VIENNA 2018 TRA
TRANSPORT RESEARCH ARENA
A digital era for transport
solutions for society, economy and environment

Proceedings of 7th Transport Research Arena TRA 2018, April 16-19, 2018, Vienna, Austria

TransportBuddy: Navigation in Human Accessible Spaces

Markus Bader, George Todoran, Florian Beck, Benjamin Binder and Klaus Buchegger

Vienna University of Technology, Karlsplatz 13, Vienna 1040, Austria

Abstract

This work presents advancements achieved in the field of autonomous service robotics in open accessible spaces, with preliminary project results. A novel navigation algorithm is presented. It integrates (i) a new trajectory planner which can be proven to be safe with respect to expected moving objects by making use of emergency trajectories, (ii) person motion models trained from historical data and integrated into the trajectory planner to mimic human-like behaviors, and (iii) a global planner which is able to coordinate multiple vehicles in tight spaces without handcrafted tracks to prevent deadlocks. The results presented are shown on real and simulated hardware which was developed and used in the projects TransitBuddy, TransportBuddy and Autonomous Fleet.

Keywords: Autonomous navigation, human space, local planner, global planner, trajectory planner, robotics

1. Introduction

In contrast to Automated Guided Vehicle (AGV) systems in industry or logistics environments, nearly all available autonomous transportation systems in public spaces rely on bringing additional benefits next to the transportation of goods in order to be cost-efficient. An increased PR and prestige are such additional benefits. These benefits and a variety of new applications help overcoming one's inhibitions to use an autonomous transportation system. However, on the long term, the transportation of goods has to be the main source of income, making the efficiency of the robot's navigation in public spaces essential and thus of interest for the research community.
Therefore, a system composed of autonomous vehicles has to outperform a classical AGV system. TransitBuddy, TransportBuddy and Autonomous Fleet are Austrian research projects with the aim of designing a vehicle for public spaces and coordinating multiple such autonomous vehicles. The development of new control technologies for navigation and environmental sensing, as well as user studies, design studies and tests in real world environments such as train stations, shopping malls and hospitals are part of these projects. This paper presents a navigation system based on a multi layered control system similar to ROS (Quigley et al. 2009) move_base1. The local planner is based on a Model Predictive Controller (MPC), which optimizes its trajectory based on multiple cost maps, cost function terms and explicit constraints, resulting into a time-space optimization at an update rate of 10-20Hz (Todoran 2017). People motion models (Kollmitz et al. 2015) are taken into account in the optimization, resulting in trajectories that slow down the vehicle in order to let people or other vehicles pass first, without path derivation, as shown in Fig. 1.
To these ends, a person-tracking system was implemented. Statistical knowledge gained by using person flow models allows the system to use priors on person motions and to overcome the limitations of classic people-tracking algorithms.
Overall multi-robot planning is based on route segments generated by a Voronoi distillation (Wang et al. 2016), taking the distance to static map entries into account to compute routing tables.
The new system's gain in performance is compared to a classic AGV system and ROS move_base using real data, and to simulated data using GazeboSim for physics simulation.

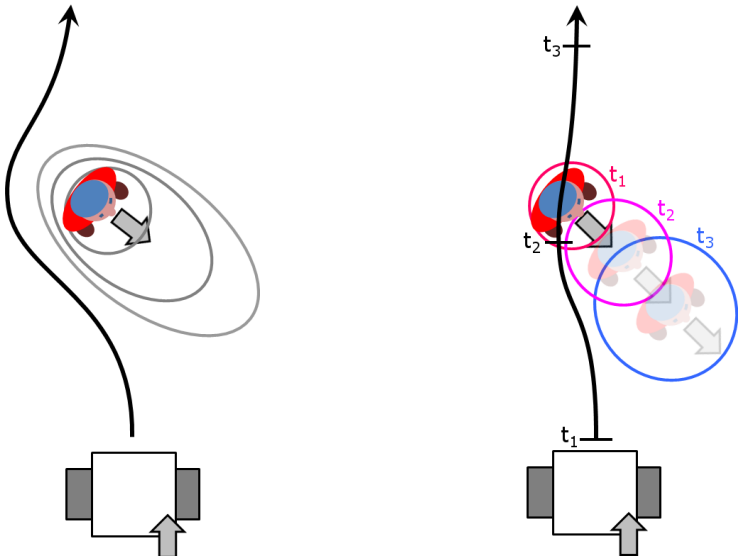


Fig. 1: Robot trajectories which take human motion into account. On the left a classical approach is shown. In this approach, a cost field at the person's location covers the person's predicted motion. The image on the right displays the system implemented, which takes time into account as well as motion uncertainty as it passes through the person's perceived location at time t_0 by predicting free space at time t_2.

This paper first presents related work in Section 2, followed by a brief description of the setup used and the approach applied in Section 4. A Results and a Conclusion section close the paper.

2. Related Work

Systems like ROS move_base are widely used in countless applications and research projects such as Hobbit (Fischinger et al. 2014), Squirrel. move_base and similar frameworks such as the reactive planner in MRPT (Blanco et al. 2008) approach the navigation problem by using two layers of planning. A global planning layer uses a graph search algorithm on a given map to find a path between the robot's current location and goal location, and a local planning layer computes control commands for the robot to follow the global path. The local planner, also denoted as the trajectory generator, takes robot kinematics into account and reacts if a detected obstacle is blocking the path. An automated vehicle which is not autonomous typically uses a PID-Controller or a flat output controller (Lies et al. 1995) as its local planner, enabling it to simply follow the global path; it stops if an obstacle obstructs it. Safety is assured by the use of certified sensing hardware such as SICK laser scanners[1]. These sensors are able to activate the vehicle's braking system as needed. Such a simple technique is typically implemented in almost all AGV systems (Ullrich 2014) in combination with a global planner which computes the global path for multiple vehicles based on predefined, often handcrafted path-segments (lines and arcs) without using a 'real' map of the environment. This simplifies vehicle self-localization, because vehicles just follow predefined segments and the vehicles' poses – position and orientation – can be identified by the current active segment to follow. This approach is applied in nearly all warehouses, including Amazon (Wurman et al. 2007).
A more complex local/on-board planning layer is needed for an autonomous vehicle (Todoran and Bader 2016). In his work, Andreasson et al. (2015) documents the need for a system which is able to work with autonomous vehicles. Map-based self-localization as well as a vehicle's state prediction on the planning and trajectory layer are important, according to Andreasson, for integration of a robust system. A similar approach with a three-layered planning is presented in Bader et al. (2015), aimed at design of a system with scalable vehicle autonomy for increased human acceptance.
Human awareness is of significant importance to autonomous vehicles in populated environments. Typically, humans are considered via some sort of cost function in the planning algorithm. In Kollmitz et al. (2015) the authors propose utilizing social cost fields based on a constant velocity prediction of human movement which can be integrated into an A* planning algorithm. Another approach is taken in Bennewitz et al. (2005), where a human motion model, a Hidden Markov Model, is learned from observed trajectories using expectation maximization. Those models are used in order to determine the probability of a person occupying a cell, which can again be integrated into A* planning.

[1] Detection and Ranging Solutions, SICK AG, Erwin-Sick-Str. 1, 79183 Waldkirch, Germany, December 2011, 8014402/2011- 12-20. Available: http://www.sick.com

Multi-layered architecture as well as a human motion model supported by a person-tracking system are described in the work presented.

3. Setup

The goal of the aforementioned research projects is the ability to drive vehicles in environments populated by humans. To these ends, a setup with a pioneer robot, shown in Fig. 2, was used to test the navigation system. Hardware costs, maintenance issues and time were reasons to test the multi- robot's behavior as well as parts of the local trajectory generator in a simulated environment. Therefore, GazeboSim[2] was utilized to test human motion models and the local planner. Because Gazebo is able to simulate kinematics and the related physics precisely enough to allow for a smooth migration of the algorithms tested onto real hardware afterwards. Stage Robot Simulator[3] was used as a simulation environment for the multi-robot planning component of the project because it uses a simplified but rapid physics simulation which allows for simple simulation of over ten robots at once, faster than real-time. An experimental sensor similar to AirSkin is mounted on the robot to detect physical contact, but due to this paper's constraints, the resulting possibilities for control of the vehicle are not discussed here. However, integration of the AirSkin sensor is a fundamental part of the research project TransportBuddy.

Fig 2: The real and simulated Pioneer 3DX robot used for experiments. 1 - emergency stop button, 2,3 - depth cameras, 4 – SICK Laser scanner, 5 - Air-Skin, 6 – Hokuyo laser scanner

4. Approach

Following the approach of Bader et al. (2015), the system presented integrates a centralized planning/coordination system based on a modified Prioritized Planner (Kleiner et al. 2015) enabling multiple vehicles to deal with deadlocks, and a local planner which uses various non-trivial cost-maps as well as a human motion model to evaluate potential trajectories. The multi-robot planner implemented, the trajectory generation as well as the human motion model implemented are discussed next, followed by the results of the experiment.

5. Mutli-robot planner

Since the local planner is based on an MPC like the one proposed in Todoran and Bader (2016), there is no need to compute trajectories on the global planning level. The MPC optimizes locally (on board) the trajectory of the vehicle by taking the current sensor reading into account. Therefore, neither obstacles nor vehicle dynamics are taken into account by the global planner. The global planner uses a given map of the environment as input. These maps are typically pixel maps and need to be pre-processed and simplified. A well-known method for simplifying maps is to build road maps (LaValle 2006). Voronoi maps (Wang et al. 2016) can be utilized to generate such road maps. Voronoi maps can be used to create a fully connected path with line-of-sight contact to every pixel on the map. Furthermore, these lines describe the path with the maximum distance to obstacles at every point. In a further step, these Voronoi-maps are used to generate graphs, as shown in Fig. 3. Each graph edge represents a path segment to be utilized by the global multi-robot path planner (GMRPP) (Binder 2018). As described, the GMRPP does not take any system dynamics into account.

The GMRPP should generate guaranteed deadlock-free paths for multiple robots. To achieve this, different planning strategies are used. LaValle (2006) identified three types of methods used to address such planning problems:

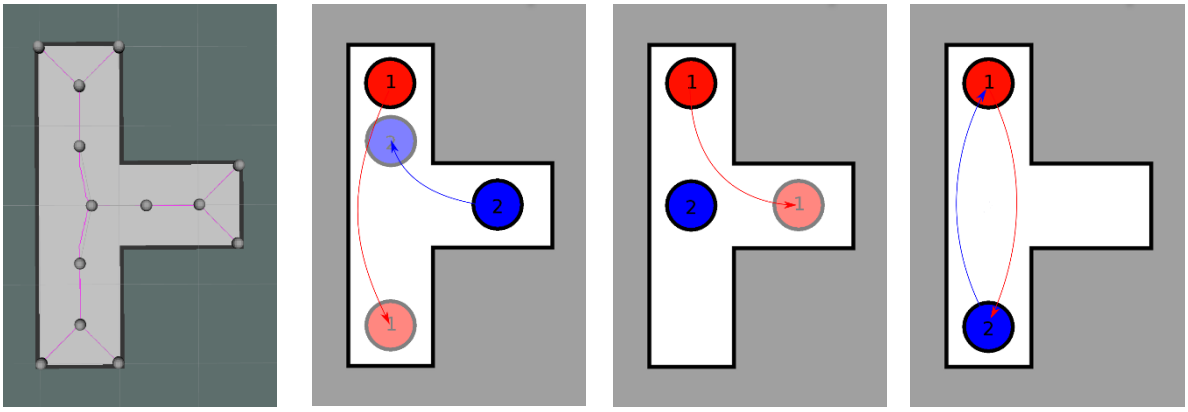
1. Centralized Methods
2. Decentralized Methods
3. Decoupled Methods.

The centralized method describes one in which all vehicles are described as one system with multiple degrees of freedom. This method has the advantage of completeness and optimality but is computationally intensive. The second and third methods split the multi-robot planning problem into multiple single-robot planning problems in

[2] GazeboSim: A 3D physics simulation http://gazebosim.org/

[3] Stage Robot Simulator: Simple and fast multi robot simulator, http://playerstage.sourceforge.net/doc/stage-svn/

order to decrease complexity, but sacrifice completeness and optimality. The decentralized method approaches the problem by attributing every robot with a priority and allowing them plan their own trajectories yielding to robots with higher priority. The decoupled method plans every robot's trajectory on its own and resolves conflicts while the plan is being executed.



a) **b)** **c)** **d)**

Fig. 3: Multi-robot planner. a) Environment with a computed Voronoi-graph (magenta) and the segmentation of the path marked with grey dots. b) A test case where the robots start in the dark-blue and dark-red spots and travel to the lighter ones. c) A test case where the red robot wants to travel to its destination (light red) and the blue one wants stay in its position. d) A test case where both robots want to switch positions.

However, using decoupled methods can produce deadlocks or at least longer paths than the other two methods. The first method is complete, but is not really ideal for a large number of robots. Thus, the GMRPP uses the prioritized-planning approach described by LaValle (2006). Since there are instances where the basic prioritized-planning approach fails, a new planner is proposed here to solve problems, in which relative execution time among robots is important. Furthermore, a revised prioritization strategy, based on the work of Bennewitz et al. (2001), and a speed scheduling strategy are used to solve even more problem sets.

In our approach, a Timed Path Coordinator is introduced. This coordinator is aware of all preplanned vehicle movement, including path and time. When a new path is being planned, the coordinator is asked, whether the planner may occupy a segment at a specific point in time. This is done in order to synchronize existing paths with the new one. This helps the planner to solve problems like two robots moving sequentially.

In environments with narrow corridors, there could be instances in which one robot has to wait for others in order to avoid deadlocks. Such an instance is shown in Fig. 3b, where the red vehicle wants to move from the top to the bottom and the blue one wants to move from the right segment to the top. To resolve this, the blue vehicle has to wait for the red one. This issue can be resolved by using a backtracking strategy, meaning that if the path coordinator rejects an expansion request due to an impending robot collision, the planner is allowed to backtrack to any crossing on its preplanned path and wait at a segment of the crossing. This would enable a lower-prioritized robot to avoid a more highly-prioritized one at a crossing.

Another problem arises if a robot is parked at a pose which blocks another robot's path. To avoid such instances, more highly-prioritized vehicles are allowed to push lower ones away. This behavior is managed by the GMRPP. Such an instance is shown in Fig. 3, where the red robot pushes the blue robot away from its position in order to pass through the crossing.

As stated by Bennewitz et al. (2001), there are problem cases in prioritized planning in which specific robot priority schemes can cause the planner to fail, whereas other priority schemes ensure the planner’s success. This paper explains how to solve this: by randomly exchanging priorities. For the GMRPP proposed, this strategy is also used and revised in the following way: priorities are only exchanged if the path coordinator has detected at least one potential collision between them. Vehicles with multiple collisions have a higher probability of being selected for priority exchange.

Fig. 3d represents a specific problem case, in which the red and the blue robots want to switch positions. Obviously one of them has to wait at the spot, at the top of the map to let the other one pass. Both robots have the same speed and neither of them is fast enough to beat the other to their common destination. Therefore, speed scheduling is introduced: it decreases the speed of preplanned robot paths if a robot fails to determine a plan and lets it retry its planning iteration. In the example shown in Fig. 3, it would work as follows: the red robot plans its trajectory and the blue one is fails to plan its trajectory. The priority scheduler detects this imminent collision and halves the red robot’s speed, giving the blue robot enough time to move to the spot above the red robot’s planned path.

The combination of these strategies enables the path planner to find solutions to scenarios, where standard prioritized planners fail (LaValle 2006) (Kleiner et al. 2015).
The resulting plan for each vehicle is sent to the trajectory generator.

5.1. Trajectory generation

Given that the working environment of the agent is assumed in our approach to be unstructured and dynamic, we believe that the local path-planning layer should possess increased capabilities. More specifically, we target:

1. Qualitative (optimal) resulting trajectories while systematically taking into account constraints
2. Reactive behavior, relying more on online perception and less on a-priori representation (static maps)
3. Formal guarantees of non-collision given certain environment and perception models
4. Non-existence of deadlocks (local minima) within a large vicinity of the agent (approx. 10 m)
5. Reduced computational complexity, targeting cycle-times of 30-100 ms,

The typical approach used in local path-planning relates to the Dynamic Window Approach (DWA) (Fox et al. 1997). In this approach, trajectory candidates within a future horizon are sampled in the input-space of a motion. Based on various cost-functions defined, the best trajectory sampled is selected and its first input is applied to the vehicle. This process is evaluated iteratively at frequencies of 10Hz. Though simple to formulate and implement, such an approach has a major drawback: with increased evaluation horizons, the sampling of the trajectory space becomes very coarse, resulting in crude, sub-optimal trajectories.
In our approach, we do not make use of sampling but rather gradient-based non-linear optimization. We formulate trajectory generation as a dynamic non-linear optimization problem, discretized using a minimal-parametric representation. More specifically, the underlying differential equations (ODEs) of the problem formulated are not encoded implicitly in the optimization problem solved but are rather solved externally using (explicit) ODE solvers. This results in considerable acceleration of solution computation while maintaining increased accuracy of the trajectory planned.
It is well known that for moving-horizon controllers (such as Model Predictive Control, MPC), although a solution exists for iteration k, it is in general not guaranteed for there to be a solution at iteration $k+1$ (Schouwenaars et al. 2002). This statement is valid even in simple instances in which system models are exact. We approach this problem in a systematic fashion by formulating a set of safety constraints so that, given uncertainty boundaries of the environment state, the existence of an emergency trajectory that leads to a stop is always guaranteed. To this end, in contrast to other approaches, we take into account the intrinsic method of visual sensing: the line of sight. This means that the agent will locally consider regions of the map that are not in its current line-of-sight to be, worse-case, non-drivable, despite being nominally obstacle-free (based on the global map). Moreover, we make use of a formulation that addresses a typical problem encountered in robust MPC (Bemporad and Morari 1999) approaches when accounting for environment dynamics: the uncertainty of the environment state increases quickly during forward-prediction, constraining the planning horizon of the trajectory generator.
One of the fundamental goals of navigation is to arrive at a given destination without colliding with obstacles. Typically, finding a path from the location of the agent to its destination is a task that is solved by the global planner using graph-search algorithms such as Dijkstra or A*. Thus, the local planner is expected to follow the global planner's path closely, being able to perform small deviations from it. This results from the fact that within a local neighbourhood, finding real-time paths that do not yield local minima is a non-trivial task. In our approach, however, we want to grant the local planner a higher degree of autonomy, allowing it to deviate a considerable distance from the global path, given arbitrary, unstructured and unmapped obstacles' configurations (with deviations of up to 5 m). Two difficulties of this requirement deviation lie in the fact that typical approaches result in a multitude of local minima when deviations from the path are considerable and the drivable space is a non-convex set. To this end, we iteratively compute solutions to specific partial differential equations (PDE) in the local window of the agent. One such PDE that results in a local-minima free metric in arbitrary environment configurations is the Laplace equation, its computation resulting in a harmonic field. However, its computation is relatively slow, especially in instances where map scalability is desired (due to the gradient vanishing problem, the equation is typically solved in log-space (Wray et al. 2016)). An alternative PDE that provides the same local-minima-free characteristics which is at least an order of magnitude faster to compute, is the solution to the Eikonal equation (typically solved using fast marching methods (Gomez et al. 2015)). An additional benefit to using such a metric is the additional degree of freedom offered by velocity damping. To this end, the field computed can be distorted in the vicinity of obstacles, as illustrated in Fig. 4. Note that as the damping distance is increased, the metric evaluates the longer corridor with a smaller metric, resulting in a behaviour of navigating in wider (but possibly longer) areas. This is, in general, desired, as this typically allows the agent to navigate at increased velocities.
Given the general mathematical formulations and the implementation of our model-predictive navigation (MPN)

framework, the agent as well as environment assumptions and models can be systematically modified and improved. As one of the use cases targeted by the system presented is navigation in human-shared environments, it is natural to model their dynamics, thus improving agent navigation while providing pleasant agent-human interaction.

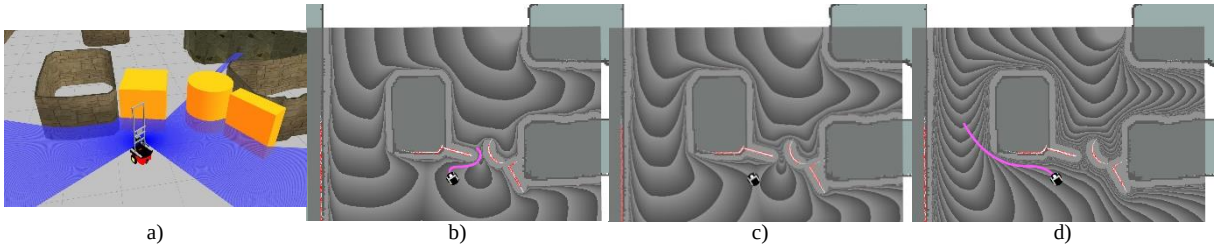


Fig. 4: a) Simulated scenario with agent in vicinity of unmapped obstacles. b) FMM field with velocity damping up to 0.5m away from obstacles; the optimized trajectory planned involves a narrow passage. c) FMM field with velocity damping up to 0.79m away from obstacles; the agent pose is located at a saddle-point of the field. The optimized trajectory can go either way. d) FMM field with velocity damping up to 2m away from obstacles; the optimized trajectory planned takes the wider and longer passage, being the shorter path under the distorted metric.

5.2. Human motion integration

In order to navigate safely around humans, it is important to keep track of their positions over time and predict their movements. Since human motion is stochastic in nature, we employ probabilistic filtering on the position and velocity of a person. Combining several filters (one for each person), together with detection data association, results in a people-tracker. The tracker not only enables awareness of the previous trajectory of a person, it can also integrate detections by several sensors, e.g. laser range finders and cameras. Classic approaches use linear Kalman filters, typically utilizing a (nearly) constant velocity model, to predict a human's motion and to track people detected. The approach implemented in this work does not rely on a linear model, since we apply instead particle filtering to keep track of people. For our motion model, we make use of historical data obtained from pedestrian simulations using a microscopic model from (Seer 2015). Locations of people detected are stored in a heat map, shown in Fig. 5. This heat map represents the likelihood of a person being in a specific location. Given the current estimate of a person's pose and velocity, we utilize the heat map to predict their pose in the next time step. It is assumed that people will move to locations where many other people have been observed before. Therefore, a person predicted to move towards the most likely-according to the heat map-location in the vicinity. To avoid the prediction spinning around a local maximum, the local area of interest can be limited to locations in front of the current estimated pose. This limitation also eliminates some non-intuitive predictions, such as sudden sharp turns of over 90°. In contrast to a constant velocity model, our approach also indirectly models obstacle avoidance for the prediction, as it mimics actual human walking patterns.

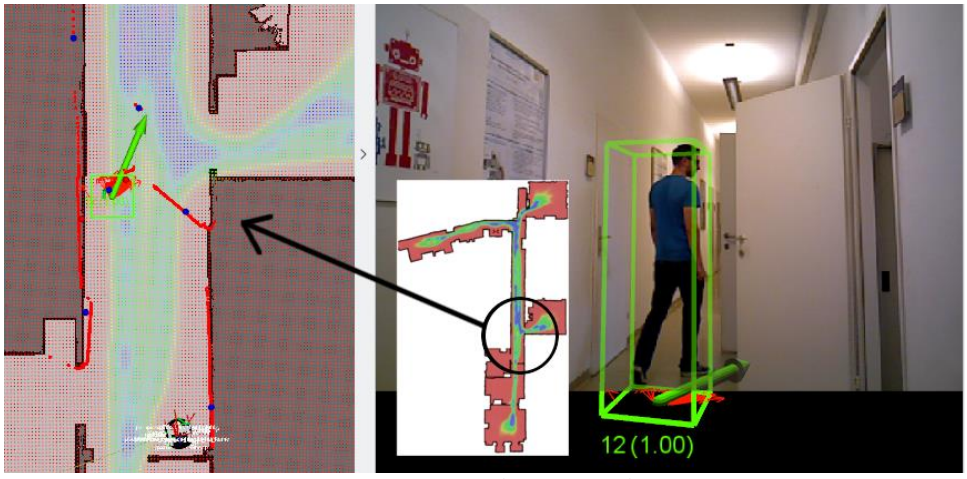


Fig. 5: People-tracking example in our lab using a heat map generated from a person-flow simulation.

In the tracking approach, the forward predictions are utilized as prior distribution. Several directions can be taken into account and weighted by their likelihood making the forward prediction more robust. Similar to the (nearly) constant velocity model, we model slight acceleration using small additive Gaussian noise. The particles drawn

from the prior are weighted by the probability of their explaining observations by the detectors. By applying low variance resampling, it is assured that particles with higher weights are replicated, narrowing down the overall variance of the tracker output. Naturally, particle filters allow for multi-modal pose hypotheses. This is an advantage, especially for occlusion handling. For example, consider a person moving behind an obstacle, making them invisible to a detector. A constant velocity model would simply move straight forward, according to the last measurement taken, meaning it would be able to move through walls, etc., completely neglecting its surroundings. In contrast, the heat map motion model enables prediction along commonly used paths, distributing particles across likely scenarios even during occlusion.

In addition to tracking, the heat map is also used by the local planner itself. For every person tracked, a trajectory for the planning horizon needs to be predicted, so that the robot can consider future person poses for the trajectory planned. Unlike for tracking, only the single most probable location from the heat map is used for predicting a change in a person's direction. A current estimated pose of the person and the target location are used to calculate the angular velocity up to the next point in time. This prediction strategy results in a single time-dependent trajectory for each person observed. The planner can then use the person-prediction trajectories to reject robot trajectories that at any point in time violate a safe distance from people. Additionally, social aspects (Kollmitz et al. 2015) can be used to prioritize trajectories that feel comfortable to the people involved, such as overtaking, or passing on the person's left. This way, trajectories that efficiently evade, overtake, or if necessary follow people at an adapted speed are selected by the planner.

An example scenario can be seen in Fig. 1. On the left, using the classic approach without temporal prediction, the trajectory planned only accounts for the person's current position. While such a trajectory would be fine for a static obstacle, it causes an unnecessary detour if the person is moving as indicated by the arrow. The trajectory planned shown on the right-hand side of Fig. 1 can be obtained using our prediction strategy. The planner knows where the person currently (t1) is and where they will be at future times (t2 and t3). Using this knowledge, the trajectory can be planned so that at time t2 it runs through the current position of the person, because the space will be free at that time. Other advantages of our approach are that if a person and the robot approach each other, the robot can pre-emptively move aside to make way for the person, or if the robot approaches a slow person from behind in a narrow hallway, it can adapt its speed to follow smoothly, instead of catching up, stopping and then accelerating again once the person has moved on a few steps.

6. Results

Since TransportBuddy and Autonomous Fleet are ongoing projects, the results presented here are preliminary. It was necessary to separately evaluate parts of the approaches presented using a simulator and the real hardware presented in Section 3. First, the performance of our local planner, named MPN, is presented with a comparison on our real robotic hardware. Second, the performance of the GMRPP is shown by using Stage Robot Simulator with two and three robots, followed by the human motion model, which was tested using GazeboSim with one robot.

In order to evaluate MPN performance we let a Pioneer 3-DX drive a route in our lab multiple times in different scenarios (see Fig.6), and we compared the run-time of the classic move_base node of ROS with our new MPN controller.

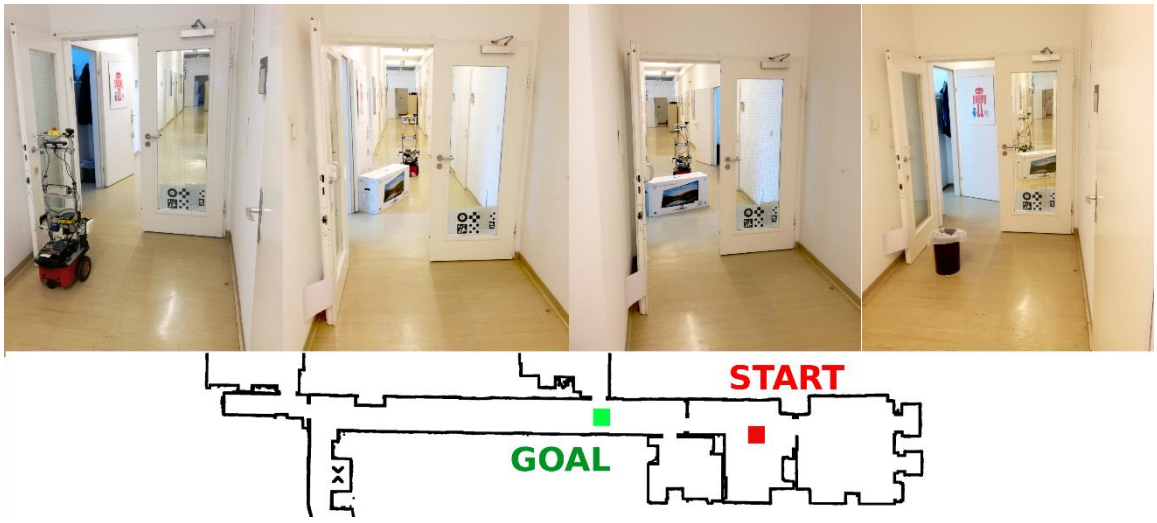


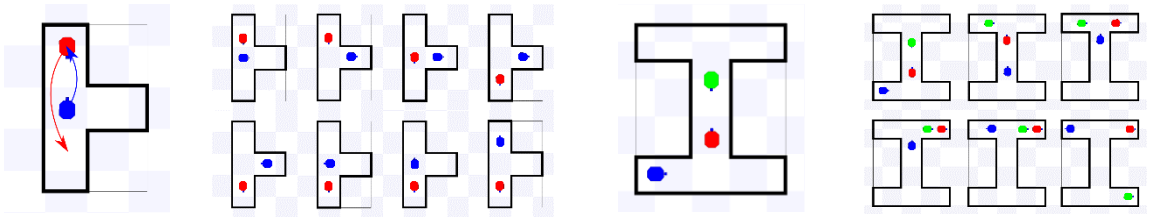
Fig. 6: Four test scenarios from, left to right: no obstacle, easy obstacle, hard obstacle, hard environment and corresponding map with start and destination (goal) positions.

Our MPN was, in contrast to move_base, always able to reach its destination. We granted ROS move_base at least four attempts per scenario to achieve meaningful run-time values. Table 1 shows, in contrast to move_base, only one time value per scenario for the MPN because the MPN repeatedly selected the same trajectories and finished with the same run time. As one can see in Table 1, our new MPN controller outperformed move_base in run time and also in success rate.

Table 1: Run time comparison of move_base to our new MPN controller in four scenarios.

	no obstacle [sec]	easy obstacle[sec]	hard obstacle[sec]	hard environment[sec]
move_base	29; 27; 33; 35	failed; 35; 33; 32	failed ...	failed ...
MPN	9	9.5	10	15

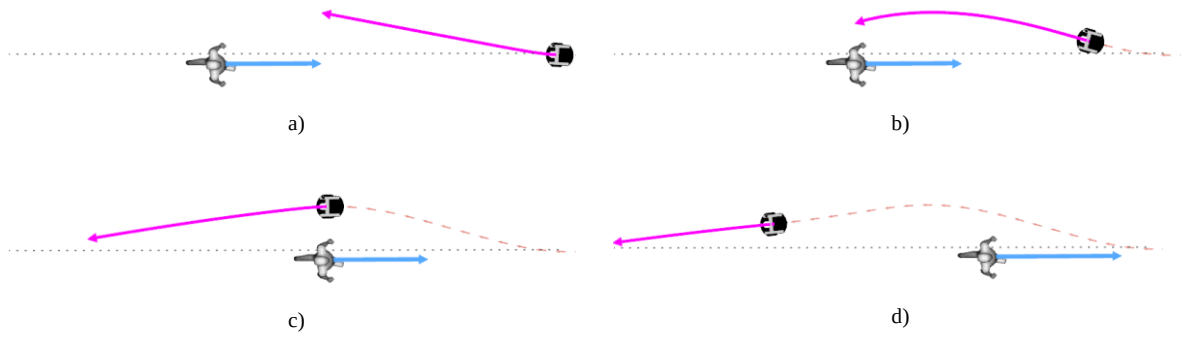
The GMRPP was tested, using Stage Robot Simulator, with multiple robots in various scenarios posing a planning problem. Since the GMRPP is event-triggered, it was sufficient to use a simple PID controller instead our MPN. Using real robots with the MPN proposed here would only change the execution time.
To evaluate the GMRPP, various scenarios with different maps and robots were tested. Two of them are depicted in Fig. 7. In the first scenario, shown in Fig. 7a, the red robot wants to move from the top to the bottom, and the blue one wants to move to the red robot's starting position. The solution to this test case is shown in Fig. 7b, which demonstrates the proper operation of the timed path coordinator and push strategy.



a) b) c) d)

Fig. 7: Two multi-robot test scenarios (a, c) and their solutions (b, d)..

The second scenario, shown in Fig. 7, depicts a test case where three robots are used. To solve this test case, backtracking and push strategies are used by the path coordinator multiple times, as shown in Fig.7d. First, the red robot pushes the green one to the top left; after the red one has passed the green one backtracks to its previous position and gets pushed to the top right by the blue robot. Finally, once the blue one has passed it, the green one can move to its destination on the bottom right.
The two test cases shown have been selected to give a rough idea of what is possible to solve with the GMRPP. Further test cases were created and tested to verify the correct functioning of the planner.



a) b) c) d)

Fig. 8: Test scenario with humans. The black dotted line shows the global plan, the magenta line shows the local trajectory, and the blue line shows the prediction of the person's path. The local trajectory is planned in order that the greatest distance to the global path occurs when robot and person meet in c), to satisfy safe distance requirements.

In our test scenario for navigation close to humans, see Fig. 8, in which a person and a robot approach each other. The person (the grey human figure) is walking from left to right, and the robot (the black, grey circle) is driving from right to left. The robot's planned global path is the thin, dotted horizontal black line, and the current local trajectory for the coming seconds is the magenta line, starting at the robot's center. The prediction of the person's walking path for the same time horizon is the blue line. The robot's pose history, as well, is indicated by small red arrows. The sequence of images Fig. 8a-d shows how the robot drives around the person approaching. In Fig. 8a, the robot plans a trajectory that deviates from the global path, as the end points of a planned straight trajectory would be too close to the end point of the person's predicted path. In Fig. 8b, one can seen that the greatest deviation of the local trajectory from the global path is not where the person currently is, but rather where the person and the robot will meet. Thus, in Fig. 8c the robot passes at a safe distance from the person. Once the robot has passed the person (Fig. 8d), it continues to drive straight towards its destination. In general, it is noticeable that the local trajectory is computed in real-time by our MPN framework, and does not make any sudden changes while the robot is avoiding the person. Without a prediction, the robot would only react to the person's current position, which could cause abrupt stops, direction changes, or even, worst case, a collision.

7. Conclusion

This work presented a new approach to navigation with an autonomous vehicle among untrained humans by using a new local controller called MPN and a new global multi-robot planner. The local planner uses emergency trajectories and multiple cost functions, including a human model, to integrate human motion predictions as well as a harmonic cost map field to cope with local minima. We presented preliminary results on (i) the local planner by running experiments with a real robot (Pioneer P3 DX), on (ii) the multi-robot planner using Stage Robot Simulator and on (iii) the human motion model integration by simulating human motions in GazeboSim.
The projects TransportBuddy and Autonomous Fleet will be completed in 2019 and we are planning to test all approaches on a real robot, and if possible with multiple real robots.

Acknowledgements

The research leading to these results has received funding from the Austrian Research Promotion Agency (FFG) according to grant agreements No. 855409 (AutonomousFleet) and No. 854865 (TransportBuddy).

8. References

Andreasson, H., Bouguerra, A., Cirillo, M., Dimitrov, D., Driankov, D., Karlsson, L., Lilienthal, A., Pecora, F., Saarinen, J., Sherikov, A., Stoyanov, T., 2015. Autonomous Transport Vehicles: Where We Are and What Is Missing, Robotics Automation Magazine 22, pp. 64–75.

Bader, M., Richtsfeld, A., Suchi, M., Todoran, G., Holl, W., Kastner, W., Vincze, M., 2015. Balancing Centralized Control with Vehicle Autonomy in AGV Systems, 11th International Conference on Autonomic and Autonomous Systems (ICAS), pp. 37–43.

Bemporad, A., Morari, M., 1999. Robust Model Predictive Control: A Survey, Robustness in Identification and Control 245, pp. 207–226.

Bennewitz, M., Burgard, W., Thrun, S., 2002. Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots, Robotics and Autonomous Systems 41, pp. 89–99.

Bennewitz, M., Burgard, W., Cielniak, G., Thrun, S., 2005. Learning motion patterns of people for compliant robot motion. The International Journal of Robotics Research 24, pp. 31–48.

Binder, B., 2018 Spatio-Temporal Prioritized Planning, Vienna University of Technology.

Blanco, J.-L., Gonzalez, J., Fernandez-Madrigal, J.-A., 2008. Extending Obstacle Avoidance Methods through Multiple Parameter-Space Transformations, Journal of Autonomous Robots 24, pp. 29–48.

Fischinger, D., Einramhof, P., Papoutsakis, K., Wohlkinger, W., Mayer, P., Panek, P., Hofmann, S., Koertner, T., Weiss, A., Argyros, A., Vincze, M., 2014. Hobbit, a care robot supporting independent living at home: First prototype and lessons learned Robotics and Autonomous Systems, Journal of Robotics and Autonomous Systems 75, pp. 60–73.

Fox, D., Burgard, W., Thrun, S., 1997. The Dynamic Window Approach to Collision Avoidance, Robotics Automation Magazine, pp. 23–33.

Gómez, J. V., Álvarez, D., Garrido, S., Moreno, L., 2015. Fast Methods for Eikonal Equations: an Experimental Survey, Computing Research Repository, arXiv:1506.03771.

Kleiner, A., Selecký, M., Čáp, M., Novák, P., 2015. Prioritized planning algorithms for trajectory coordination of multiple mobile robots, IEEE Transactions on Automation Science and Engineering 12, pp. 835–849.

Kollmitz, M., Hsiao, K., Gaa, J., Burgard, W., 2015. Time dependent planning on a layered social cost map for human-aware robot navigation, IEEE European Conference on Mobile Robots (ECMR).

LaValle, S. M., 2006. Planning Algorithms, Cambridge University Press.

Liess, M., Lévine, J., Martin, P., Rouchon, P., 1995. Flatness and defect of non-linear systems: introductory theory and examples, International Journal of Control 61, pp. 1327–1361.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J. Wheeler, R., Ng, A. Y., 2009. ROS: an open-source Robot Operating System, ICRA Workshop on Open Source Software.

Schouwenaars, T., Feron, E., How, J., 2002. Safe Receding Horizon Path Planning For Autonomous Vehicles, 40th Annual Allerton Conference on Communication, Control, and Computing.

Seer S., 2015. A unified framework for evaluating microscopic pedestrian simulation models, Vienna University of Technology.

Todoran, G., Bader, M., 2016. Expressive navigation and Local Path-Planning of Independent Steering Autonomous Systems, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4742–4749.

Todoran, G., 2018. Optimal Local Path-Planning and Control for Mobile Robots, Vienna University of Technology.

Ullrich, G., 2014. Fahrerlose Transportsysteme, Springer.

Wang, Q., Wulfmeier, M., Wagner, B., 2016. Voronoi-based heuristic for nonholonomic search-based path planning, Intelligent Autonomous Systems 13, Springer International Publishing, pp. 445–458.

Wray, K. H., Ruiken, D., Grupen, R. A., Zilberstein, S., 2016. Log-space harmonic function path planning, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1511–1516.

Wurman, P. R., D'Andrea, R., Mountz, M., 2007. Coordinating Hundreds of Cooperative, Autonomous Vehicles in Warehouses, 19th National Conference on Innovative Applications of Artificial Intelligence, pp. 1752–1759.