

Towards Online Characterization of Autonomously Navigating Robots in Unstructured Environments

Jeffrey N. Twigg, Jason M. Gregory and Jonathan R. Fink

Abstract—Autonomous platforms are confronted by a diversity of challenges in unstructured environments, which make monitoring performance a non-trivial task. Some of these environments are so complex that they preclude persistent, nearby operator oversight. This absence of oversight motivates the need for an online monitoring system, specifically for robots operating in difficult environments. We develop a test methodology and set of online monitoring metrics by extending methods for characterizing robotic-systems using offline metrics. We implement this test methodology in an unstructured, outdoor environment and show the resulting performance information gained from our online monitoring solution. This online monitoring approach is generalizable such that it characterizes any robotic system that meets our set of hardware and software criteria.

I. INTRODUCTION

Disaster recovery [10], homeland security [18], rescue operations [19], and military missions [15] are all situations where it is vital to know the current level of reliability/performance of a self-navigating system because of the required stand-off distance. While there are many component level monitoring tools, it is not clear that top-level performance can be resolved from the states that each component reports. Another method of determining performance is to observe the data that a navigating robot transmits to an operator, but this assumes there is sufficient bandwidth and wireless infrastructure to enable such operations. This performance monitoring problem becomes even more difficult as the number of tasks and robots increase.

We extend existing approaches for calculating offline metrics to create online metrics. We call this *monitoring*; that is, the process of measuring the current performance of in-situ autonomous navigation. This performance evaluation is designed for a navigating robotic system as an extension of other characterization techniques. Currently, there are online diagnostics which monitor performance of individual components. One example is the diagnostics software in the Robot Operating System (ROS) which are used to characterize failures in hardware [20]. Other approaches have evaluated the quality of maps produced by Simultaneous Localization and Mapping (SLAM) techniques [28], [23]. While these approaches are useful for understanding component online performance, component level performance is not always enough to evaluate full system performance.

There have also been different approaches towards characterizing full self-navigating systems. Investigations into

the Crusher [22] [27] and other robots [7] [16] have been performed in field experiments. However, the focus of these approaches was not on monitoring current performance. One way to accomplish this is to create a highly controlled environment to challenge a robot in specific ways [13]. Often this approach is formulated as a competition with goal completion as a method for characterizing performance [11] [17] [29]. Other approaches involve rigorous ground truth systems for detecting deviations from global values [12] or extensively sampling of an environment [6]. We refer to approaches that use an external validation system as *benchmarking* approaches. While these approaches address questions about overall performance, they cannot be easily applied to monitor a robot's current navigation performance in an unstructured environment because validation systems do not exist or cannot be installed.

However, one benchmarking approach is especially relevant because it characterizes the robot's performance without a large amount of infrastructure external to the robot; it uses cameras mounted on a robotic system which allows it to verify its successful navigation to landmarks in an indoor environment [26]. This approach tests the robotic system's ability to navigate between surveyed *landmarks*, which are visually identified by a computer mounted on a robot in an indoor environment. In these experiments, three different robots are tested and small changes to the environment are made across trials. In order to vary the conditions of the test different *challenges* are developed for each set of *landmark* navigation trials. We extend this work by relaxing some of the constraints regarding external validation and adopt some principles for our methodology.

In this work we propose a metric-based approach towards monitoring the current performance of a robot navigating in an unstructured environment. The contributions of our work are: 1) we propose a methodology that enables quick characterization of autonomously-navigating robotic systems in an unstructured environment (Sections II and III); and 2) we demonstrate that a set of online, velocity based metrics are useful for developing a qualitative understanding of the performance of a robotic system through a series of experiments (Sections IV and V). Furthermore, we validate these metrics by generating a number of offline metrics comparable to other approaches.

II. APPROACH

As with benchmarking, a monitoring method should be easy, scientific, and comparable [26] [17]. In order to main-

United States Army Research Laboratory, 2800 Powder Mill Rd, Adelphi, MD 20783 (jeffrey.n.twigg, jason.m.gregory1, jonathan.r.fink3).civ@mail.mil

tain the simplicity of an indoor benchmarking approach [26], we also describe how to survey waypoints for monitoring tests. While there are factors out of our control in unstructured environments, we attempt to limit their affects through intelligent waypoint choices. Finally, we make results comparable to disaster and security contexts by limiting the amount of surveying and instrumentation external to the robot.

Our monitoring approach consists of defining a set of GPS-referenced goal positions, referred to as *waypoints*, for an autonomous robot to execute, during which quantitative metrics are measured to produce qualitative conclusions regarding the performance of navigation. The low cost of surveying and executing a series of waypoints described in Section III make this technique cheap and easy. Furthermore, metrics are defined in such a way that performance can be compared for different sets of waypoints as well as different regions within the environment. To use our monitoring approach, we define a set of basic, but necessary assumptions, which makes this method generalizable to a range of platforms and environments. These assumptions are:

- GPS data must be present in the environment while the experiment is being setup.
- It must be possible to place a *starting platform* at a set of desired locations in the environment.
- The robot must produce a map in the GPS frame.
- The robot must provide a method for tracking time and estimating its pose.
- The robot must possess a navigation interface by which waypoints can be defined.

A difficult part of experimenting in an unstructured environment is achieving repeatable trials. While it may be possible to assume that navigation in a structured indoor environment is deterministic, this assumption does not hold in outdoor scenarios. The coefficient of friction between the wheels of a ground platform can change as humidity varies, or dirt accumulates on the wheels. Flying platforms will also experience changes with wind velocity. The angle of the sun can change the effectiveness of different laser and camera sensors. These and other changes in the environment over time make waypoint navigation trajectories non-deterministic. Even though it is not possible to make this environment deterministic or as rigorous as an indoor one, we make the performance of the robot in an outdoor environment more controlled through proper waypoint choice.

One of the goals of this monitoring approach is monitoring performance information without having to instrument or survey the environment in order to maintain comparability to disaster and security contexts. As result, we rely on GPS to provide an external reference frame which is often available in these disaster and security contexts. Unfortunately, poor GPS information can also complicate the online analysis which we have developed for monitoring robots because of how the surveyed goal positions are tied to GPS. Errors in GPS can cause shifts in GPS defined positions. As a

result, these points might project into an obstacle which is a goal the robot cannot navigate to or it could shift into a position making it easier for a robot to achieve its goal. In Section III we talk about how to mitigate the errors in this approach caused by GPS. In Section V we demonstrate that this approach is still effective despite poor GPS.

A. Metrics

In the context of autonomous navigation, we can adapt metrics presented in previous efforts towards both monitoring and benchmarking. For example, time moving, average velocity, time to failure, and comparison with a path deemed “ideal” are all applicable to testing navigation. We note, however, not all of these metrics can be implemented in an online fashion.

Time stopped in comparison with time moving is a useful metric [22] in autonomy because it is indicative of whether or not the robot is spending more time planning than moving. Time stopped could indicate that there is a failure which the robot may or may not be able to recover from. Our approach separates time moving into two categories: *time moving forward* and *time moving backward*. Time spent moving is indication the planner and supporting systems are functional. Moving backward is more often part of recovery operation on our platforms so it should be tracked differently than time moving forward.

Another relevant metric is average velocity [27]. In our implementation, we track average velocity while the robot is moving forward and backward. Average velocity is an indicator of the agility of the platform mechanics as well as mapping and planning.

Some metrics are only applicable in cases of successful or failed navigation so they cannot be implemented online. Mean time between failures, for example, is a metric [29] that is only applicable for failures. The difference of the total path distance and the ideal path distance is a metric can be computed at the end of execution. Smaller deviations from the ideal path suggest improved navigation performance [26]. Our approach computes this value in cases of successfully navigating to all of the defined waypoints.

Metrics which can be implemented online are related to time, distance, and velocity as these are fundamental to progress and do not necessarily require execution completion to compute. Velocity of the robot with respect to the environment and velocity with respect to the next waypoint are the two values that form the basis of our monitoring approach.

These characterization metrics can be used to determine change in performance over time. To derive these metrics, we describe the how to set up the required experiments.

III. METHODOLOGY

We describe how to quickly identify and survey waypoints for an experiment using our approach. The connected series of surveyed waypoints constitute what is referred to as a *mission*. For each mission, we executed two different types of trials: remote-controlled, referred to as *ideal path*,

and *autonomous*. We present our instantiation of software, comprised of different *analyzers*, as an example of how to log the necessary information during monitoring missions. The development of this methodology along with our monitoring approach is our contribution to the body of performance analysis research for autonomous systems.

A. Environment Survey and Set-up

Choosing waypoints for monitoring experiments is important for obtaining relevant results. It is useful to follow some heuristics when considering waypoint placement based on GPS properties and platform characteristics. In our implementation we say the robot has arrived at a waypoint if it is able to navigate to a position within 2 m of the corresponding GPS waypoint. Errors in GPS localization can make it so that the perceived position of a waypoint can shift. This shifting directs a lot of waypoint choices. If the waypoint is too close to an obstacle, the robot could perceive the waypoint as beyond or inside an obstacle which will dramatically change the results.

Waypoint choice is also platform dependent. For our 1m scale platforms, we chose waypoints 3 to 5 m from obstacles and at least 5m from each-other. These waypoint distances are aligned with the maneuverability of the platforms and how close they can safely get to large obstacles.

It is also desirable to choose waypoints that intuitively vary based on the level of difficulty for a specific platform to achieve them. This choice of waypoints forms the basis of an experiment. For example some waypoints should be connected through flat and sloping ground for ground platforms, or wide versus narrow corridors for flying platforms. This is a similar approach where *challenges*, a fundamental attribute of a mission, are assigned to different groups of landmarks [26]. We also define one challenge as a control for the other challenges. This challenge consists of connecting points where there is nothing to inhibit the navigation of the platform. The point of this control challenge is to determine the robot's performance under ideal conditions.

B. Mission Initialization

It should be noted that for each starting location, we specified a starting orientation using a compass to limit the variability in the initial conditions for every mission. The robot is initialized in the same location and orientation using a custom initialization platform that fit each robot exactly and held a location and orientation for a series of trials. This method of initialization is similar to an existing approach where the robot is equipped with up-facing camera and placed under a landmark so its camera is directly beneath the center of the landmark [26]. Our method satisfies an analogous constraint in an unstructured environment.

C. Ideal Path Generation

As defined in [26], the ideal path is the most direct path between two landmarks, as taken by a robot or person moving normally. In this case, we have a human controller teleoperate



Fig. 1. Starting Platform with robot position and orientation fixed by removable bracket

the robot between waypoints because they can take into consideration the surrounding environmental factors, plan optimized paths, provide obstacle detection and avoidance, and efficiently navigate to desired locations, all without the influence of sensor error bias. In each ideal path, the robot begins in the starting location and orientation, and a human operator manually drives the robot using a wireless joystick. The operator walks behind the robot and always has line-of-sight vision to the robot and the environment immediately surrounding the robot. At each waypoint location, the user stops the robot and presses a button on the joystick to indicate that the robot has successfully achieved the waypoint. The robot acknowledges this progress and then instructs the operator to continue to the next waypoint. This process is repeated until every waypoint in the trial is achieved, at which point the trial ends. During each ideal path trial, all mapping and navigation data is recorded, as well as the commanded velocities provided by the operator.

Using the autonomous system to generate the ideal path is important because it captures the limitations of the platform. For example, larger platforms might be able to traverse curbs that smaller platforms may not. It is also possible that a small flying platform might be able to fly through a window of size where a larger flying platform may not. These differences in platform capability mean that the reference ideal path needs to be determined by platforms of similar size and means of locomotion; otherwise, there will likely be some level of bias in the results of the platform performance [26].

D. State Analysis Framework

We developed a software suite to illustrate the means in which the aforementioned metrics can be observed for the different components of the system and refer to these tools as *Analyzers*. These Analyzers provide offline tools in a manner similar to that of [25] as well as online performance metrics for monitoring. They are implemented in the form of ROS nodes and are designed to be flexible for different platforms. The Analyzer software is modular and each Analyzer is responsible for monitoring the state of a specific hardware or software component, or set of components, in the system.

The *Platform Analyzer* is mainly responsible for tracking diagnostic messages, namely warnings and errors, from the robot's sensors. The Platform Analyzer also measures the average and maximum forward and backward velocities as well as the different ratios of time moving forward, backward, and stopped. Using these different measurements, the Platform Analyzer is able to provide insight regarding the robot's performance.

The *Navigation Analyzer* handles all things related to the robot's global planner, local planner, and local controller. This Analyzer measures the navigation durations, distances, and paths that the robot executes during a mission. From this, we can compute the progress the robot achieves in a trial as it relates to the total number of waypoints in a trial or the total distance of a mission. It is also possible to compute the euclidean distance to the next waypoint.

IV. EXPERIMENTS

To test our integrated monitoring approach, we performed a series of experiments using two ground robots at a military training facility, which represents a complex, unstructured environment. We surveyed the operational environment for waypoints, specifically so that relevant *challenges* were present for each mission.

A. Robot Description

We used two different ground robots for the generation of performance data in a real-world environment. The two robotic platforms are the iRobot PackBot [5] and the Army Research Lab (ARL)-developed SPEAR. The PackBot, seen in figure 2, is a military-grade, tracked platform capable of speeds up to 2 m/s and traversing both indoor and outdoor terrains. The SPEAR, seen in figure 3, is similar to the PackBot in size and speed; however, the SPEAR is a light weight, wheeled, research platform. To enable autonomous operations, the PackBot is outfitted with a processing payload containing a Quad-Core Intel i7 ICOM express board and a 256 GB solid-state drive (SSD). The SPEAR is equipped with MIO-5290 Advantech board with a Dual-Core Intel i5 processor and a 512 GB SSD. The robots collect 3D point cloud data by nodding a Hokuyo UTM-30LX-EW LiDAR [3] with a Dynamixel servo. This Hokuyo LiDAR has a 270° field of view, 30 m range, and 1 mm resolution. Accurate state information is achieved using a MicroStrain 3DM-GX3-25 inertial measurement unit (IMU) [4] mounted on a custom-made vibration isolator. Additionally, a Garmin 18x PC GPS sensor [2] is elevated on a mast in an effort to receive better GPS measurements. Finally, an ASUS Xtion Pro Live can provide RGB data [1]; however, in this work it was not used for sensing or localization. Both robots use Ubuntu 14.04 (Trusty) and leverage the open-source *Robotics Operating System* (ROS) Indigo [20] to support higher-level algorithms for mapping, navigation, and autonomous capabilities.

We adopt a modern, *graph-based* solution to address the simultaneous, localization and mapping problem (SLAM)

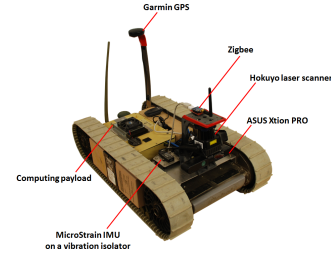


Fig. 2. iRobot PackBot with ARL sensing and computing payload

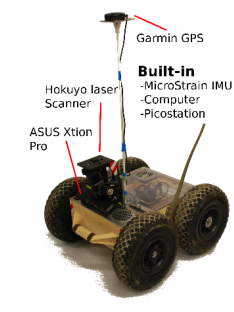


Fig. 3. ARL Small Portable Economical Autonomous Robot

problem based on the square-root smoothing and mapping ($\sqrt{\text{SAM}}$) technique [8] and the *GTSAM* software library developed at the Georgia Institute of Technology [9]. Our technique, referred to as *OmniMapper*, leverages the Generalized Iterative Closet Point (ICP) algorithm [24] for dense inter-frame matching of point cloud data and loop closure constraints. GPS measurements, when available, are robustly incorporated into our solution based on the techniques described in our previous work [21]. In terms of planning and navigation, we rely on the Search-Based Planning Library (SBPL) [14].

Towards maximizing the performance of our mapping and navigation software components, we pay particular attention to sensor calibration. More specifically, we calibrate both the MicroStrain IMU and Hokuyo laser rangefinder using internally-developed software before any experimentation.

B. Chosen Waypoints

The operating environment contained small, cinder block buildings and different obstacles that introduced clutter. Buildings were connected by gravel roads; the off-road terrain was uneven and included a mixture of tall grass and dirt, making the SLAM problem non-trivial for our low height platforms. Using previous experience with our platforms we chose waypoints which would have different *challenges*. Examples of these locations are shown in figure 4. From our list of surveyed locations, we constructed missions that consisted of a single starting location, and three or more waypoints. These missions were designed to vary the type of *challenge* by controlling the 1) number of waypoints; 2) distance between waypoints; 3) terrain complexity; and 4)



(a) Waypoint near aperture



(b) Waypoint near clutter

Fig. 4. Two surveyed waypoints



Fig. 5. Mission definitions with waypoints connected by rough approximation of ideal path and labelled by mission number. Missions 1, 2, 3, 4, 5 correspond to the blue, yellow, black, teal and magenta lines, respectively. The reference mission trials are all represented by the dark green line and act as a control for the *challenges* in other missions

obstacle aperture size. The resulting missions can be seen in figure 5. In this figure, missions begin at the numbers and are denoted by individual colors.

V. RESULTS

Over the course of these experiments, we completed 140 trials, during which the robots autonomously traversed over 13.2 km and operated autonomously for 6.4 hours. Some percentage of this constituted methodology refinement, but most is data used in reported results. Table I summarizes the trials and experiments conducted. Multiple trials of different experiments yield a representative data set for offline metrics and assessing the value of online monitoring.

The varying number of trials for each experiment is due to the varying level performance of the autonomy on the SPEAR and PackBot platforms. In each experiment the goal was to get a range of performance from failure to success. In some experiments it was either more difficult to get the full spectrum of performance, so these experiments have more trials.

In the following subsections we present the offline metrics established by other characterization approaches and show

TABLE I
EXPERIMENTS AND TRIALS

| Test Number | Challenge Type | PackBot trials | SPEAR trials | Ideal Distance (m) |
|-------------|----------------|----------------|--------------|--------------------|
| R | ref. | 14 | 4 | 459 |
| 1 | clutter(A) | 10 | 3 | 78 |
| 2 | clutter(B) | 9 | 8 | 75 |
| 3 | aperture | 10 | 4 | 25 |
| 4 | down-hill | 7 | 5 | 106 |
| 5 | up-hill | 8 | 4 | 97 |

the different modes and causes of failures the robots experience. Then we show how some of these metrics can be plotted for an approach which monitors the condition of a navigating robot. Finally, we analyze the merits and faults in our approach to testing and how it can be improved.

A. Offline Results

Offline metrics serve several purposes. First, they can be used for comparison with other more rigorous testing methods. These metrics also show how consistently the robot performs in an environment when presented with different challenges and serve as a measure of nominal performance while monitoring a robotic system online.

The reference experiment serves a similar function except that it does so for the offline metrics. Since the reference mission uses waypoints without associated challenges it acts as a control for the other missions with challenges. Another useful aspect of the reference experiment is that it can provide information about how challenging the given environment is in relation to other test environments.

Our method does not use an external ground truth system, but instead draws quantitative conclusions regarding the performance of the robot. The ratio of time moving forward to time stopped increases in terrain where the map is more clear and planning is easier. This ratio goes down in the more cluttered environment where the robot has to stop and replan as new obstacles obstruct the global plan. The difference in time moving forward to time stopped ratio between the different platforms also results from map building. The PackBot mapping system in this experiment had a more conservative model of the obstacles in the environment. As a result, it spent more time stopping and replanning.

These sorts of phenomena controlled other values in the same manner as average velocities. Clutter in the environment made it so that robots took longer paths around obstacles which were not dangerous. This increased the path distance. The Avg. Trial Time given Failure is an indication of at what time the robotic system will fail if it is going to fail at all. In this reference experiment both the SPEAR and PackBot are likely to fail at the beginning of the experiment. This suggests that initializing the system and localizing within the map is the most difficult part of the reference mission.

B. Monitoring Results

As discussed previously, monitoring consists of tracking the velocity of the platform with respect to the next way-

TABLE II
OFFLINE METRICS

| Test Number | Platform | Time Moving Forward / Time Stopped | Time Moving Backward / Time Stopped | Avg. Vel. forward | Avg. Vel. backward | Actual Dist./Ideal Dist. (Success only) | Avg. Trial Time (s) (Failure only) |
|-------------|----------|------------------------------------|-------------------------------------|-------------------|--------------------|-----------------------------------------|------------------------------------|
| R | SPEAR | 14.1 | 0.32 | 0.79 | -0.14 | +19% | 24.9 |
| R | PackBot | 3.5 | 0.15 | 0.90 | -0.39 | +29% | 115.0 |
| 1 | SPEAR | 4.2 | 0.43 | 0.62 | -0.18 | +52% | 97.5 |
| 1 | PackBot | 2.6 | 0.35 | 0.79 | -0.37 | - | 97.5 |
| 2 | SPEAR | 8.1 | 0.67 | 0.72 | -0.32 | +48% | 111 |
| 2 | PackBot | 3.5 | 0.47 | 0.80 | -0.48 | +40% | 89.5 |
| 3 | SPEAR | 5.9 | 0.30 | 0.73 | -0.13 | - | 92.8 |
| 3 | PackBot | 2.8 | 0.18 | 0.80 | -0.39 | +122% | 108 |
| 4 | SPEAR | 14.4 | 0.66 | 0.70 | -0.14 | +30% | 177.4 |
| 4 | PackBot | 3.4 | 0.31 | 0.35 | -0.38 | - | 93.2 |
| 5 | SPEAR | 16.5 | 1.04 | 0.69 | -0.15 | +0.2% | - |
| 5 | PackBot | 5.7 | 0.35 | 0.70 | -0.28 | +18% | 76.2 |



Fig. 6. Successful execution of Mission 1: Overhead view, starting from first orange waypoint near the "1" at the top of the figure and ending successfully at last orange waypoint at the bottom of the figure. The surveyed blue path between points is offset to show the actual path in red and the planned path in green.

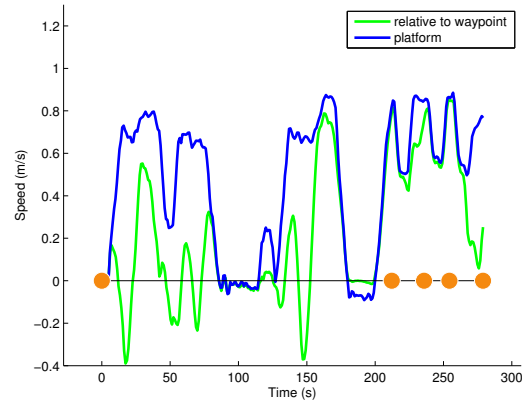


Fig. 7. Successful execution of Mission 1: The velocity of the robot relative to the waypoint is plotted in green, while the platform velocity is plotted in blue. The time the robot arrives at a waypoint is shown with an orange circle. Here, the robot makes more progress towards the waypoint when the difference between the relative and platform velocities is small.

point and platform velocity. In the following figures, these velocities are averaged over a 10 second window to improve readability.

One example is of success in figures 6 and 7. The other is of a failure in figures 8 and 9. In figures 6 and 8 a satellite view of the map is overlaid with a cost-map designating obstacle and clear cells of the environment. The green line is the global plan of the robot and the red line is the trajectory the robot followed. The course color lines reflect the mission of the same color and number from figure 5. Orange circles are the waypoints. The corresponding waypoints are shown on the x-axis for reference in the monitoring output in figures 7 and 9.

When looking at the result of the successful trial (figures 6 and 7) there are a couple things to notice. There are instances of good and poor performance. Different velocities with respect to the next waypoint can indicate different levels of performance in relation to the platform's velocity.

At the beginning of the trial, the robot starts at the initial

waypoint (waypoint nearest the trial number marker). The robot drives in and out of a courtyard which does not bring it closer to its goal in figure 6. In figure 7 this is where the robot has positive and negative velocities in relation to the goal. The integral of these velocities would show that the robot has not gotten closer to the goal as long as it was in the courtyard. While not moving closer to the goal might be indicative of poor performance, it could also indicate a complex environment where the robot has to maneuver through a set of obstacles which require it to drive away from the goal in order to follow a planned path.

Having a forward velocity near the reference average forward velocity is a sign of good performance. Constant forward velocity means that the robot is getting more information about the environment and this movement better informs the mapping and navigation. Even if the robot is not moving towards its goal, it is gaining information so that it can plan a successful path to its goal.

Very poor performance is where there is little or no velocity relative to the next waypoint and the robot is not moving.



Fig. 8. Failed execution of Mission 4: Overhead view, starting from first orange waypoint near the "4" at the bottom of the figure and ending at last orange waypoint at the top of the figure. The surveyed cyan path is offset to show the actual path in red and the planned path in green. The trial ends with the robot driving into a wall.

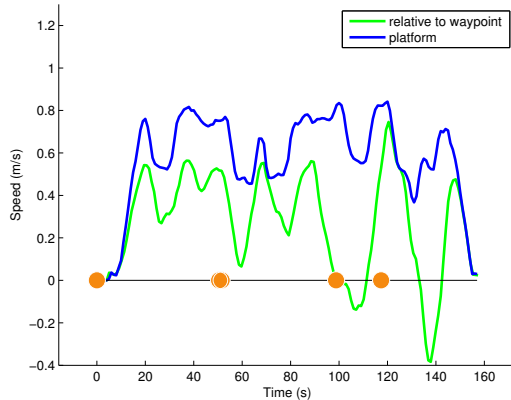


Fig. 9. Failed execution of Mission 4: The velocity of the robot relative to the waypoint is plotted in green, while the platform velocity is plotted in blue. The time the robot arrives at a waypoint is shown with an orange circle. Here, the robot initially appears to be performing well as the difference between the relative and platform velocities is not large, however after the waypoint at time 120s the robot does not maintain this level of performance.

This indicates that there is a software or hardware problem. Around the 175-200 second mark, the robot becomes high-centered on a sewer-grate which is 6 inches higher than the ground, but obscured by grass. As a result little progress is made. In this situation the robot is attempting to maneuver off of the sewer-grate but makes slow progress. In other trials, the robot was not able to maneuver of the sewer-grate and the trial ended in failure. After leaving the robot leaves the sewer-grate, there is a period of high performance until the last goal is achieved. In this high performance situation, the velocity of the platform is high and closely matches the velocity with respect to the next waypoint.

Elements of good and poor performance can be seen in the failure of a trial in challenge mission 4. In this trial the same indicators of good and poor performance are true. At the start, there is relatively little clutter and the robot is able

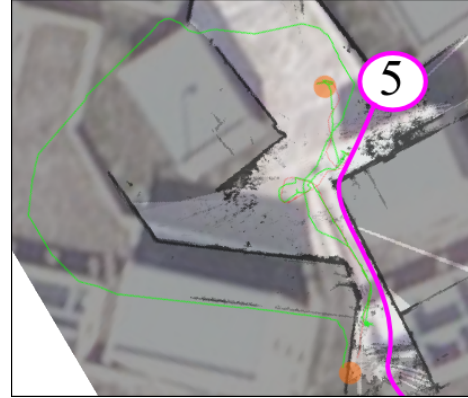


Fig. 10. Failed execution due to GPS: Here, the high level planner attempts to find a plan to orange waypoint which appears inside the wall of building. This leads to a failure from the global planner.

to quickly make it to the penultimate waypoint. Around this waypoint, the robot is not able to follow the global plan and crashes into a wall. The poor performance is visible in figure 9 as the velocity with respect to the next waypoint oscillates around zero. While this oscillation did not indicate failure, in figure 7, it is an indicator of poor performance.

C. Validity of Integrated Ground Truth

While monitoring does yield some meaningful performance metrics, it is not always possible to know if poor performance is going to lead to failure. Another issue is that imprecise ground truth through GPS can cause problems. GPS is not precise. This creates situations where the GPS tells the robot it has reached a waypoint and it has not. It also creates an error where GPS waypoint are assigned in an incorrect place, like inside an obstacle. An example of this can be seen in figure 10. Over the course of our trials there were 7% where poor GPS caused problems with the trial. This number seems relatively small in relation to the total number of experiments performed, but is still the main drawback of this monitoring approach.

VI. CONCLUSIONS

We present a methodology and initial metrics which begin to address the task of monitoring a robotic system executing autonomous navigation in an unstructured environment. Our approach is founded on existing work in the realm of benchmarking and monitoring. We then validate our approach by conducting a series of experiments in a realistic, outdoor environment. The ideal path data indicates performance for a mission and the reference missions helped determine the nominal performance of the robots in this particular environment. The metrics that we propose enabled qualitative conclusions regarding the measure of performance for each robot tested. In addition, monitoring the online velocities proved informative as this information can be used to characterize current performance.

There are a number of ways which this research might be extended. First, it would be advantageous to conduct

experiments at different sites to see how performance varies. Establishing additional metrics that correlate to performance based on obstacle distance or terrain difficulty would make performance more predictive. Using a support vector machine to partition successful and failure level performance would also be useful in anticipating failure due to poor performance. Predicting waypoint arrival time based on past and current performance may help autonomy programs and operators determine how to schedule their tasks. This same tracking of current and past performance might inform what the likely future performance is. An autonomous robot which is able to use this monitoring data might be able to recognize poor performance and change navigation strategies or ask for human assistance. Another useful tool could be the ability to define an area in the environment that challenges a robotic system. This insight could be used for further investigation or be marked as dangerous for other robotic-systems operating in the same areas.

REFERENCES

- [1] "ASUS Xtion Pro Live," http://www.asus.com/us/Multimedia/Xtion_PRO_LIVE/. [Online]. Available: http://www.asus.com/us/Multimedia/Xtion_PRO_LIVE/
- [2] "Garmin GPS," <https://buy.garmin.com/en-US/US/oem/sensors-and-boards/gps-18x-oem/prod27594.html>. [Online]. Available: <https://buy.garmin.com/en-US/US/oem/sensors-and-boards/gps-18x-oem/prod27594.html>
- [3] "Hokuyo LiDAR," <http://www.hokuyo-aut.jp/02sensor/07scanner/download/products/utm-30lx-ew/>. [Online]. Available: <http://www.hokuyo-aut.jp/02sensor/07scanner/download/products/utm-30lx-ew/>
- [4] "MicroStrain IMU," <http://www.microstrain.com/inertial/3DM-GX3-25>. [Online]. Available: <http://www.microstrain.com/inertial/3DM-GX3-25>
- [5] "PackBot," <http://www.irobot.com/For-Defense-and-Security/Robots/510-PackBot.aspx>.
- [6] F. Amigoni, V. Schiaffonati, and M. Verdicchio, "Methods and Experimental Techniques in Computer Engineering," pp. 55–68, 2014. [Online]. Available: <http://link.springer.com/10.1007/978-3-319-00272-9>
- [7] J. Carlson, R. R. Murphy, and A. Nelson, "Follow-up Analysis of Mobile Robot Failures," *Robotics and Automation, 2004. Proceedings. IEEE International Conference on*, no. April, pp. 4987–4994, 2004.
- [8] F. Dellaert and M. Kaess, "Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing," *The International Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, Dec. 2006. [Online]. Available: <http://ijr.sagepub.com/cgi/doi/10.1177/02783649060072768>
- [9] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," GT RIM, Tech. Rep. September, 2012. [Online]. Available: <http://smartech.gatech.edu/handle/1853/45226>
- [10] J. Gregory, J. Fink, E. Stump, J. Twigg, J. Rogers, D. Baran, N. Fung, and S. Young, "Application of Multi-Robot Systems to Disaster-Relief Scenarios with Limited Communication," in *10th Conference on Field and Service Robotics (FSR)*, 2015.
- [11] D. Holz, L. Iocchi, and T. van der Zant, "Benchmarking intelligent service robots through scientific competitions: The robocup@home approach," in *AAAI Spring Symposium: Designing Intelligent Robots*, 2013. [Online]. Available: <http://www.dis.uniroma1.it/~iocchi/publications/AtHome-AAAISS13.pdf>
- [12] J. How, B. Bethke, a. Frank, D. Dale, and J. Vian, "Real-time indoor autonomous vehicle test environment," *Control Systems, IEEE*, vol. 28, no. 2, pp. 51–64, 2008.
- [13] A. Jacof, E. Messina, H.-M. Huang, A. Virts, A. Downs, R. Norcross, and R. Sheh, "Guide for Evaluating , Purchasing , and Training with Response Robots Table of Contents," *ASTM International Standards Committee on Homeland Security Applications; Operational Equipment; Robots (E54.08.01)*, pp. 1–40, 2013.
- [14] M. Likhachev, "Search-Based Planning Library," <https://github.com/sbpl/sbpl>. [Online]. Available: <https://github.com/sbpl/sbpl>
- [15] P. Lin, G. Bekey, and K. Abney, "Autonomous Military Robotics: Risk, Ethics, and Design," *Design*, p. 108, 2008. [Online]. Available: http://ethics.calpoly.edu/ONR/_report.pdf
- [16] R. R. Lindemann and C. C. Voorhees, "Mars Exploration Rover Mobility Assembly Design, Test and Performance," *2005 IEEE International Conference on Systems, Man and Cybernetics*, vol. 1, pp. 450–455, 2005. [Online]. Available: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1571187>
- [17] O. Michel, F. Rohrer, and Y. Bourquin, "Rat's life: A cognitive robotics benchmark," *Springer Tracts in Advanced Robotics*, vol. 44, pp. 223–232, 2008.
- [18] R. R. Murphy, "Rescue robotics for homeland security," *Communications of the ACM*, vol. 47, no. 3, p. 66, 2004.
- [19] R. R. Murphy, S. Tadokoro, D. Nardi, A. Jacoff, P. Fiorini, H. Choset, and A. M. Erkmén, "Search and R 50," *Search*, pp. 1151–1173, 2006.
- [20] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. B. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *International Conference on Robotics and Automation*, ser. Open-Source Software workshop, 2009.
- [21] J. Rogers, J. Fink, and E. Stump, "Mapping with a ground robot in GPS denied and degraded environments," in *American Control Conference*, 2014. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6859100
- [22] B. Ross, J. Bares, D. Stager, L. Jackel, and M. Perschbacher, "An advanced teleoperation testbed," *Field and Service Robotics*, pp. 297–304, 2008. [Online]. Available: <http://www.springerlink.com/index/n262124554525058.pdf>
- [23] S. Schwertfeger and A. Birk, "Map evaluation using matched topology graphs," *Autonomous Robots*, vol. 40, pp. 761–787, 2015.
- [24] A. V. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Robotics: Science and Systems*, 2009.
- [25] B. Shrewsbury, Z. Henkel, C. Y. Kim, and R. R. Murphy, "RESPOND-R Test Instrument," 2013.
- [26] C. Sprunk, J. Roweckamper, G. Parent, L. Spinello, G. Tipaldi, W. Burgard, and M. Jalobeanu, "An Experimental Protocol for Benchmarking Robotic Indoor Navigation," in *International Symposium on Experimental Robotics (ISER)*, 2014.
- [27] A. Stentz, J. Bares, and T. Pilarski, "The crusher system for autonomous navigation," *AUVSs Unmanned Systems*, no. 2, pp. 31–36, 2007. [Online]. Available: <http://www.frc.ri.cmu.edu/~axs/doc/auvsi07.pdf>
- [28] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 573–580, 2012.
- [29] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, and Pasc, "Stanley: The Robot that Won the DARPA Grand Challenge," *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.