Ref. Ares(2017)2117980 - 25/04/2017

Mobile Robots with Novel Environmental Sensors for Inspection of Disaster Sites with Low Visibility

Grant agreement no: 645101

Project start: January 1, 2015

Duration: 3.5 years

Deliverable 5.4

Navigation algorithms for hazardous environments

Due date: M24

Lead beneficiary: ORU

Dissemination Level: Public

Main Authors:

Robert Krug (ORU)

1. Introduction

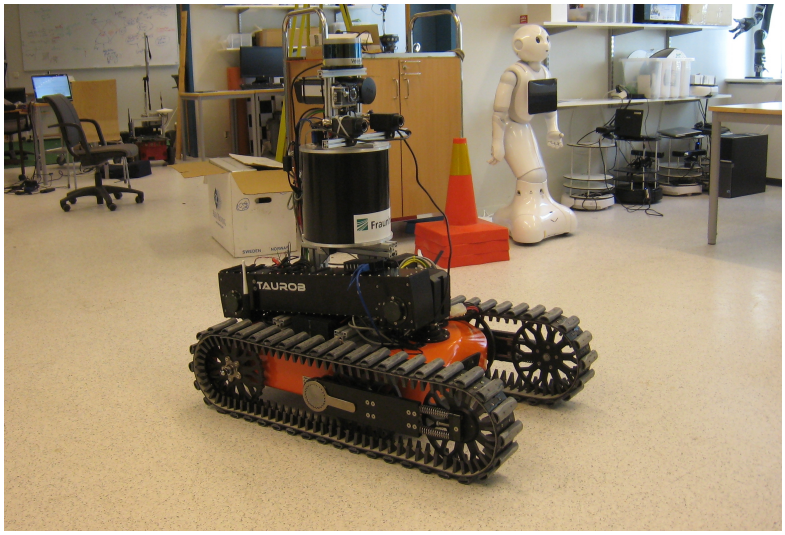
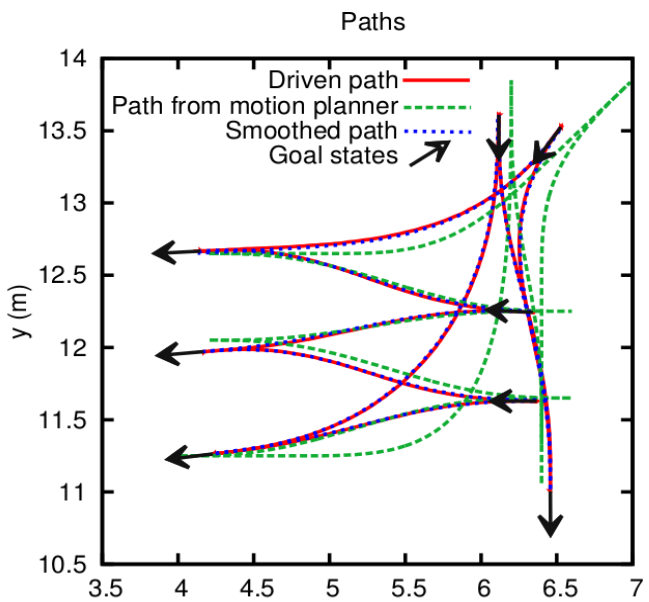


Fig. 1: The ORU version of the Taurob Tracker platform used in the Smokebot project.

This document is a technical report within the scope of the Horizon 2020 project SmokeBot. In search and rescue scenarios, a robot needs to perform navigation in unknown environments. The SmokeBot project focuses on civil robots supporting fire brigades and aims to enhance the performance of robots under harsh conditions, for example: smoke, dust or fog. This report summarizes the results of Task 5.3, "Save Navigation / Hazard Avoidance". Task 5.3 is part of Work Package 5, "Autonomous Behavior", which aims at reducing the mental workload for the operator by implementing sensor oriented planning modes to complement teleoperation. The particular objective of Task 5.3 is to implement a safe autonomous navigation module for the Taurob Tracker platform shown in Fig. 1. The system needs to function under adverse conditions which allow the operator to use high-level commands (e. g. go to the door on the other side of the room). This report is in part based on previous publications of the autonomous navigation pipeline developed at ORU [1, 2], and in particular on the most recent work described in [3].

Autonomous navigation in real-world environments is a challenging task in many respects. One of the key open challenges is fast planning and execution of trajectories to reach arbitrary target positions and orientations with high accuracy and precision. This is especially challenging if non-holonomic vehicle constraints are taken into account, as it is necessary for the continous-track vehicle used in Smokebot. The kinematic structure of the Taurob Tracker corresponds to a differential drive and thus the vehicle cannot instantaneously move in arbitrary directions. Furthermore, the non-holonomic nature means that even in cases when an error in the final pose can be reliably detected, it is not possible for the tracking controller to correct it over a short distance. Correspondingly, state-of-the-art motion planners (based either on Rapidly Exploring Random Trees (RRT), Probabilistic Roadmaps (PRM) or state space lattice search) have so far not been demonstrated to generate trajectories that enable autonomous vehicles to achieve the required goal pose accuracy for industrial applications. In addition, the trajectories obtained by current planners are not guaranteed to be directly drivable — discontinuities in vehicle controls and within the trajectories are often left to be handled by the

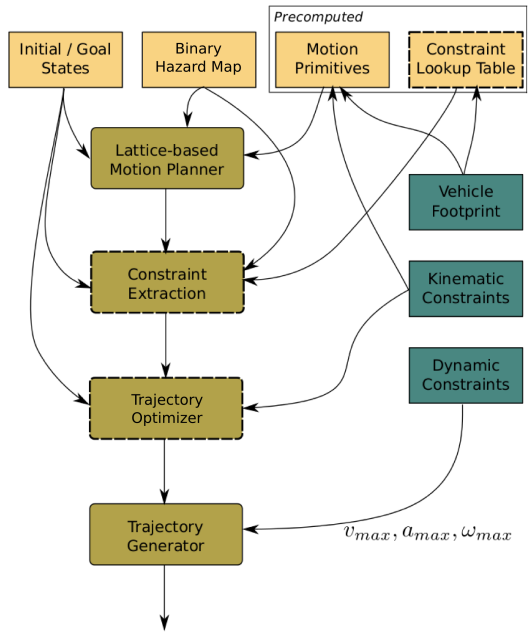
Fig. 2: An exemplary motion planning problem where a non-holonomic vehicle has to sequentially drive to a number of given goal poses (black arrows). The figure shows the paths obtained from a lattice-based motion planner (note that these paths contains additional backward/forward motions and the goal is not always where the direction changes). Also shown are the corresponding smooth paths obtained from the trajectory optimization step as well as the actual trajectories driven by the vehicle. The illustration is



controller. This is a well-known problem which has recently received attention [4, 5], but no solution has yet been proposed which can both guarantee the required accuracy and be used online.

At ORU, a trajectory optimization-based approach which leverages the output of a lattice-based motion planner as an initial solution guess has been pioneered since several years. Here, smooth collision-free trajectories are generated which adhere to the constraints imposed by the vehicle kinematics while optimizing a desired performance criterion. Achieving high end pose accuracy depends on all navigation subsystems. Therefore, we developed a complete navigation system composed of a lattice-based motion planner, a trajectory optimizer and a model predictive tracking controller. The system does not require a transformation of the input path to a new representation, which is smooth by definition (e. g., splines). Thus, it can directly incorporate state-space constraints in the optimization process. The resulting navigation pipeline generates smooth drivable trajectories for arbitrary goal poses in a few seconds, a fraction of the runtime reported by previous approaches [4]. In previously published work [3] we showed that our system can generate and execute smooth paths (see example in Fig. 2) with highly accurate end poses.

Fig. 3: Overview of the processing steps detailed in the subsequent sections. Sets of position and angular inequality constraints are extracted using precomputed lookup tables using the vehicle footprint. Motion primitives of a vehicle are generated off-line taking into account its kinematic constraints.



2. Related Work

Over the years, many different techniques for automatic path and trajectory generation have been proposed. Classic combinatorial methods are not very well suited in the presence of differential constraints (e.g., kinematic constraints for non-holonomic vehicles) and analytical solutions cannot effectively cope with obstacles [6]. To overcome these problems, sampling-based approaches have been introduced and studied in recent years. In particular, three families of methods are currently widely used: Probabilistic Roadmaps (PRMs) [7], Rapidly-exploring Random Trees (RRTs) [8, 9, 10] and lattice-based motion planners [11, 12, 13]. All sampling-based approaches have been shown to be effective in high-dimensional configuration spaces. Lattice-based motion planners, in particular, combine the strengths of the approaches discussed above with well-studied classical AI graph exploration algorithms, such as A*, ARA* and D* Lite [14]. Differential constraints are incorporated in the search space by means of pre-computed motion primitives, which sample the state space on a regular lattice. The search space is then explored using efficient graph search techniques. However, all sampling-based approaches to non-holonomic motion planning generate paths and trajectories that typically present discretization errors and discontinuities, within the trajectory itself or between the terminal and goal state. This is a known problem which prevents current solutions to achieve the end pose accuracy required by many application scenarios. In recent years, several solutions have been suggested for post-processing the trajectories from sampling-based motion planners [4, 5]. So far, the methods suggested are computationally too expensive to be used online, as smoothing requires time in the order of hundreds or even thousands of seconds [4].

2.1 Chosen Approach

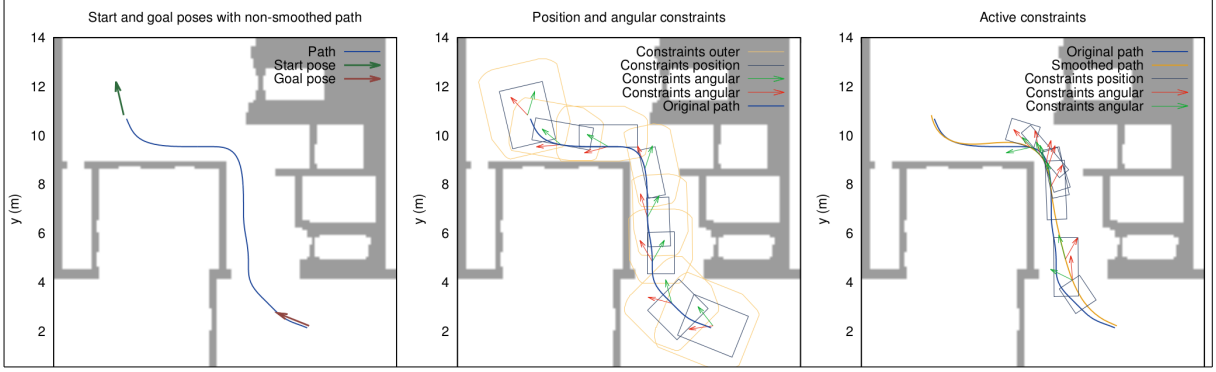
By contrast, our approach is designed to be used online as demonstrated in [3]. Although the Taurob tracker has a differential-drive, it is modelled as a car-like vehicle. This choice avoids turning on the spot, by enforcing a maximum steering angle of the virtual steering wheel. Turning on the spot is an unfavourable movement as it accelerates track wear and, in extreme cases, can lead to track loss. The control space $u = (v, \omega)$ is composed of forward speed v and angular velocity ω. The state space $s = (x, y, \theta, \varphi)$ consists of all vehicle configurations composed of 2D position $[x, y]$ of its reference point, heading θ and virtual steering angle φ. The corresponding state transition $ds/dt=f(s,u)$ can then be obtained from the kinematic model. Given initial and goal states, we strive to calculate a set of N control points together with the corresponding state points which take the vehicle from start to goal pose without colliding with any known obstacle. A solution to the problem is valid, if also additional vehicle-dependent constraints are respected such as the maximum allowed steering angle, the maximum steering velocity, the maximum forward and reverse velocity and the maximum acceleration.

For our system we adopt the four-step approach shown in Fig. 3. First, we calculate coarse, kinematically drivable paths with a lattice-based motion planner. Next, we extract an envelope of free and safe space around the planned path, represented in the form of convex polyhedral constraints on the vehicle state. Subsequently, the path is given as an initial guess to a trajectory optimizer, which also takes as inputs the extracted constraints, a kinematic model of the vehicle and a performance criterion. The result is an optimized, collision-free, continuous and drivable path. A final trajectory generation module can impose additional constraints on the maximum linear/angular velocities and linear acceleration. In the future, we plan to additionally feed back information provided by the hazard maps. The potential of using a trajectory optimization-based approach is further highlighted by the fact that this paradigm is also used in recent works for aerial AGV motion planning and execution [15, 16] as well as planning of walking gaits and manipulation movements [17, 18].

3. Motion Planning

The first step of our approach is a lattice-based motion planner [13], which quickly computes coarse initial paths. Given a model of vehicle manoeuvrability, the intuition behind lattice-based motion planning is to sample the state space in a regular fashion and to constrain the motions of the vehicle to a lattice graph. Each vertex of the lattice represents a valid pose of the vehicle $s = (x, y, \theta, \varphi)$, while each edge encodes a motion which respects the non-holonomic constraints of the vehicle. The lattice is constructed over a grid whose cells have a fixed side length, while θ and φ at each lattice vertex are constrained to pre-define finite sets of headings and steering angles, respectively. The planner uses a set of pre-computed, kinematically feasible motion primitives, which are repeatedly applied to build the lattice. Information about the static obstacles in the environment is provided to the planner by an available hazard map, which currently is represented as a binary occupancy grid. This map is used to prune the search graph to obtain collision-free paths. The motion primitives are automatically generated to fully capture the mobility of the vehicle and then reduced for efficiency purposes, as described in [19]. The lattice graph is then explored using ARA* [20]. Effective heuristic functions [21], as well as pre-computed vehicle footprints for each motion primitive are employed to speed up the exploration of the lattice. At this stage, the planner generates a path but no controls. The obtained path is subject to an error on the order of the lattice discretization.

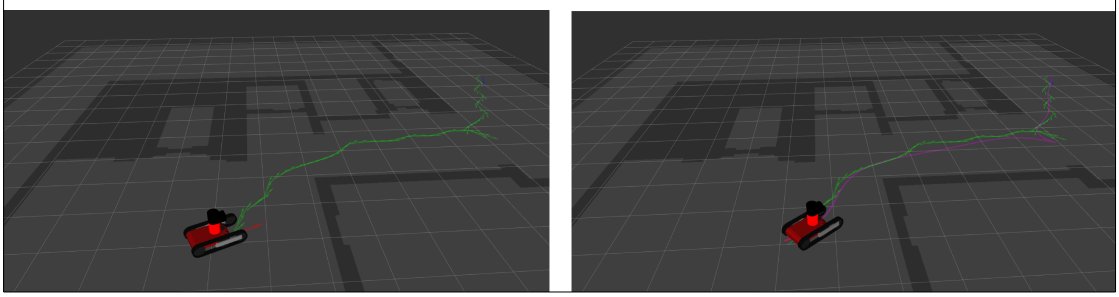
Fig. 4: Visualization of the trajectory optimization. (Left) Given start and goal poses, the lattice-based motion planner calculates a coarse initial path. (Centre) Spatial constraint extraction. Position constraints (blue boxes), angular constraints (red and green arrows) and their spatial combination (yellow line) are calculated for each position node in the initial path. For the sake of clarity, the figure represents only a sub-set of state points. (Right) The smooth path obtained from the optimization and the corresponding active constraints.



We then utilize the collision-free path provided by the motion planner and associate an envelope of space around it in which path nodes can be safely moved and thus provide redundancy for the subsequent trajectory optimization step. Naturally, “larger” spatial constraints on the vehicles position and orientation will allow greater flexibility into the smoothing step and more likely a better path. Using this method, we perform a one-time pre-computation of a potentially large number of constraint sets. These sets can then be used on-line, avoiding the complex and time consuming collision checks required by other approaches [22], [23]. A constraint set is considered collision free if it does not contain any occupied cell in the aligned occupancy map. Furthermore, a constraint set defines an area which can be obtained by sweeping the footprint of the vehicle over all the positions/orientations such that the constraints are satisfied. The selection of appropriate constraints is not trivial: given the free area around a specific state, assigning larger bounds to one of the two sets of constraints limits the values which can be assigned to the other set. Most importantly, the selection of appropriate constraints must be fast. To quickly identify an adequate set of constraints when a new path is generated by the planner, we us a set of heuristics as detailed in [3].

As mentioned above, the initial path obtained by the lattice-based planner is discretized and thus the start and goal states used by the planner not necessarily correspond to the given initial and end state (see Fig. 1). Therefore, we refine the initial path using trajectory optimization as illustrated in Fig. 4. The key to efficiently solving the resulting non-convex optimization problem is that an uninformed search is not necessary. From the lattice-based motion planner we already have a cost-optimal, obstacle-free path which we can use to initialize the decision variables of the optimization problem. We only focus the optimization on generating a path and not on obtaining the fastest trajectory. This is done in order to reduce the complexity of the problem. The goal is to minimize the total distance travelled and the total amount of turning applied on the virtual steering wheel. We formulate the objective as quadratic function of the control values. Ideally, only the side constraints should be activated during the optimization step. Forward and backward bounds are essential to guarantee collision free paths, but should preferably not be active apart from when the vehicle changes directions. The approach was formulated and solved using the ACADO Toolkit [24], applying a direct multiple shooting method.

Fig. 5: Smokebot navigation in the ROS ecosystem: (Left) A coarse initial path is computed using the lattice-based planner. (Right) The initial path is refined using trajectory optimization.



4. Control

In our vehicle navigation system, we use a Model Predictive Controller (MPC) [1] to track the paths obtained from the motion planning module as outlined above. The core idea of this type of controllers is to model how the states of the vehicle evolve over a time preview window, given a set of control inputs. The controller then optimizes the control output using an objective function based on the trajectory to follow. In our implementation we use a preview window of 25 control steps and each control step has a duration of 40 ms. The controller gains were set to equally weight heading and distance.

5. Implementation

For Smokebot, we developed a ROS-based interface which allows an easy switch between simulation mode using Gazebo and the actual platform. We implemented to operation modes: (a) semi-autonomous teleoperation, where an operator can indicate a desired target pose (e. g., a door or an elevator) by simply clicking on a map visualized in Rviz as illustrated in Fig. 5. The robot will then navigate to the specified goal using the pipeline described above. (b) autonomous navigation to a pre-assigned safe home pose, to which the robot will navigate autonomously in case of signal loss. We use occupancy grid maps in the navigation setup. These maps can either be pre-loaded or acquired using SLAM based on the on-board sensors of the robot which include sensory information from an on-board radar which is used in adverse conditions.

6. Discussion

During early test runs, the setup described above performed well and was able to consistently navigate to the assigned target poses with sufficient accuracy. A complete numerical benchmark of the system has been previously carried out and is reported in [1, 2]. Localization of the robot is currently based on odometry information only, as the fused map SLAM algorithm proved to be too slow to provide pose information at a frequency required by the tracking controller (25Hz). Therefore, the main goal for

future work is to fuse the robot odometry with the absolute pose information from the SLAM system to obtain accurate localization at a sufficient update rate.

Bibliography

[1] F. Pecora, M. Cirillo and D. Dimitrov, "On mission-dependent coordination of multiple vehicles under spatial and temporal constraints," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5262-5269.

[2] H. Andreasson *et al*., "Autonomous Transport Vehicles: Where We Are and What Is Missing," in *IEEE Robotics & Automation Magazine*, vol. 22, no. 1, pp. 64-75, March 2015.

[3] H. Andreasson, J. Saarinen, M. Cirillo, T. Stoyanov and A. J. Lilienthal, "Fast, continuous state path smoothing to improve navigation accuracy," *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 662-669.

[4] P. Cheng, E. Frazzoli and S. LaValle, "Improving the Performance of Sampling-Based Motion Planning With Symmetry-Based Gap Reduction," in *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 488-494, 2008.

[5] K. M. Seiler, S. P. Singh, S. Sukkarieh, and H. Durrant-Whyte, “Using lie group symmetries for fast corrective motion planning,” *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 151 - 166, 2012.

[6] J. Marshall, T. Barfoot, and J. Larsson, “Autonomous underground tramming for center-articulated vehicles,” *Journal of Field Robotics*, vol. 25, no. 6-7, pp. 400–421, 2008.

[7] L. Kavraki, P. Svestka, J. Latombe, and M. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *IEEE Trans. on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[8] S. M. LaValle, “Rapidly-exploring random trees: A new tool for path planning,” TR 98-11, Computer Science Dept., Iowa State University, 1998.

[9] S. Karaman and E. Frazzoli, “Sampling-based algorithms for optimal motion planning,” *The Int. Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[10] F. Islam, J. Nasir, U. Malik, Y. Ayaz, and O. Hasan, “RRT*-smart: Rapid convergence implementation of RRT* towards optimal solution,” in *Proc. of the Int. Conf, on Mechatronics and Automation (ICMA)*, 2012.

[11] D. Ferguson and M. Likhachev, “Efficiently using cost maps for planning complex maneuvers,” *Proc. of the ICRA Workshop on Planning with Cost Maps*, 2008.

[12] M. Pivtoraiko and A. Kelly, “Fast and feasible deliberative motion planner for dynamic environments,” *Proc. of the ICRA Workshop on Safe Navigation in Open and Dynamic Environments: Application to Autonomous Vehicles*, 2009.

[13] M. Cirillo, T. Uras, and S. Koenig, “A lattice-based approach to multi-robot motion planning for non-holonomic vehicles,” *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*

(IROS), 2014.

[14] S. Koenig and M. Likhachev, “D* lite,” *Proc. of the National Conf. on Artificial Intelligence (AAAI)*, 2002.

[15] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart and E. Galceran, "Continuous-time trajectory optimization for online UAV replanning," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 5332-5339.

[16] M. Nieuwenhuisen and S. Behnke, "Local multiresolution trajectory optimization for micro aerial vehicles employing continuous curvature transitions," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 3219-3224.

[17] Zucker, M., Ratliff, N., Dragan, A.D., Pivtoraiko, M., Klingensmith, M., Dellin, C.M., Bagnell, J.A. and Srinivasa, S.S., 2013. CHOMP: Covariant Hamiltonian optimization for motion planning. *The International Journal of Robotics Research*, vol. 32(9-10), pp. 1164-1193, 2013.

[18] *N. Ratliff, M. Toussaint and S. Schaal, "Understanding the geometry of workspace obstacles in Motion Optimization," IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 4202-4209.*

[19] M. Pivtoraiko and A. Kelly, “Kinodynamic motion planning with state lattice motion primitives,” *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2011.

[20] M. Likhachev, G. Gordon, and S. Thrun, “ARA*: Anytime A* with provable bounds on sub-optimality,” *Advances in Neural Information Processing Systems*, vol. 16, 2003.

[21] R. A. Knepper and A. Kelly, “High performance state lattice planning using heuristic look-up tables,” *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2006.

[22] B. Lau, C. Sprunk, and W. Burgard, “Kinodynamic motion planning for mobile robots using splines,” *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2009.

[23] C. Sprunk, B. Lau, P. Pfaff, and W. Burgard, “Online generation of kinodynamic trajectories for non-circular omnidirectional robots,” *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2011.

[24] B. Houska, H. Ferreau, and M. Diehl, “ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization,” *Optimal Control Applications and Methods*, vol. 32, no. 3, pp. 298 - 312, 2011.