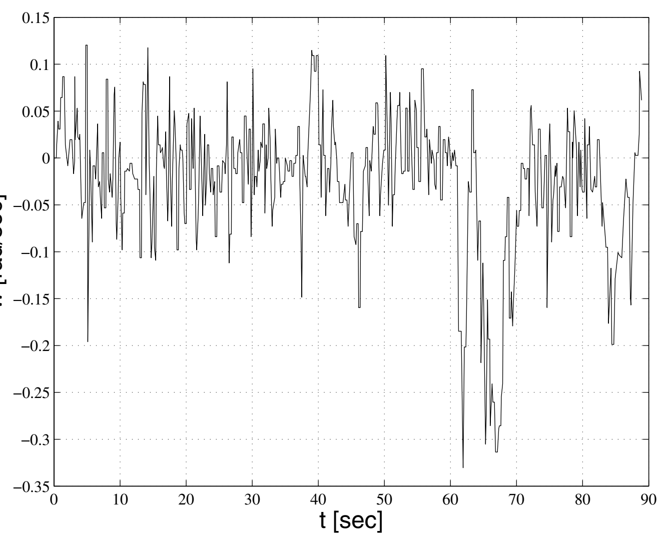

Fig. 9. Rotational velocity profile

mental run calibrated odometry [17] and Monte Carlo external localization procedure were used [18].

VI. Conclusion and Future Work

In this paper a novel integrated approach to real-time mobile robot control is proposed, which integrates focussed D* graph search algorithm for path planning and dynamic window module for generation of possible robot trajectories. Integration is performed by a single path alignment measure based on geometric comparison between possible robot trajectories and the so called effective path. The effective path is determined according to detection of a reference point on the global geometric path where the path direction starts changing significantly by observing path direction change points along the path. The length and orientation of the effective path directly determines optimal reference velocity vector in the next sampling instant that is a combined objective of obstacle clearance and path alignment. Although the straight line segments were used to generate immediate effective path lengths to which the dynamic window trajectories were compared to, the proposed scheme may be applied to any smooth path curvature if an appropriate effective path length is found. Motion control method was tested on a Pioneer 2DX mobile robot using laser range finder. Secure and smooth motion of the robot traverse was obtained in cluttered environment. Creating occupancy grid map using sonar will be elaborated further.

Acknowledgments

This research has been supported by the Ministry of Science, Education and Sports of the Republic of Croatia under grant No. 0036018.

References

[1] O. Khatib, “Real-Time Obstacle Avoidance for Manipulators and Mobile Robots,” *The International Journal of Robotics Research*, 5(1), pp. 90–98, 1986.

[2] J. Borenstein and Y. Koren, “The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots,” *IEEE Transactions on Robotics and Automation*, 7(3), pp. 278-288, 1991.

[3] J. Minguez and L. Montano, “Nearness Diagram Navigation (ND): A New Real Time Collision Avoidance Approach,” *IROS’00, IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000.

[4] R. Simmons, “The Curvature-Velocity Method for Local Obstacle Avoidance,” *ICRA’96, IEEE International Conference on Robotics and Automation*, 1996.

[5] R. Simmons, “The Lane-Curvature Method for Local Obstacle Avoidance,” *IROS’98, IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1998.

[6] D. Fox and W. Burgard and S. Thrun, “The Dynamic Window Approach to Collision Avoidance,” *IEEE Robotics & Automation Magazine*, 4(1), pp. 23-33, 1997.

[7] J.C. Latombe, *Robot Motion Planning,* Kluwer Academic Publishers, Dodrecht, Netherlands, 1991.

[8] O. Brock and O. Khatib, “High-speed navigation using the Global Dynamic Window Approach,” *ICRA’99, IEEE International Conference on Robotics and Automation*, 1999.

[9] S. Quinlan and O. Khatib, “Elastic Bands: Connecting Path Planning and Control,” *ICRA’93, IEEE International Conference on Robotics and Automation*, 1993.

[10] J. Stuart and R. Norvig and P. Norvig, *Artificial Intelligence - A Modern Approach,* Prentice Hall Series, 1995.

[11] A. Stentz, “Optimal and Efficient Path Planning for Partially - Known Environments,” *ICRA’94, IEEE International Conference on Robotics and Automation*, 1994.

[12] A. Stentz, “The Focussed D* Algorithm for Real-Time Replanning,” *International Joint Conference on Artificial Intelligence*, 1995.

[13] K. Maček and I. Petrović and E. Ivanjko, “An Approach to Motion Planning of Indoor Mobile Robots,” *ICIT’03, IEEE International Conference on Industrial Technology*, pp. 969-973, 2003.

[14] R. Philippsen and R. Siegwart, “Smooth and Efficient Obstacle Avoidance for a Tour Guide Robot,” *ICRA’03, IEEE International Conference on Robotics and Automation*, 2003.

[15] C. Stachniss and W. Burgard, “An Integrated Approach to Goal-directed Obstacle Avoidance under Dynamic Constraints for Dynamic Environments,” *IROS’02, IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.

[16] K.O. Arras and J. Persson and N. Tomatis and R. Siegwart, “Real-Time Obstacle Avoidance for Polygonal Robots with a Reduced Dynamic Window,” *ICRA’02, IEEE International Conference on Robotics and Automation*, 2002.

[17] E. Ivanjko, I. Petrović, N. Perić “An approach to odometry calibration of differential drive mobile robots,” *EDPE’03, International Conference on Electrical Drives and Power Electronics*, pp. 519-523, 2003.

[18] K. Konolige and K. Chou, “Markov Localization using Correlation,” *International Joint Conference on Artificial Intelligence*, pp. 1154-1159, 1999.