Using Metareasoning on a Mobile Ground Robot to Recover from Path Planning Failures*

Sidney L. Molnar[1†], Matt Mueller[2], Robert Macpherson[2], Lawrence Rhoads[2], and Jeffrey W. Herrmann[1,2]

***Abstract*— Autonomous mobile ground robots use global and local path planners to determine the routes that they should follow to achieve mission goals while avoiding obstacles. Although many path planners have been developed, no single one is best for all situations. This paper describes metareasoning approaches that enable a robot to select a new path planning algorithm when the current planning algorithm cannot find a feasible solution. We implemented the approaches within a ROS-based autonomy stack and conducted simulation experiments to evaluate their performance in multiple scenarios. The results show that these metareasoning approaches reduce the frequency of failures and reduce the time required to complete the mission.**

I. INTRODUCTION

An autonomous mobile robot uses path planning to determine the best way to get to a goal location [1]. Given a graph representation of the environment, metric path planning methods (such as A*) can find the shortest path (in this graph) to the goal. Replanning must occur when the robot encounters a new (or previously unknown) obstacle that makes the current path infeasible.

In practice, a path planning method might fail because it cannot find a feasible solution, which might be due to a problem with the algorithm (or its implementation) or an issue with the graph representation that it is using. If the robot has no way to recover from this failure, then it will be unable to continue, and it cannot complete its mission.

Metareasoning, which monitors and controls a robot's reasoning processes, provides a way to avoid mission failure due to path planning errors.

This paper describes two metareasoning approaches that enable a robot to select a new path planning algorithm when the current planning algorithm cannot find a feasible solution. We implemented both approaches within a ROS-based autonomy stack and conducted simulation experiments to evaluate the performance of each metareasoning approach. The results show that both metareasoning approaches reduce the frequency of failures and reduce the time required to complete the mission.

The remainder of this paper is organized as follows: Section II reviews existing work on path planning and metareasoning. Section III presents the metareasoning approaches that we developed. Section IV discusses the results of simulation experiments that we conducted to evaluate the metareasoning approaches. Section V concludes the paper.

*This work was supported by the U.S. Army Research Laboratory (Award W911NF2120076).

[1]Department of Mechanical Engineering, University of Maryland, College Park, MD 20740, USA

[2]Institute for Systems Research, University of Maryland, College Park, MD 20740, USA

[†]Corresponding author: Sidney L. Molnar, smolnar@umd.edu

II. RELATED WORK

A. Path Planning

Robot path planning has been a popular research topic for many years because a mobile robot needs to navigate through an environment in order to complete its mission. Many techniques have been developed [2], including algorithms (such as Dijkstra's algorithma and A*) for searching a graph for the shortest path. Other planning approaches include probabilistic roadmaps [3] and sampling-based search approaches such as rapidly-expanding random trees (RRT) [2] and RRT* [4], which are useful for motion planning as well. RRT^X [5] is an asymptotically optimal algorithm for replanning in dynamic environments with unpredictable obstacles; it reuses the search graph and quickly updates it based on new information. D* [6] and similar algorithms can repair a path through a graph that becomes suboptimal or infeasible due to changes in the environment.

B. Metareasoning

Metareasoning is reasoning about reasoning (or deciding how to decide) [7], [8], [9]. Metareasoning monitors and controls reasoning and decision-making processes. For a robot (or other intelligent agent), this enables bounded rationality by giving the robot the ability to reason about its own reasoning process and to consider the cost and time of computation as it considers which action to perform next.

For the engineer who is designing a robot or intelligent agent, using metareasoning delays some design decisions about the reasoning system. The engineer does not have to select and implement one algorithm which likely performs well in some situations and poorly in others. Instead, the engineer delegates decisions to the metareasoning approach, which is particularly relevant to controlling path planning, where we would like to benefit from the strengths of different algorithms.

Due to this flexibility, metareasoning can improve performance. For example, the SATzilla program, created to solve propositional satisfiability (SAT) problems, used algorithm selection (an important metareasoning approach) to win gold medals in the 2007 and 2009 SAT competitions [10]; moreover, "the solvers contributing most to SATzilla were often not the overall best-performing solvers, but instead solvers that exploit novel solution strategies to solve instances that would remain unsolved without them." Algorithm selection has been used in numerous settings, including multi-agent

search [11], combinatorial and continuous optimization [12], [13], [14], [15], and CPU thermal management [16].

Rabiee *et al.* [17] proposed an approach to competence-aware path planning. A *competence-aware* agent can manage risk by assessing the probability of a failure and taking actions to avoid future failures. Their path planning approach uses introspective perception to estimate the likelihood of failures due to degraded perception and then generates low-risk paths by solving a stochastic shortest path problem.

Svegliato *et al.* [18] proposed a safety metareasoning system that is designed to reduce the likelihood that a robot will be perform an unsafe act (such as driving into a crevice). The approach includes safety processes that monitor different risks and suggest adjustments to the task reasoning process (e.g., so that the robot's path moves away from a crevice). Simulation results showed that the approach reduced the frequency of safety concerns.

For a UAV that needs to fly quickly through a forest, Jarin-Lipschitz *et al.* [19] described a path planning approach that used metareasoning to adjust dispersion, a key planning parameter. The planning module uses a minimum-dispersion-based motion primitive planner to determine the trajectories that vehicle should follow. The value of the dispersion parameter affects the graph and the quality of the path that the local planner creates. The meta-level used three rules that changed the dispersion value in response to planner performance.

Our approach does have some similarities with that of Svegliato *et al.* [20], who described a metareasoning approach (called introspection) that interrupts an autonomous vehicle's regular reasoning process whenever an exception handler indicates that an abnormal situation is occurring (e.g., a garbage truck blocking a two-lane road). It then invokes a different decision process that is appropriate for that abnormal situation. The approach presented here is also a type of introspection, but this approach modifies the regular reasoning process (instead of interrupting it). Unlike the approach by Jarin-Lipschitz *et al.*, which adjusts a parameter value when path planning fails, our approach selects a new path planning algorithm to respond to a failure.

Although algorithm selection can be viewed as an optimization problem, a metareasoning approach that needs to solve a complex problem will add substantial overhead and might delay the reasoning processes. Thus, we were interested in simpler approaches that can quickly select a planning algorithm when necessary to overcome a path planning failure. This paper describes a study that adds to our knowledge of metareasoning by presenting novel metareasoning approaches that have low overhead and presenting results that describe their benefits.

III. APPROACH

A. Autonomy Stack

This research focused on the ARL Ground Autonomy Stack [21], which can be used to create autonomous mobile ground robots. This autonomy stack uses the Robot Operating System (ROS) and includes algorithms for image processing, simultaneous localization and mapping (SLAM), path planning, navigation, and controlling the robot's velocity. Each of these algorithms uses a network of nodes and topics that communicate by publishing and subscribing to messages that initiate callback functions to influence the robot's actions.

The autonomy stack currently includes four global path planners and three local path planners that were added by ARL researchers and their collaborators. (We did not create or modify these planners.)

The global path planners find a path from the robot's current location to the goal location within a relatively static map. The SBPL global path planner constructs a graph using motion primitives and uses A* to find the best path to the goal location along that graph [22]. The RDGP global path planner constructs a path from the start location to the goal location by finding a fixed number of points evenly spaced along the line segment between those locations' 3D coordinates; the target pose at each point is the same as the target pose at the goal location. The GLS global path planner uses a generalized lazy search [23]. The EASL global path planner uses an efficiently adaptive state lattice and searches with RRT* [24].

The local path planners determine the best way for the robot to move along the global path using information from its immediate environment, allowing the robot to navigate around dynamic and recently-discovered obstacles. The local plan operates independently to the waypoint goal. The NLOPT local path planner uses parameterized control functions, formulates the optimal control problem as a nonlinear optimization problem, and solves that problem numerically [25]. The MPPI local path planner uses a version of model predictive control to make path planning decisions quickly [26]. The RHMPC local path planner creates a fine-grained graph of the region around the robot and uses A* to find the best path to the next waypoint along that graph [27].

B. Path Planning Failures

Path planners can sometimes fail for several reasons. For instance, different path planning algorithms become more or less successful as a result of environmental factors. If a path planner is being used in an environment that is not conducive to its functionality, then the algorithm will likely struggle to find feasible solutions that allow the robot to navigate over the terrain and around the obstacles. The second reason a path planner might fail is when an algorithm uses a predictive heuristic function that is unaware of future obstacles in the path. As the ground robot comes across a new obstacle, the planning algorithm might not have enough time to find a new plan before the ground robot collides with the new obstacle.

Figure 1 shows an example of a simulated robot that is stuck behind some trees in a forest and unable to correct its path planning algorithm to find a way out of its current position, which is surrounded by many obstacles. Because of the path planning failure, the robot cannot complete its mission. Motivated by such undesirable events, we investigated whether metareasoning can overcome path planning failures.

Fig. 1. Simulated robot stuck behind a cluster of trees, where its planner is unable to create a feasible path.

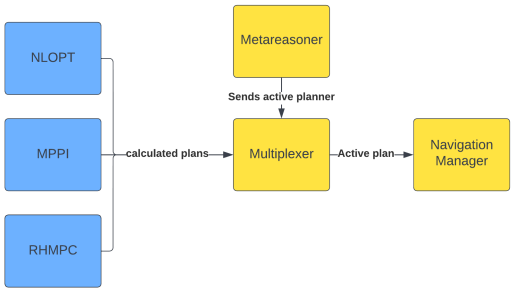

Fig. 2. Parallel metareasoning logic.

C. Metareasoning Approaches

After studying the autonomy stack and how the global and local path planners operate within that structure, we developed two metareasoning approaches that "switch" from one planner to another when a path planning failure occurs.

Parallel Approach. The "parallel" approach uses three local path planners (NLOPT, MPPI, and RHMPC) and one global path planner (EASL). The autonomy stack starts ROS nodes for all three local path planners, and they run continuously during the mission. The metareasoning node decides which local planner is "active" and informs the multiplexer node. That node receives information from all three local planners but transmits only the output of the active local planner to the navigation manager. The outputs from the other local planners are not used. Initially, the active local planner is NLOPT. A path planning failure occurs if one of the following events occurs: the active planner sends a stuck message, the navigation manager cancels the active local planner, or the robot has not moved for three seconds. When a failure occurs, the metareasoning node makes another local planner active. In particular, if NLOPT was active, then the new active local planner will be MPPI; if MPPI was active, then the new active local planner will be RHMPC; if RHMPC was active, then the new active local planner will be NLOPT. In addition, the metareasoner node checks the status of the local planner nodes as more than one local planner might be failed. If the new active local planner is failed, then the remaining local planner becomes the active one. If all three local planners fail, then the robot cannot complete its mission. Figure 2 exemplifies the logic used for the parallel approach.

Sequential Approach. The "sequential" approach uses a predetermined sequence of global and local planner combinations. When the metareasoning node receives a stuck or error message produced by one of the planners, it calls the next planner combination through a new launch file. By launching new planner nodes under the same name as the old planner nodes, the original nodes are automatically killed and replaced by the new nodes. Figure 3 demonstrates the logic used for the sequential approach.

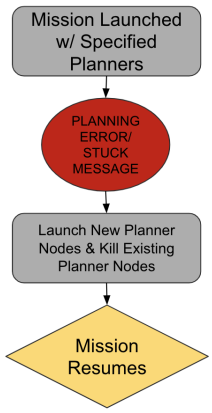

Fig. 3. Sequential metareasoning logic.

Four global planners (SBPL, GLS, EASL, and RDGP) and two local planners (NLOPT and MPPI) were used in the sequential approach. We first tested these combinations in three simulation scenarios. For each run of the simulation, we recorded whether the robot failed to reach the goal location (a mission failure) or (if it was successful) the time needed to reach the goal. The "success rate" is the number of successes divided by the number of runs. The "time to success" is the average time for successful runs.

We selected the five combinations that had the best time to success and used those for one metareasoning policy, which will be known as the Sequential TS policy (Table I). We also selected the five combinations that had the best success rate and used those for a second metareasoning policy, which will be known as the Sequential SR policy (Table II). Within each policy, we sequenced the planner combinations from best performance to worst performance.

TABLE I

THE SEQUENTIAL TS (TIME-TO-SUCCESS) METAREASONING POLICY

Call Order	Planner Combination (global-local)	Average Time To Success (s)
1	**EASL-NLOPT**	179.46
2	**EASL-MPPI**	195.04
3	**RDGP-MPPI**	212.95
4	**GLS-MPPI**	239.58
5	**GLS-NLOPT**	240.63

TABLE II

THE SEQUENTIAL SR (SUCCESS RATE) METAREASONING POLICY

Call Order	Planner Combination (global-local)	Average Success Rate (%)
1	**GLS-MPPI**	70
2	**EASL-MPPI**	68
3	**EASL-NLOPT**	63
4	**GLS-NLOPT**	53
5	**SBPL-MPPI**	38

D. Testing

To evaluate the metareasoning approaches, we used the Unity simulation engine to simulate the behavior of a mobile ground robot (a Clearpath Warthog) that was controlled by the autonomy stack while completing a mission to move from a start location to a goal location via a series of waypoints in a simulation scenario.

We prepared three simulation scenarios. In each one, the robot was given a goal location. In the least challenging scenario, known as "Market Loop" (Figure 4), the robot was given more waypoints to guide the path planning; in the intermediate scenario, known as "Market Loop (Hard)" (Figure 5), only two waypoints were given. In the most challenging scenario, known as "Forest Run" (Figure 6), the robot was given some waypoints, but they led the robot through a hilly area with many trees. In each of the figures, the waypoints are given as indigo circles. The red line connecting the waypoints indicate the robot's expected path.

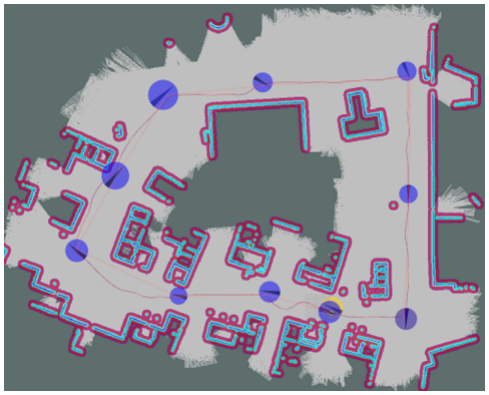

Fig. 4. Market Loop: the least challenging scenario.

IV. RESULTS

A. Performance of Algorithms

The various individual global-local path planner combinations were tested first. Each combination was tested 20 times for each of the three scenarios, except for the EASL-NLOPT and EASL-MPPI planner combinations which were tested 30 times for each course. The average performance metrics with regard to time to success and success rates for each combination are listed in Table III and IV respectively.

The Sequential SR and Sequential TS metareasoning policies were also tested over each scenario 20 times apiece, while the parallel metareasoning policy was tested 30 times for each scenario. Tables III and IV show the performance metrics for each of the metareasoning policies with regard to time-to-success and success rate respectively.

Figures 7, 8, and 9 show the spread of time-to-success results for the Market Loop, Market Loop (Hard) and Forest Run scenarios respectively. Figure 10 shows success rate results for each planner combination and metareasoning policy with respect to each scenario.

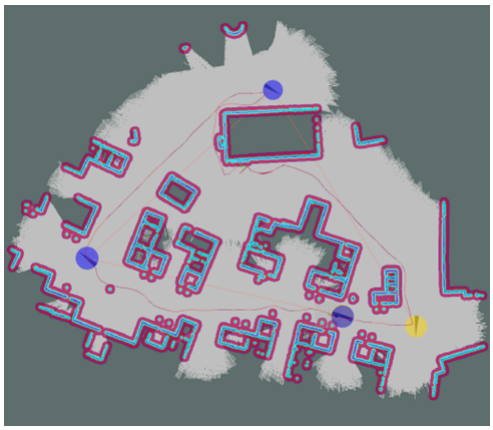

Fig. 5. Market Loop (Hard): the intermediate scenario.

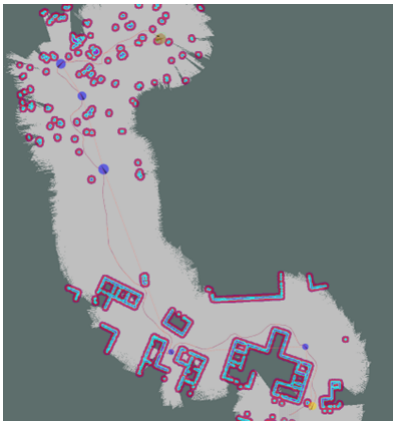

Fig. 6. Forest Run: the most challenging scenario.

TABLE III

AVERAGE TIME TO SUCCESS (SECONDS) FOR EACH PLANNER COMBINATION AND METAREASONING POLICY (HIGHLIGHTED CELLS ARE THE BEST FOR EACH SCENARIO)

Planner Combinations	Market	Market (Hard)	Forest	Average (All)
GLS-MPPI	250.36	209.41	258.97	239.58
GLS-NLOPT	250.88	232.39	238.61	240.63
SBPL-MPPI	260.26	N/A	265.47	262.86
SBPL-NLOPT	319.03	N/A	221.35	270.19
RDGP-MPPI	212.95	N/A	N/A	212.95
EASL-MPPI	199.88	165.65	219.58	195.04
EASL-NLOPT	169.90	165.82	202.65	179.46
Metareasoning Policy	Market	Market (Hard)	Forest	Average (All)
Sequential TS	169.19	215.68	258.37	214.41
Sequential SR	190.04	161.30	174.63	175.63
Parallel	181.81	170.01	193.58	181.80

TABLE IV

SUCCESS RATE (%) FOR EACH PLANNER COMBINATION AND METAREASONING POLICY (THE HIGHLIGHTED CELLS ARE THE BEST FOR EACH SCENARIO)

Planner Combinations	Market	Market (Hard)	Forest	Average (All)
GLS-MPPI	75	85	50	70
GLS-NLOPT	80	65	15	53
SBPL-MPPI	75	0	40	38
SBPL-NLOPT	30	0	5	12
RDGP-MPPI	80	0	0	27
EASL-MPPI	70	83	50	68
EASL-NLOPT	83	77	30	63
Metareasoning Policy	Market	Market (Hard)	Forest	Average (All)
Sequential TS	95	85	50	77
Sequential SR	85	80	40	68
Parallel	90	100	80	90

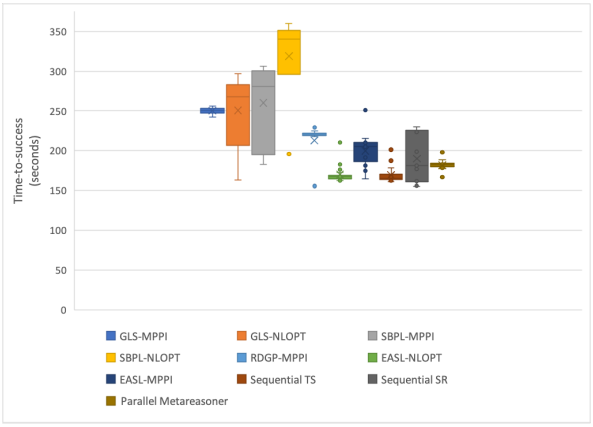

Fig. 7. Box plots showing time-to-success results for each planner combination and metareasoning policy in the Market Loop scenario.

For the time-to-success metric, the EASL-NLOPT planner combination outperformed all other stand-alone planner combinations in the Market Loop and Forest Run scenario, falling just short of the EASL-MPPI planner combination in the Market Loop (Hard) scenario by less than 0.2 seconds. The EASL-NLOPT planner combination also had the best overall time-to-success average of any other planner combination.

For the success rate metric, the GLS-MPPI planner combination outperformed all other planner combinations for the Market Loop (Hard) and Forest Run scenarios, tying the EASL-MPPI planner combination at 50% for Forest

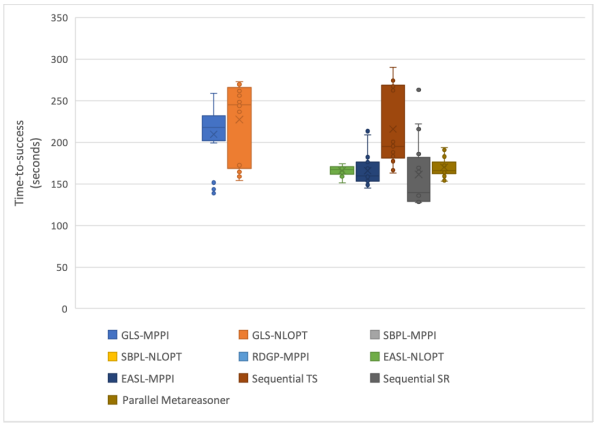

Fig. 8. Box plots showing time-to-success results for each planner combination and metareasoning policy in the Market Loop (Hard) scenario.

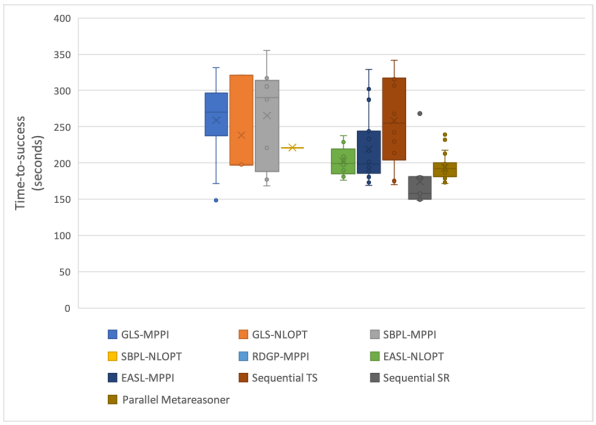

Fig. 9. Box plots showing time-to-success results for each planner combination and metareasoning policy in the Forest Run scenario.

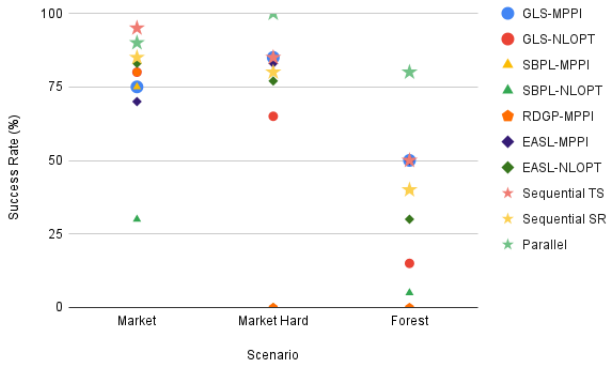

Fig. 10. Success rates for planner combinations and metareasoning policies in each of the scenarios.

Run. The EASL-NLOPT planner combination outperformed the other planner combinations only in the Market Loop scenario. The best overall performance for success rate in all three scenarios was again the GLS-MPPI planner combination.

In the Market Loop scenario, using the Sequential TS metareasoning policy led to time-to-success performance that was better than all seven planner combinations, but it did not improve time-to-success for three of the planner combinations in the Market Loop (Hard) and any of the planner combinations Forest Run scenarios. Using the Sequential TS metareasoning policy led to a better success rate than any planner combination in the Market Loop scenario. Its success rate was better than six planner combinations in the Market Loop (Hard) scenario, and its success rate was better than five planner combinations in the Forest Run scenario.

Using the Sequential SR metareasoning policy led to time-to-success performance that was better than six of the planner combinations in the Market Loop scenario. Its time-to-success performance was better than all seven planner combinations in the Market Loop (Hard) scenario and the Forest Run scenario. Using the Sequential SR metareasoning policy led to a better success rate than any planner combination in the Market Loop scenario and five of the planner combinations in the Market Loop (Hard) scenario. Its success rate was better than four of the planner combinations in the Forest Run scenario.

Using the Parallel metareasoning policy led to time-to-success performance that was better than six of the planner combinations in the Market Loop scenario. Its time-to-success performance was better than five planner combinations in the Market Loop (Hard) scenario, and its time-to-success performance was better than all seven planner combinations in the Forest Run scenario. For all three scenarios, using the Parallel metareasoning policy led to a better success rate than any planner combination.

For the time-to-success performance metric, the Sequential TS metareasoning policy outperformed the other two metareasoning policies in the Market Loop scenario. The Sequential SR metareasoning policy outperformed the other two policies for time-to-success performance in the Market Loop (Hard) and Forest Run scenarios. The Sequential SR metareasoning policy performed best overall in terms of time-to-success.

For the success rate performance metric, the Parallel metareasoning policy outperformed the two sequential policies for the Market Loop (Hard) and Forest Run scenarios. The Sequential TS metareasoning policy outperformed the other two policies for success-rate performance in the Market Loop scenario. The Parallel metareasoning policy performed best overall in terms of success rate.

Overall, although no metareasoning policy dominated the other metareasoning policies in all cases on both performance metrics, adding metareasoning did improve the robot's performance. Thus, these results suggest that engineers consider metareasoning to overcome the limitations of using fixed planning algorithms.

V. CONCLUSION

This paper presented two metareasoning approaches for controlling the global and local path planners that operate in an autonomy stack for a mobile ground robot. Both approaches react to path planning failures by selecting a new planner. The *parallel* approach runs a single global planner and three local planners simultaneously, and its metareasoning node determines which local planner's output is used by the autonomy stack's navigation manager. The *sequential* approach starts a single planner combination (a global planner and a local planner) but stops this and starts a different combination when a failure occurs.

The results of simulation testing show that both metareasoning approaches were successful at reducing the frequency of mission failures and time needed to complete the mission on the scenarios that we considered. The parallel metareasoning policy increased the frequency of mission success more than the sequential policies that we tested.

One disadvantage of the parallel metareasoning approach is that it requires more changes to the autonomy stack. Additional nodes are needed to run the local planners and to filter the planners' output so that the navigation manager uses only the output of the currently active local planner. The sequential metareasoning approach requires adding only the metareasoning node, which monitors the existing ROS topics and stops and starts the appropriate ROS nodes when needed. Unlike the parallel metareasoning approach, which is running a global planner and multiple local planners simultaneously, the sequential metareasoning approach runs one global planner and one local planner, which should reduce the computational burden of the autonomy stack. Our future work will include quantifying the computational resources required.

Future work should consider other sequences of planners and using a parallel approach with multiple global planners (not only local planners). This work took the existing global and local planners as given; we did not consider adding new planners to the autonomy stack, although there might be other algorithms that can perform better than the existing global and local planners. Including better planners might reduce the need for metareasoning (because they fail less often) or improve the performance of the metareasoning approaches. In addition, it might be productive to develop and test a competence-aware path planning approach [17] to predict path planning failures and plan paths that avoid locations where path planning failures are more likely.

Ultimately, we envision developing metareasoning approaches that can optimize all of the reasoning processes in the autonomy stack subject to the limits of the computational resources that are available on the robot.

ACKNOWLEDGMENT

The authors appreciate the suggestions and assistance provided by Kevin Carey, Siddharth Gopal, Eashwar Sathyamurthy, and our collaborators at the Army Research Laboratory.

REFERENCES

[1] Robin Murphy. *Introduction to AI Robots*. The MIT Press, 2019.

[2] Steven M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[3] L Kavraki, P Svestka, J Latombe, and MH Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[4] S Karaman and E Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, 2011.

[5] M Otte and E Frazzoli. RRT-X: Asymptotically optimal single-query sampling-based motion planning with quick replanning. *International Journal of Robotics Research*, 35(7):797–822, 2016.

[6] A Stentz. The focussed D* algorithm for real-time replanning. In *International Joint Conference on Artificial Intelligence, Montreal, Canada*, page 1652–1659, 1995.

[7] Michael T. Cox and Anita Raja. *Metareasoning: Thinking about Thinking*. MIT Press, Cambridge, Massachusetts, 2011.

[8] Stuart Russell and Eric Wefald. *Do the Right Thing: Studies in Limited Rationality*. MIT Press, Cambridge, MA, USA, 1991.

[9] Stuart Russell and Eric Wefald. Principles of metareasoning. *Artificial Intelligence*, 49(1):361–395, 1991.

[10] Lin Xu, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Evaluating component solver contributions to portfolio-based algorithm selectors. In *International Conference on Theory and Applications of Satisfiability Testing*. Springer, Berlin, 2012.

[11] Estefany Carrillo, Suyash Yeotikar, Sharan Nayak, Mohamed Khalid M. Jaffar, Shapour Azarm, Jeffrey W. Herrmann, Michael Otte, and Huan Xu. Communication-aware multi-agent metareasoning for decentralized task allocation. *IEEE Access*, 9:98712–98730, 2021.

[12] Lars Kotthoff. *Algorithm Selection for Combinatorial Search Problems: A Survey*, page 149–190. Springer International Publishing, 2016.

[13] Kevin Leyton-Brown, Eugene Nudelman, and Yoav Shoham. Empirical hardness models: Methodology and a case study on combinatorial auctions. *Journal of the ACM*, 56(4):1–52, 2009.

[14] Mario A. Muñoz, Michael Kirley, and Saman K. Halgamuge. *A Meta-learning Prediction Model of Algorithm Performance for Continuous Optimization Problems*, volume 7491 of *Lecture Notes in Computer Science*, page 226–235. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[15] Mario A. Muñoz, Yuan Sun, Michael Kirley, and Saman K. Halgamuge. Algorithm selection for black-box continuous optimization problems: A survey on methods and challenges. *Information Sciences*, 317:224–245, 2015.

[16] Michael Dawson and Jeffrey W. Herrmann. Metareasoning approaches for thermal management during image processing. In *Proceedings of the ASME 2022 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, St. Louis, Missouri, 2022.

[17] Sadegh Rabiee, Connor Basich, Kyle Hollins Wray, Shlomo Zilberstein, and Joydeep Biswas. Competence-aware path planning via introspective perception. *IEEE Robotics and Automation Letters*, 7(2):3218–3225, 2022.

[18] Justin Svegliato, Connor Basich, Sandhya Saisubramanian, and Shlomo Zilberstein. Metareasoning for safe decision making in autonomous systems. In *2022 IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[19] Laura Jarin-Lipschitz, Xu Liu, Yuezhan Tao, and Vijay Kumar. Experiments in adaptive replanning for fast autonomous flight in forests. In *2022 IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[20] Justin Svegliato, Kyle Hollins Wray, Stefan J Witwicki, Joydeep Biswas, and Shlomo Zilberstein. Belief space metareasoning for exception recovery. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1224–1229. IEEE, 2019.

[21] U.S. Army. SARA CRA overview. https://www.arl.army.mil/business/collaborative-alliances/current-cras/sara-cra/sara-overview/.

[22] Maxim Likhachev. Search-based planning with motion primitives. https://www.cs.cmu.edu/˜maxim/files/tutorials/ robschooltutorial_oct10.pdf.

[23] Aditya Mandalika, Sanjiban Choudhury, Oren Salzman, and Siddhartha Srinivasa. Generalized lazy search for robot motion planning:

Interleaving search and edge evaluation via event-based toggles. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 29, pages 745–753, 2019.

[24] Benned Hedegaard, Ethan Fahnestock, Jacob Arkin, Ashwin Menon, and Thomas M. Howard. Discrete optimization of adaptive state lattices for iterative motion planning on unmanned ground vehicles. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5764–5771, 2021.

[25] Thomas M Howard and Alonzo Kelly. Optimal rough terrain trajectory generation for wheeled mobile robots. *The International Journal of Robotics Research*, 26(2):141–166, 2007.

[26] Nolan Wagener, Ching-An Cheng, Jacob Sacks, and Byron Boots. An online learning approach to model predictive control. In *Robotics: Science and Systems 2019*, 2019.

[27] Thomas M. Howard, Colin J. Green, and Alonzo Kelly. Receding horizon model-predictive control for mobile robot navigation of intricate paths. In *A. Howard et al. (Eds.): Field and Service Robotics 7*, pages 69–78, 2010.