International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 3, Issue 11, pp 19-25, November - 2019

Combining Algorithm Classes for Navigation Tasks of Mobile Robots

Zlata Jelačić

Assistant Professor, Faculty of Mechanical Engineering, University of Sarajevo, Bosnia and Herzegovina

***Abstract* - A motion planner for mobile robots is commonly built out of a number of different algorithms that solve the two steps of motion planning: representing the robot and its environment and searching a path through the represented environment. However, the available literature on motion planning lacks a generic methodology to arrive at a combination of representations and search algorithm classes for a practical application. This paper presents a recipe to select appropriate algorithm classes that solve both steps of motion planning and to select a suitable approach to combine those algorithm classes. The recipe is verified by comparing its outcome to the motion planners that has been successfully applied on robots in practice.**

Keywords: service robots, motion planning, mobile robots, algorithm classes, representation classes.

I. INTRODUCTION

Motion planning considers the planning of a path for a robot in an environment that can contain dynamic and static obstacles and is generally characterized by uncertainty. A motion planner can serve three different goals: mapping, covering and navigation. The focus of this paper is on motion planning for mobile robots moving on the ground plane with navigation as a goal. Motion planning typically consists of two steps [1], namely:

1. Representing the robot and its environment, and
2. Searching a path in the represented environment.

These representations and search algorithms are abundant in literature [2], [3] and [4]. Furthermore, as a single algorithm generally does not suffice for a practical application, planners consist of combinations of algorithms in order to satisfy the designer's requirements [5-7]. Because combinations of possibly approximate algorithms are deemed necessary in practice, various approaches have been presented in literature [2], [4] and [7].

However, the available literature lacks a generic methodology to arrive at a combination of representations and search algorithms that can handle all requirements. The selection of an appropriate combination of representations and search algorithms depends on i) the desired behavior as specified by the designer, ii) the dynamics and the kinematics of the robot at hand, iii) the dynamics of the environment at hand, and iv) the presence of uncertainty. Typically, these conditions are translated into seven requirements, serving as a basis for the selection of these algorithms:

- ***Completeness:*** the guarantee that either a solution is returned if one exists or failure is returned if no solution exists. Completeness commonly only refers to search algorithms but in fact also concerns the representations as poor design choices for the representation may prohibit any search algorithm from returning a solution even though one exists.
- ***Optimality:*** a solution can be optimal according to criteria such as distance, time, energy use, etc.
- ***Correctness:*** the guarantee that if a solution is returned, this will lead the robot to its goal.
- ***Dealing with dynamic constraints:*** a robot has dynamic constraints, e.g., a limited acceleration, and can also be kinematically constrained, e.g., a car cannot move sideways.
- ***Robustness against a dynamic environment:*** the environment changes if obstacles can move.
- ***Robustness against uncertainty:*** the representation of the robot and obstacles generally is uncertain and the environment can be (partially) unknown.
- ***Computational complexity:*** a measure of memory usage and time to acquire a solution. Keeping the computational complexity tractable usually comes at the expense of one or more of the previously mentioned criteria.

In practice, a motion planner is typically desired to be complete and to return an optimal and correct solution in real-time. However, there exists no motion planner able to achieve all these requirements while satisfying the robot constraints in presence of moving obstacles and uncertainty [8]. Therefore, the motion planning problem at hand is often approximated by a simplified but still realistic version [2], which introduces a trade-off between the various requirements.

The small amount of literature that discusses combinations of representations and search algorithms is not generic. Either only a single step of the motion planning problem, i.e. either the representation or the search algorithm, is discussed or, in more cases, not all requirements are discussed as some are considered not relevant for a specific robot or goal. In [9] some combinations of algorithms are discussed that, given a specific representation of the environment, find a path. The work in reference [10] gives an overview of algorithms that represent the environment and algorithms that find a path and discusses combinations of both. How the combined algorithms can deal with robot constraints is not discussed. There is also an overview of combinations of algorithms that find a path for robots with certain kinematic constraints [11]. It does however not consider how to deal with moving obstacles or a partially known environment. Also, motion planners are compared that combine multiple algorithms but with a focus on unmanned aerial vehicles (UAVs) ([7] and [12]) instead of mobile robots moving only on the ground plane.

Moreover, this does not consider how combined algorithms provide robustness against a dynamic environment. Next, there is an overview that discusses planners that have covering as a specific goal [3]. These planners assume a mostly static environment and thereby exclude robustness against a dynamic environment. Also, robot constraints are not taken into account. There are examples of combinations of multiple algorithms that make a well-performing motion planner in practice for a specific mobile robot [13-18]. Such motion planners give insight in the choice for a combination of algorithms with respect to predefined requirements, but they do not give generic directions for possible combinations.

This paper contributes by presenting a recipe for the selection of a combination of motion planning algorithms for navigation of mobile robots. This recipe is intended for either novices in robotics or for experts on one of the many other topics in robotics who require navigation on their mobile robot for a practical application. Carefully considering this recipe and the discussed design criteria can be a first step in designing a navigation system for a mobile robot.

II. SELECTING AN APPROACH TO COMBINE ALGORITHM CLASSES

In this section the available approaches to combine motion planning algorithms are discussed, namely: a coupled, a decoupled, a hierarchical and a reactive approach. Selecting an appropriate approach to combine motion planning algorithms is of paramount importance as this system design serves as a guideline for the subsequent design choices. Many of the design decisions made in this stage concern the available environment models, e.g., the availability and uncertainty in these models and the presence and predictability of moving obstacles are already discussed in this section. Depending on the requirements of the application these approaches have to be combined with additional features, such as re-planning or a bounded uncertainty region. A recipe, visualized in Figure 1, is presented to select an approach based on the requirements that are introduced in Section II. This section primarily adopts the terminology as used in [1] and [2].

a) Approaches to combine algorithm classes

First of all, in trajectory planning or a coupled approach, the robot's dynamic constraints are directly satisfied in the search for a solution [19]. These methods search for a solution in the state space S [20]. A state lattice is an example of a representation for planning in the state space satisfying the robot's dynamic constraints ([4], [6]). Methods that search directly in the state-time space ST are called state×time search methods [21]. The notion of time allows the planner to plan motions that are optimal in the sense of time in the presence of moving obstacles. A coupled approach is generally computationally expensive [8]. To keep computations tractable, such an approach requires algorithms that exploit some form of approximation to arrive at a solution. The completeness of these approximation algorithms is difficult and impractical to prove [7].

A second approach is to decouple the search for a solution by first searching for a partial solution in the configuration space C. This decoupled approach, also called path planning, generally consists of the following steps [4]:

- Path planning: obtain a collision-free path in C.
- Path transformation: ensure that a path satisfies any kinematic constraints in S ([11], [19]).
- Trajectory planning: use a timing function to transform a path into a trajectory that satisfies dynamic constraints.

If this sequence is required to deal with moving obstacles, decoupling into a path planning part and a motion timing part is also possible. This approach follows the first two steps of the decoupled trajectory planning approach, but the trajectory that is formed in the third step also accounts for moving obstacles. Recent approaches to create trajectories from a path are CHOMP [22] and STOMP [23], which use optimization techniques to minimize some cost function while satisfying the dynamic constraints of the robot. Planning in the configuration space C is computationally less expensive than in the state space S due to the lower dimension. The inherent consequence is a negative influence on the completeness and optimality of the approach [2]. For example, a path obtained in C with sharp turns might be unfeasible for a non-holonomically constrained car. Also, a single path, obtained by a search in C, can be transformed into a time-optimal trajectory but this trajectory is not guaranteed to be globally time-optimal.

Third, a hierarchical approach acquires a solution in two or more consecutive plans ([7],[10]). This approach uses a planner to find a solution to the goal, using the configuration space or the full-dimensional solution space, followed up by a planner that searches for a solution in the vicinity of the robot using the full-dimensional solution space. This is also known as global motion or path planning and local motion planning. Similar to the decoupled approach this is a decoupling that sacrifices optimality, but it is not necessarily incomplete [24]. A hierarchical approach is also possible in C. One could first plan a global path that only considers position and subsequently use a local planner that also takes the orientation into account and hence plans in a higher dimension.

The fourth approach is a reactive approach or sensor-based approach. This approach is based on a class of algorithms that are also known as obstacle avoidance algorithms. Rather than searching for a path or trajectory on a certain representation, these compute an instantaneous motion that avoids collision based on the latest sensor information.

This approach does not need a representation in the form of a metric map as opposed to the previously discussed approaches. They can be divided into four classes: potential field [25], first order methods or velocity obstacle methods [26], receding horizon control ([27], [28]) and the (global) nearness diagram ([29], [30]). Typically, these methods only compute a desired direction of motion and hence the amount of planning is reduced to a minimum. However, as such an approach does not consider the planning problem in the entire world space, there is no guarantee that the resulting trajectory will lead to the goal. These local minima are a well-known drawback of reactive methods. Furthermore, the resulting trajectory is hardly ever optimal. Therefore, a reactive planner is typically used in combination with a global planner and hence it can be considered as a special form of a hierarchical approach. Besides the selection of a global planner and a reactive planner, the integration of these modules is important.

One should, e.g., consider how the system should react if either of the planners cannot compute a solution, whether a global plan is computed every sample, etc. Examples of approaches combining global planners with reactive methods can be found in, e.g. [13], [14], [18] and [31].

b) Recipe to select an approach to combine algorithm classes

The following design criteria, that are numbered A1 through A7, form a recipe to select a suitable approach to combine motion planning algorithms. The criteria are based on the requirements that are introduced in Section I. The recipe is visualized in Figure1.

A1) Is the environment known? In an environment that is not completely known, a plan can become unfeasible as sensor data is acquired during execution and thus a new plan is required. This requires re-planning. A re-plan can be acquired by re-computing the entire representation of the solution space, or by updating the representation and performing a new search.

A2) Is there uncertainty in the obstacle representation? Uncertainty in the representation of obstacles, e.g., uncertainty due to imperfect sensors, is assumed to be contained in a bounded uncertainty region [2]. In practice, a motion planner gains robustness by inflating the obstacle representation with a distance that is equal to the present position uncertainty of obstacles. This robustness, however, comes at the expense of completeness: if obstacles are inflated too much, the planner might not be able to find a solution anymore.

A3) Does the environment contain moving obstacles? To be complete and to get an optimal solution in the presence of moving obstacles it is necessary to represent their trajectories with a time dimension in the solution space. Nevertheless, this significantly increases computational complexity. Re-planning is also possible to deal with a changing environment, but this will result in suboptimal plans and the planner might not be complete. A further alternative to re-planning is to update or modify the existing plan. An example of this approach is elastic bands [32].

A4) Can the motion of obstacles be predicted? If obstacles are predictable, i.e., have a known trajectory, it is necessary to add a time dimension to the solution space, yielding a state-time space ST, in order to be complete and to ensure a time-optimal solution. A trajectory can be modeled using a prediction model if obstacles have no exactly known trajectory. For example, an obstacle position can be extrapolated in time using a maximum acceleration model. In general, one can say that the effectiveness of including a prediction model depends on the amount of uncertainty in the prediction. As the inflation of the obstacle representation increases, the number of possible solutions decreases, and the remaining ones will be further from optimality. A time dimension will unnecessarily increase the dimension of the solution space if obstacles are not predictable. Re-planning and using a reactive approach are alternatives to deal with moving obstacles. The inherent consequence is that the solution will not be optimal, and the planner is possibly incomplete.

A5) Is a solution in the form of a plan necessary in the full dimension and size of the solution space? A coupled approach searches for a solution in the full dimension of the solution space and therefore it can be complete and return optimal solutions. However, the complexity of a representation is exponential with the number of dimensions of the solution space. A coupled approach can thus be computationally too complex to achieve a solution within a designer's desired time constraint. The complexity can be negotiated using a hierarchical, reactive or decoupled approach but this is generally at the cost of completeness and optimality.

A6) Is a solution in the form of a plan necessary and possible within a certain time constraint? Achieving a solution in the form of a plan that avoids collision can be too time-consuming. An environment can change so rapidly due to moving obstacles and uncertainty that a plan can become unfeasible before the robot is able to even execute it. In such a case planning becomes less relevant and a reactive approach is necessary. As a reactive approach typically only considers the latest sensor information, it is not complete, and its solution is not optimal. One must also question whether an explicit plan is actually necessary. It can be sufficient to apply a reactive approach that uses the structure of the environment to control the robot to its desired goal pose instead of pre-computing the entire motion.

A7) Local planning possible? A hierarchical approach is generally preferred as a local planner in this approach still obtains a solution in the full solution space. Therefore, there is a better chance of finding a solution and this solution is generally closer to optimality than in case of a decoupled approach. Despite a lower computational complexity, a hierarchical approach might still not be able to return a solution in time. A decoupled approach that is generally of a lower computational complexity could then be used.

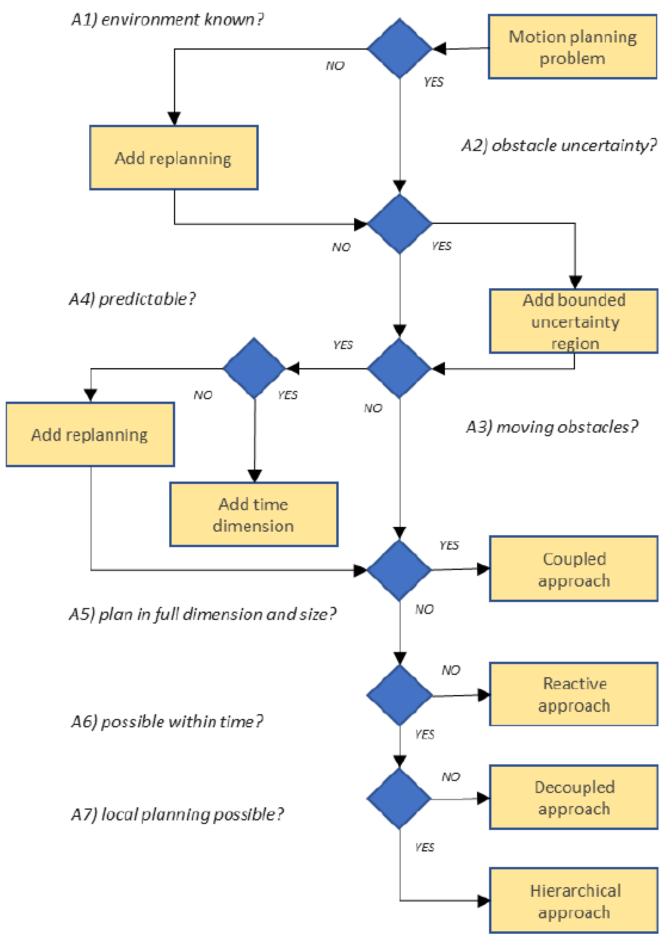

Figure 1: Approach to combine motion planning algorithms

III. VERIFICATION

Three use cases from literature involving the application of a motion planner in practice are evaluated to verify the proposed recipe: i) the PR2 service robot that autonomously navigated a marathon distance of 42.15 km in a real office environment [34], and ii) a flexible automated guided vehicle (AGV) in a partially structured warehouse [35].

a) Application of the recipe to use case I

The PR2 service robot has an omni-directional base platform with a top speed of 1 m/s. It navigates in an office environment that is cluttered with a variety of arbitrarily shaped obstacles [18].

Selecting an approach to combine algorithm classes: The office environment is only partly known so the robot must re-plan as it receives new information about the world space (A1). The depth accuracy of the sensors that are used for navigation is 1.5 cm. Hence, the resulting uncertainty must be captured in a bounded uncertainty region (A2). The office environment is populated by people, so the environment contains moving obstacles (A3). The trajectories of people are unpredictable (A4) and the world space is only partially known. Therefore, the representation of the world space will change rapidly. As the robot is desired to navigate at a reasonable speed of about 0.5 m/s, a reactive approach is the most suitable approach to avoid collision in real-time (A5, A6). A reactive approach requires an obstacle avoidance algorithm and a global planner.

b) Application of the recipe to use case II

Contrary to many AGV's, the fork-lift truck considered [35] incorporates a high degree of on-board autonomy to be flexible with respect to floor layout changes and to decrease the amount of manual work when establishing the a priori knowledge of the workplace.

Selecting an approach to combine algorithm classes: Re-planning is required due to the presence of unknown and possibly dynamic obstacles such as other vehicles (A1). Again, a finite sensor resolution demands a bounded uncertainty region (A2). The environment does contain moving obstacles (A3) but there is no prediction of their velocity or future position (A4). It is not necessary to plan in the full state space S (A5) but a plan in the reduced space is necessary to get as close as possible to optimality (A6). Due to the presence of kinematic constraints, a hierarchical approach is necessary (A7).

IV. CONCLUSION AND FUTURE WORK

This paper presents a recipe for an appropriate approach to combine algorithm classes for a mobile robot that moves on a ground plane. By considering both steps of a motion planner, for all requirements as introduced in Section I, the recipe can be used to find an appropriate combination of motion planning algorithms classes.

The recipe is verified with two use cases where existing motion planners are successfully applied in practice. The recipe's outcome resembles the existing motion planners and the recommendations from the recipe reveal the points for improvement that are indicated by the original authors. It can therefore be concluded that the recipe is beneficial to the selection of an appropriate motion planning approach and representations and search algorithm classes. To verify the recipe more extensively, future work will apply the recipe to more robots which already feature a motion planner that functions well in practice. Furthermore, this research should be expended with the analysis into robot representation and selection of a suitable class of search algorithms given an approach to combine motion planning.

REFERENCES

[1] Hwang Y, Ahuja N (1992) Gross motion planning - a survey. *ACM Computing Surveys* 24(3):219-291.

[2] Latombe J (1990) Robot motion planning. *Springer.*

[3] Choset H (2005) Principles of robot motion: theory, algorithms, and implementation. *MIT Press.*

[4] LaValle S (2006) Planning algorithms. *Cambridge University Press.*

[5] Kant K, Zucker SW (1986) Toward efficient trajectory planning: The path-velocity decomposition. *International Journal of Robotics Research* 5(3):72-89.

[6] Brock O, Khatib O (1999) High-speed navigation using the global dynamic window approach. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, vol 1, pp 341-346.

[7] Goerzen C, Kong Z, Mettler B (2010) A survey of motion planning algorithms from the perspective of autonomous UAV guidance. *Journal of Intelligent & Robotic Systems* 57(1-4):65-100.

[8] Canny JF (1988) Complexity of robot motion planning. *MIT press.*

[9] Ferguson D, Likhachev M, Stentz A (2005) A guide to heuristic-based path planning. In: Workshop on Planning under Uncertainty for Autonomous Systems at *International Conference on Automated Planning and Scheduling.*

[10] Giesbrecht J (2004) Global path planning for unmanned ground vehicles. Tech. Rep.DRDC`Suffield TM 2004-272, *Defence R&D Canada – Suffield.*

[11] Laumond JP, Nissoux C (2000) Visibility-based probabilistic roadmaps for motion planning. *Advanced Robotics* 14(6):477-493.

[12] Dadkhah N, Mettler B (2012) Survey of motion planning literature in the presence of uncertainty: considerations for UAV guidance. *Journal of Intelligent & Robotic Systems* pp 1-14.

[13] Thrun S, Bennewitz M, Burgard W, Cremers AB, Dellaert F, Fox D, Hahnel D, Rosenberg C, Roy N, Schulte J, et al (1999) MINERVA: A second-generation museum tour-guide robot. In: Proceedings of the *IEEE International Conference on Robotics and Automation*, vol 3.

[14] Stachniss C, Burgard W (2002) An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments. In: *IEEE/RSJ Int. Conf. on Intell. Robots and Systems*, vol 1, pp 508-513.

[15] Hsu D, Kindel R, Latombe JC, Rock S (2002) Randomized kinodynamic motion planning with

moving obstacles. *International Journal of Robotics Research* 21(3):233-255.

[16] van den Berg J, Ferguson D, Kuffner J (2006) Anytime path planning and replanning in dynamic environments. In: Proceedings of the *IEEE International Conference on Robotics and Automation*, pp 2366-2371.

[17] Dolgov D, Thrun S, Montemerlo M, Diebel J (2008) Practical search techniques in path planning for autonomous driving. *Ann Arbor* 1001.

[18] Marder-Eppstein E, Berger E, Foote T, Gerkey B, Konolige K (2010) The office marathon: Robust navigation in an indoor office environment. In: Proceedings of the *IEEE International Conference on Robotics and Automation*, pp 300-307.

[19] Donald B, Xavier P, Canny J, Reif J (1993) Kinodynamic motion planning. *Journal of the Association for Computing Machinery* 40(5):1048-1066.

[20] Pivtoraiko M, Knepper RA, Kelly A (2009) Differentially constrained mobile robot motion planning in state lattices. *Journal of Field Robotics* 26(3):308-333.

[21] Fraichard T (1998) Trajectory planning in a dynamic workspace: a `state-time space' approach. *Advanced Robotics* 13(1):75-94.

[22] Ratliff N, Zucker M, Bagnell J, Srinivasa S (2009) CHOMP: Gradient optimization techniques for efficient motion planning. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, pp 489-494.

[23] Kalakrishnan M, Chitta S, Theodorou E, Pastor P, Schaal S (2011) STOMP: Stochastic trajectory optimization for motion planning. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, pp 4569-4574.

[24] Zhang H, Butzke J, Likhachev M (2012) Combining global and local planning with guarantees on completeness. In: *IEEE International Conference on Robotics and Automation*, pp 4500 -4506.

[25] Khatib O (1986) Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research* 5(1):90.

[26] Fiorini P, Shiller Z (1998) Motion planning in dynamic environments using velocity obstacles. *International Journal of Robotics Research* 17(7):760-772.

[27] Fox D (2001) KLD-sampling: Adaptive particle filters and mobile robot localization. *Advances in Neural Information Processing Systems* 14(1):26-32.

[28] Ogren P, Leonard NE (2005) A convergent dynamic window approach to obstacle avoidance. *IEEE Transactions on Robotics* 21(2):188-195.

[29] Minguez J, Montano L (2000) Nearness diagram navigation (ND): a new real time collision avoidance approach. In: Proceedings of the *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol 3, pp 2094-2100.

[30] Minguez J, Montesano L, Montano L (2004) An architecture for sensor-based navigation in realistic dynamic and troublesome scenarios. In: Proceedings of the *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol 3, pp 2750-2756, Maravall.

[31] Philippsen R, Kolski S, Macek K, Jensen B (2008) Mobile robot planning in dynamic environments and on growable costmaps. In: Workshop on Planning with Cost Maps at the *IEEE International Conference on Robotics and Automation.*

[32] Quinlan S, Khatib O (1993) Elastic bands: connecting path planning and control. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, vol 2, pp 802-807.

[33] Zavlangas P, Tzafestas S (2002a) Integration of topological and metric maps for indoor mobile robot path planning and navigation. *Methods and applications of artificial intelligence* pp746-746.

[34] Urmson C, Anhalt J, Bagnell D, Baker C, Bittner R, Clark M, Dolan J, Duggins D, Galatali T, Geyer C, et al (2008) Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics* 25(8):425-466.

[35] Martinez-Barbera H, Herrero-Perez D (2010) Autonomous navigation of an automated guided vehicle in industrial environments. *Robotics and Computer-Integrated Manufacturing* 26(4):296-311.

AUTHOR'S BIOGRAPHY

Zlata Jelačić is Assistant Professor in Department of Mechanics at the Faculty of Mechanical Engineering, University of Sarajevo, Bosnia and Herzegovina. PhD thesis was in the field of rehabilitation robotics, namely the development of active above-knee prosthesis with actuated knee and ankle joints.

Citation of this Article:

Zlata Jelačić, "Combining Algorithm Classes for Navigation Tasks of Mobile Robots" Published in *International Research Journal of Innovations in Engineering and Technology (IRJIET)*, Volume 3, Issue 11, pp 19-25, November 2019.

****