Special Issue: DARPA Subterranean (SubT) Challenge

Regular Article

System for multi-robotic exploration of underground environments CTU-CRAS-NORLAB in the DARPA Subterranean Challenge

Tomáš Rouček[1], Martin Pecka[1], Petr Čížek[1], Tomáš Petříček[1], Jan Bayer[1], Vojtěch Šalanský[1], Teymur Azayev[1], Daniel Heřt[1], Matěj Petrlík[1], Tomáš Báča[1], Vojtěch Spurný[1], Vít Krátký[1], Pavel Petráček[1], Dominic Baril[2], Maxime Vaidis[2], Vladimír Kubelka[2], François Pomerleau[2], Jan Faigl[1], Karel Zimmermann[1], Martin Saska[1], Tomáš Svoboda[1] and Tomáš Krajník[1]

[1]Faculty of Electrical Engineering, Czech Technical University
[2]Université Laval, Canada

Abstract: We present a field report of the CTU-CRAS-NORLAB team from the Subterranean Challenge (SubT) organized by the Defense Advanced Research Projects Agency (DARPA). The contest seeks to advance technologies that would improve the safety and efficiency of search-and-rescue operations in GPS-denied environments. During the contest rounds, teams of mobile robots have to find specific objects while operating in environments with limited radio communication, e.g., mining tunnels, underground stations or natural caverns. We present a heterogeneous exploration robotic system of the CTU-CRAS-NORLAB team, which achieved the third rank at the SubT Tunnel and Urban Circuit rounds and surpassed the performance of all other non-DARPA-funded teams. The field report describes the team's hardware, sensors, algorithms and strategies, and discusses the lessons learned by participating at the DARPA SubT contest.

Keywords: subterranean robotics, GPS-denied operation, robot teaming, emergency response

1. Introduction

Recent advances of artificial intelligence, machine perception, environmental mapping, localization in combination with the availability of computational power have opened new possibilities to move robotics closer to the holy grail of achieving autonomous operation in hazardous environments [Atkeson et al., 2018]. However, similar to other fields of science, robotics is facing a crisis

Received: 15 January 2021; revised: 8 May 2021; accepted: 28 June 2021; published: 18 August 2022.

Correspondence: Tomáš Rouček, Faculty of Electrical Engineering, Czech Technical University,
Email: roucek.tomas@fel.cvut.cz

DOI: https://doi.org/10.55417/fr.2022055

Figure 1. Team of ground robots exploring entry hallway (left) and crashed UAV being observed by hexapod (right) during DARPA SubT at Satsop nuclear powerplant.

of research reproducibility. The first cause is the cost of experiments, which quickly rises with the amount of logistic and technical issues which need to be solved for any field experiments. This also has to include new or destroyed equipment which happens rather often such as seen in Figure 1. The high cost of field experiments causes the evaluations to be performed on datasets which were often gathered for a different purpose, thus making them either not sophisticated enough or not able to capture uncertainty and unpredictability of real scenarios. Another reason is the complexity of the systems where most robots are comprized of a large number of smaller submodules from which are usually tested separately. Thus, method interoperability, compatibility, and their impact on the efficiency of the whole system often remain neglected. The last reason is that failures of experiments are often blamed on technical issues, and the reliability and robustness of the methods are not reported. Instead, scientific papers focus on issues of accuracy or computational complexity of the individual methods rather than the reliability of the integrated systems [Lier et al., 2016, Dillmann, 2004].

With all the mentioned problems, the performance of robotic systems cannot be optimal in real-world situations which are impacted by the robustness of the deployed systems. Robotic contests offer a potential solution to the problem. The results of many of these contests, such as MiroSot Leage [Wong et al., 2005], Eurobot [Obdrzalek, 2010], RoboTour [Iša and Dlouhý, 2010], RockIn [Schneider et al., 2014] or MBZIRC [Spurný et al., 2018], show that the success depends more on reliability and interoperability than other aspects.

One of the active fields of research is autonomous exploration, where a team of robots is supposed to build a complete and accurate spatial [Burgard et al., 2000] or spatiotemporal [Santos et al., 2016] map of its environment. Autonomous exploration is challenging not only because it encompasses self-localization, map building, path planning and motion control, but mainly because the robot has to operate in previously unknown areas. Even though most of the problems mentioned above are sufficiently tackled separately, the number of systems that can achieve a reliable exploration of truly unknown environments is low [Santos et al., 2016].

The need to quickly explore, find and locate objects in real-world environments is motivated by search and rescue (SAR) operations, where the time required to find survivors or environmental hazards is critical. The use of robotic technologies, which offer the ability to create detailed maps with positions of potential victims and hazardous areas cannot only speed up the SAR operations but also make them safer for the rescue teams. Moreover, robots can be used to search in locations inaccessible

by humans, such as gas-filled tunnels, unstable structures, large crevices, or contaminated areas. The ability to create accurate maps can also mitigate the danger of mining disasters by preventing accidental breaches into abandoned tunnels that might be flooded or otherwise contaminated [Huber and Vandapel, 2006].

Search-and-rescue missions often have the robots controlled manually via a radio link since their ability to operate semiautonomously is severely limited. Even modern rescue robots offer only limited features such as intelligent camera switching or automated stop at negative obstacles. The lack of autonomy means that a single operator can control only one robot at a time and teleoperating the robot imposes a significant cognitive load. Moreover, the primary sensory modality for teleoperation is video, which requires a high bandwidth, which, in turn, limits the number of robots deployable in the same mission. Furthermore, maintaining a long-range, high-quality radio link between a control station and a robot that moves indoors or underground is challenging at the least. A solution to the problem is to endow the robots with the ability to operate in a semiautonomous manner, process the bulk of the sensory data onboard and use the radio link to report only the data that are necessary for efficient and safe mission control. The problem of underground exploration is addressed by the Defense Advanced Research Projects Agency (DARPA) Subterranean (SubT) Challenge, which offers an ideal situation to test the robustness and reliability of robotic systems in search and rescue scenarios.

The DARPA SubT Challenge is organized not only to fund but also motivate the development of robotics systems that are capable of supporting search and rescue operations without any supporting infrastructure in the underground of postdisaster environments. The contest emphasizes the reliability and efficiency of complete integrated systems rather than the efficiency of the individual modules and components. In particular, the performance of robotic teams is evaluated by their ability to quickly and accurately locate relevant objects in underground sites with a variable degree of complexity.

In this field report, we describe the multi-robot system developed for the DARPA SubT challenge by the team CTU-CRAS-Norlab of the Czech Technical University in Prague and Université Laval. Later on, we describe more details about the hardware of the robots, communication solution, localization systems, mapping, navigation, as well as our approach to multi-robot coordination.

2. Related work

Although there are multiple stages of the disaster management cycle [Murphy, 2014], our primary focus is on the first one: "rescue activities during or in the immediate aftermath of a disaster to save lives". Robotic systems were used in several first stage scenarios such as searching for survivors after the September 11, 2001 attacks, or Hurricane Katrina in 2005. However, the majority of such deployments was teleoperated [Kruijff et al., 2014]. The reason is that the robustness and explainability of the high-level autonomy methods are still far from being acceptable by rescuers.

However, adverse environmental factors in underground environments negatively impact not only the cameras but also other sensors used for teleoperation. Additional factors like radio bandwidth limitations, data outages and delays, and the need for fast mission execution imposes significant stress onto the robot operator, which can result in performance deterioration with severe consequences. These factors virtually prohibit the possibility of operating larger teams of robots simultaneously. This limitation motivated the research in autonomous and semiautonomous exploration of environments with conditions adverse to radio communication and led to the organization of the SubT Challenge.

The work of [Murphy et al., 2009] summarized the impact of mine accidents and stressed the potentials of deploying robots in underground search and rescue operations. Later works, e.g., [Neumann et al., 2014] investigated the use of three-dimensional (3D) laser scanners for precise mapping, and others demonstrated the feasibility of systems that can venture hundreds of meters into abandoned mines autonomously. Automated systems capable of creating 3D maps

of underground environments were presented several years ago, e.g., in [Thrun et al., 2003] and [Huber and Vandapel, 2006]. While both of these systems could create detailed and extensive 3D maps, the robots were controlled manually, and neither autonomous exploration nor object detection was considered. Shortly after these proof-of-concept systems, more authors addressed the problem of underground mapping. An introduction to the problem of underground mapping along with a comparison of underground mapping systems was provided in [Morris et al., 2006]. [Morris et al., 2006] conclude that the sensor fidelity, data processing methods and mapping algorithms were not ready to constitute a reliable system capable of autonomous exploration and 3D mapping.

However, the need for reliability of underground operation requires robust solutions capable of withstanding explosions and fire [Wang et al., 2014] as well as operation in flooded mines [Maity et al., 2013]. These often rely on preinstalled infrastructures like ceiling lights, radio beacons or magnetic wires in combination with reactive navigation behaviours [Rusu et al., 2011].

However, development of low-weight, high-performance computers and solid-state lidars as well as progress in localization, mapping and learning algorithms allowed to integrate various degrees of autonomy into robotic systems. Many of the works focused on mobility over adverse terrain, e.g., the authors of [Zhao et al., 2008] proposed to use a marsupial setup consisting of a "mother" and "baby" robot, which is released to map hard-to-access areas. Domain transfer techniques such as generative adversarial networks (GANs) simplified transfer from the simulator to real platforms [Pecka et al., 2018a] allowing for autonomous control of flippers on skid-steer robots. Moreover, significant progress in reinforcement learning and trajectory optimization techniques allowed for robust control of legged platforms like ANYmal [Hwangbo et al., 2017]. Reduced dimensions of sensors and computational hardware allowed to deploy real-time simultaneous localization and mapping (SLAM) in aerial platforms, making them capable of operating in GPS-denied environment for SAR scenarios [Faessler et al., 2016, Petrlík et al., 2020].

The availability of off-the-shelf GPUs allowed for real-time processing of high-resolution images by deep convolution networks [Redmon et al., 2016], which are known to achieve human-comparable accuracy in many tasks such as object recognition, which is vital for visual based SAR.

DARPA's SubT directly ties back to most requirements in search and rescue scenarios and motivates multiple teams to perform their best. For the Tunnel and Urban circuit of the challenge, each team brought its approach to the problem. Team Explorer utilizes custom unmanned aerial vehicles (UAVs) and unmanned ground vehicles (UGVs) with great terrain handling capabilities due to their size, which allows them to efficiently drive through uneven terrain where smaller robots would struggle. The bulk of the robots allows them to carry and drop many communication nodes throughout the course, ensuring robust communication links during the deployment [Tatum, 2020]. The teams experience with underground exploration robust platforms allowed them to win the first Tunnel circuit.

Team CoSTAR focuses on their Networked Belief-aware Perceptual Autonomy system, which takes into account the uncertainty of all possible parts and is designed to perform long tasks outside of the range of communications. In [Bouman et al., 2020] they focus on new Boston Dynamics platforms, which with the help of two Husky platforms, won the entire Urban circuit. Legged platforms were deployed earlier in the Tunnel circuit by Pluto [Miller et al., 2020] and Cerberus [Bjelonic et al., 2020] where the legged platforms did not perform well due to slippery surfaces even when equipped with wheels to serve as walking/rolling platform. Other rather unconventional methods of locomotion were explored by team NCTUs [Huang et al., 2019], which deployed a blimp.

2.1. Contributions

This paper provides an overview of a system utilized by CTU-CRAS-NORLAB during the SubT Challange. This system builds upon the experiences of the aforementioned works and on the lessons learned during projects aimed at robotic search and rescue missions [Kruijff-Korbayová et al., 2015]. Specifically we provide our contributions as

Table 1. Summary of DARPA organised events for SubT with their environments and artifacts for each round. Artifacts from SubT Integration Exercise (STIX) are common for all Circuits [DARPA, 2019].

Event	Date	Environment	Artifacts
STIX Exercise	04/2019	Silver Mine	Backpack, Survivor, Cellphone
Tunnel Circuit	08/2019	Coal Mine	AKU Drill, Extinguisher
Urban Circuit	02/2020	Nuclear Plant	CO2 Elevation, Duct Vents
Cave Circuit	Canceled	Natural Cave	Rope, Helmet
Final Event	09/2021	Louisville Cavern	All above, Translucent SubT Logo

- Robust multirobot system with redundant communications (Sections 5 and 4)
- Visual, radio and olfactory based object localisation (Section 5.4)
- ICP based localization and mapping (Section 5.5)
- Adaptive traversability (Section 5.6)
- Elevation mapping and entropy based exploration (Section 5.7)

3. Contest specification

The Subterranean Challenge (SubT) is a contest organized by DARPA. In the SubT Challenge, a team of mobile robots has to actively search an environment to locate and report the positions and types of specific artifacts in a previously unknown, subterranean environment with limited human interaction. The performance of the robotic team is evaluated by the number of objects detected, and the accuracy of their position estimation [DARPA, 2019]. All participants have to adhere to the same rules, which are evolving throughout the whole competition, based on previous observations or sites where each circuit takes place. Moreover, running in parallel is a virtual component of the same challenge, which takes place solely in a simulation.

3.1. Competition rules

With limited time (usually 30 minutes) and personnel (maximum 10 people), each team has to prepare and then deploy a robotic system to explore a previously unknown environment inaccessible to the personnel. The robotic fleet has a mission to find, locate, and then report back to base the position of the previously given objects called artifacts within a limited time (usually 1 hour). While it is not required by the rules for a system to be fully autonomous, only one person is allowed to control any aspect of the mission and view the data reported back by the fleet.

3.1.1. Score

All artifacts that have to be found change for each circuit and are specified by DARPA beforehand. For a team to obtain a point, it is required to

- find and recognize the correct type of artifacts,
- localize the artifact with a maximum of 5 m of euclidian distance from the ground truth in the global coordinates,
- and report the correct type and position back to home base.

There are 20 artifacts on the course. Each team can send 40 reports during one run, making it not possible to brute force the problem. Additionally, each artifact has to be reported only once, otherwise a point is subtracted from the team. If any teams would have the same number of points, other factors such as how fast the artifacts were reported or how far teams traveled are then used as a tie breaker. More information can be found on the DARPA SubT website [DARPA, 2019].

Table 2. Hardware information and payload of each robot type.

Robot	Husky A200	Absolem	Hexapod	UAV
Movement	Wheeled	Tracks	Legged(6)	Quadrotor
Speed [m/s]	1	0.4	0.2	1
Traversability	Low	Medium	Medium	High
Computation power	High	High	Low	Medium
Max. payload [kg]	50	10	1	0.4
Runtime	1.5h	1.5h	1h	10-20m (payload dependent)
Communications	All	All	Mote	Wifi,Mote
STIX	0	2	0	2
Tunnel	1	2	2	2
Urban	1	3	1	2

3.2. Contest Challenges

The whole Challenge is split into four consecutive rounds, called "circuits" where each round takes place in a specific domain (such as "Tunnels", "Urban" and "Caves") and combines them all in the final round. Each of these environments is built to provide various challenges and serves as an incubator for new technologies and approaches in several robotics problems such as limited communications, traversability, multiagent integration, sensing, etc.

4. Robots and Sensory Equipment

Robots utilized by our team include wheeled and tracked UGVs, six-legged spiderlike robots, and UAVs. This usage of several types of robots is crucial since each platform has different capabilities. Larger robots with wheels are able to traverse vast distances since they are the fastest platform even compared to tracked robots which are capable of traversing difficult terrain where the wheeled robots would get stuck. The legged robots and quadrotors do not carry the same equipment as larger robots but can access narrow tunnels, elevated vantage points, and places that would otherwise be inaccessible by larger robots. The specific equipment carried by each platform type varied throughout the development but the general types of sensory and computation equipment stayed mostly the same, see Table 2.

4.1. Wheeled robot - Husky A200

The Husky A200 from Clearpath Robotics is a differentially driven wheeled platform. The platform is built to be rugged and capable of traversing mud, gravel, light rocks, and steep declines/inclines. The onboard sensors changed between circuits, consisting of lidar with a 360° field of view and a range up to 200 m, which is the primary sensor for localization and mapping. In the "Tunnel" circuit we used a Robosense lidar with 32 lines, and in the "Urban" circuit we used a Robosense 3D lidar with 16 lines. To detect objects of interest, the Husky is equipped with 5 Bluefox RGB cameras positioned to achieve a 360° field of view and one Bluefox camera to look upwards. All the cameras are running at a reduced 15 FPS with extended exposure to allow for low light operations. The camera feed is then passed to the onboard NVidia Jetson TX2, which is used for object detection. Calculations for localization, mapping, and control were performed on a separate control PC, Intel NUC5-i5 on the "Tunnel" and Dell OptiPlex 3070 on the "Urban".

Since the payload capacity far exceeds the weight of the sensory and computational equipment, the robot carries some extra devices. In the "Tunnel" circuit, it was equipped with two eight-slot containers for communication relays. In the " Urban" circuit, the design of communication relay containers was changed, so the Husky carried only four relay modules. These relay units are deployed during the mission to extend the range of low-bandwidth communications from the command station [Čížek and Faigl, 2019]. Other possible attachments are docks for legged robots

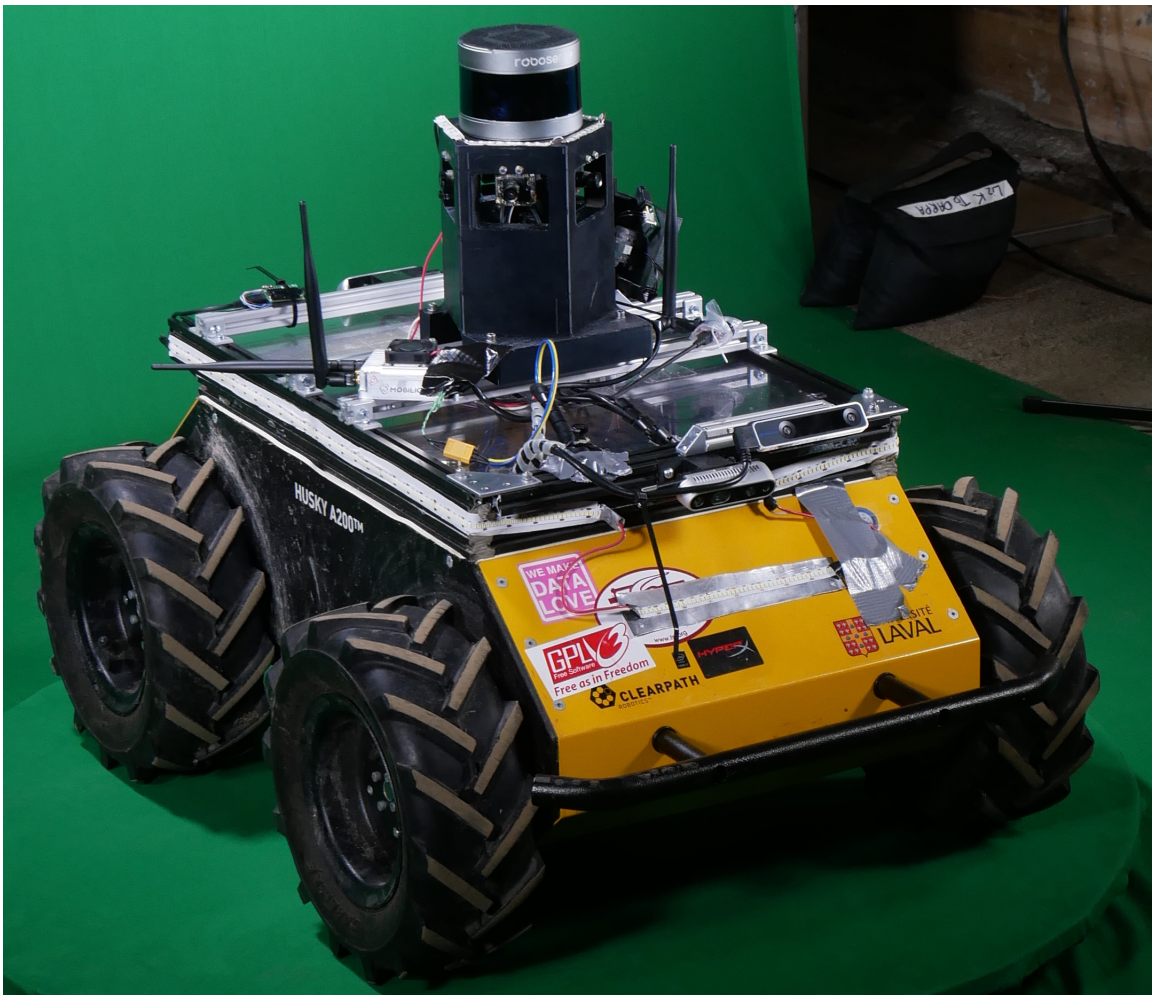


Figure 2. The Husky platform with a Robosense lidar on top of a sensory tower which houses five cameras.

and UAVs, which could be deployed on the most suitable locations for those specific platforms [Saska et al., 2012].

4.2. Tracked robots - Absolem

The three tracked Absolem platforms made by BlueBotics SA are designed to traverse terrain typical of disaster sites. All Absolem robots drive on two main tracks, each having two additional subtracks called flippers, which are controlled independently during the drive of the robot. The flippers are set by a custom controller, which takes the flipper position and data from mounted sensors to assess the best setting for those flippers to help the robot traverse large obstacles such as stairs, curbs, and even gaps larger than the robot itself. The primary sensor for localization and mapping is a SICK LMS-151 line lidar mounted on the robot's front, pivoting around its forward axis. The lidar is attached to a motor that rotates it. This makes the scanning plane sweep the space, making it possible to acquire a 3D point cloud of its surroundings. Processing power is provided by an internal PC with Intel Core i7, which runs localization, mapping, and navigation. PointGrey Ladybug 3 omnicamera (now FLIR Ladybug 3) is used to feed 4-K pictures at 6 FPS to a Jetson TX2 board for object detection and camera feed. As the computational demands of our algorithms increase, for

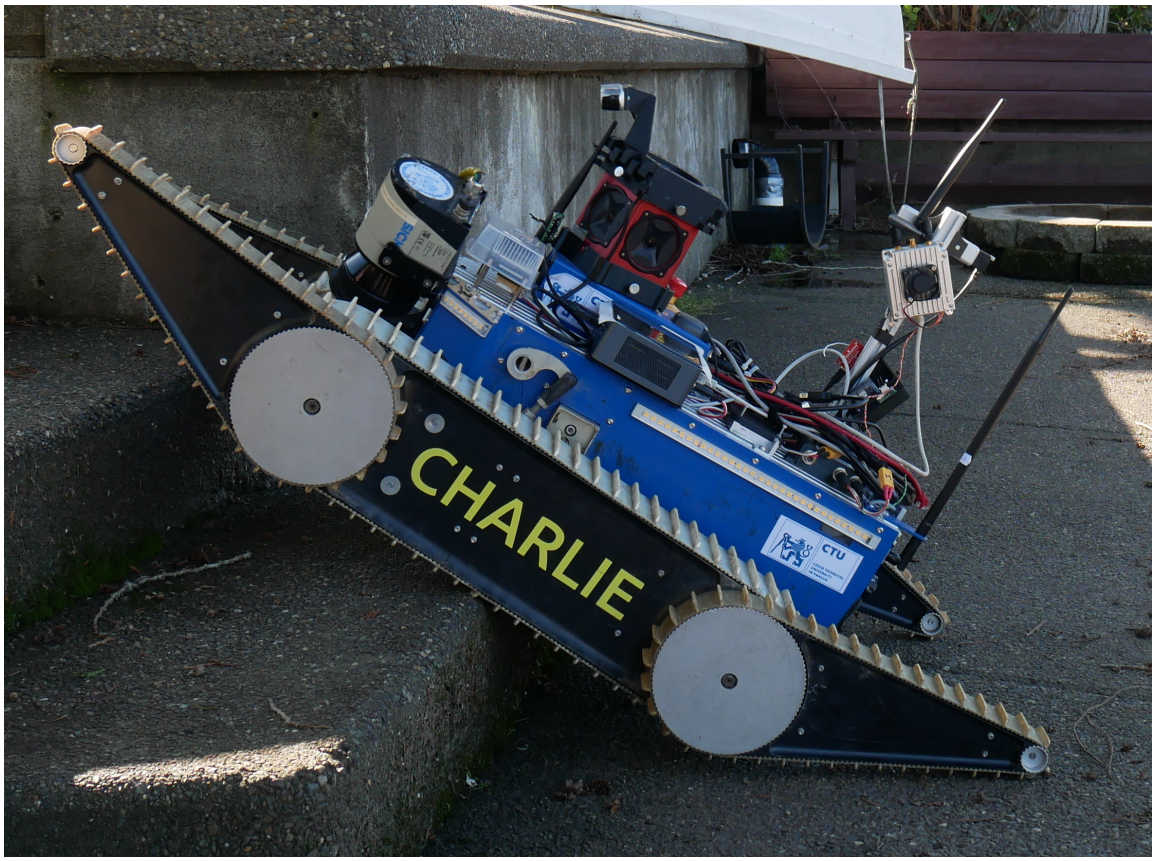


Figure 3. Absolem climbing concrete stairs using extended flippers.

the "Urban" circuit, we have also added an offloading computer (Intel NUC8-i7) which can process the mapping and other CPU-intensive operations. The Absolem robots are supposed to thoroughly search areas not accessible by the Husky robot, e.g., those accessible only by stairs. This is possible due to the tracked design and size, which allows for enough sensors to be mounted to enable the same algorithms to be run for the Husky and the Absolem. This software will be described further on in Section 5.

4.3. Crawling robots - hexapods

The crawling spiderlike robots are derived from the six-legged PhantomX Mark II platform. Navigation, mapping, and localization are done by a custom rig comprised of two Intel Realsense cameras supported by LEDs to compensate for low light conditions. For localization, T265 is used since it can run onboard SLAM, providing an estimate of the robot's actual position. The onboard SLAM makes it possible to lower the computational load of a robot's computer [Bayer and Faigl, 2019]. A D435 RGBD camera does mapping, exploration, and object detection. Computation is done on a NVidia Jetson TX2, which is the only computer on those platforms, mainly because it can perform object detection. The main strong point of the hexapods is an ability to crawl over hard terrain or through vents or ducts that would be otherwise inaccessible to other ground robots [Faigl et al., 2016]. Since their speed is rather slow, they are not much use except for some small extensions as communication nodes. This can be are largely overcome by potentially supporting them with Husky robot to carry them to the places of interest during the competition.

Figure 4. Hexapod in grass "sitting" in power save mode.

4.4. Aerial robots - quadrotors

The aerial robots are based on the F450 kit by DJI, controlled by an onboard Intel® NUC and the PixHawk autopilot. Their primary sensor to perform localization and mapping in the "Tunnel" circuit [Petrlík et al., 2020] was RPLidar A3 2D lidar, which provides 16 000 distance measurements per second in a 360° circle in one plane. The sensor range is 25 m, and its (adjustable) rotation rate is set to 10 Hz, which provides sufficient overview to safely move in narrow tunnels with speeds of up to 1 m/s. Laser-based localization was done by the Hector SLAM [Kohlbrecher et al., 2011] package, which is supposed to work well in conditions that can be anticipated in the subterranean environment [Santos et al., 2013]. In the Urban circuit, we have upgraded to a 3D lidar Ouster OS1-16, which provides 360° scans in 16 beams. This greatly improved the mapping and localization capabilities of the UAVs. For object detection, each UAV carries at least two Bluefox RGB cameras with an LED illumination strip. The detection itself is performed by a custom trained YOLOv2 CNN on the CPU of the onboard i7 NUC PC, which also performs localization and mapping. With this setup, the processing of one image on a single thread takes approximately 1.7 s, which is just sufficient in order not to miss an object when flying at standard velocities. Both UAVs are set up to automatically run the exploration process after launch to be deployed easily by any member of the team.

4.5. Communication

Of the previously mentioned problems, one of the most limiting is the communications constraints. During search and rescue situations, interweaved hallways, underground tunnels, shielded rooms, and thick steel-reinforced concrete walls make it hard or even impossible to maintain any stable radio link. This prevents direct teleoperation by the operator, which otherwise could substitute

Figure 5. One drone platform with propeller guards and Ouster sensor on top.

Table 3. Communication methods comparison.

Type	WiFi	Mobilicom	MOTE
Range	Short	Medium	Large
Fequency	5 GHz	2.3 GHz	900 MHz
Bandwidth	50 Mbit/s	1 Mbit/s	432 bits/s
Size[cm]	Varies	8 x 8 x 3	2x3x0.5 on robot + antenna 4 x 4 x 10 dropped

for complex decision-making algorithms that are required for any autonomy. In the ideal case, the operator would have no constrained link to the robots, which would provide him with all the information about their surroundings. Alternatively, robots would be fully autonomous, requiring no communications eliminating the need for the operator. Autonomy is hard to do reliably, and even fully autonomous robots would have to share some of the information back to the operator, so our team implements several different communication systems to deliver at least some data to the human supervisor of the system. These solutions are all redundant to a certain degree. Losing one of the connections usually does not mean any connection at all but rather an unstable link. This can manifest as lower quality and frame rate of the video, fewer status reports, or just position reporting without any other data. All of them have different levels of reliability, range, bandwidth, and usage when it comes to their specifications. All robots were also equipped with the capability to be connected by wire using ethernet for setup.

4.5.1. Short-range link: WiFi

During testing and usual usage, the robots are connected to standard Wi-Fi which has a bandwidth that allows for the transfer of high data-usage information such as direct sensor measurements or real-time video feed. This link is used during the initial setup of the deployment, where all the robots are in close range of each other since it is necessary to verify readiness for the mission and do the initial setup. Later on, robots use the same Wi-Fi link to send data back to the base station, but the signal deteriorates quickly, and the subsequent midrange communication needs to take over. Additionally, the robots can use the Wi-Fi to share other gathered information such as 3D maps or past positions with others and can use this link in case of failure of the others. When a UAV was sent to the mission without the Mobilicom unit (described below), it used Wi-Fi to connect to the base station when it returned to report its findings.

4.5.2. Mid-range link: Mobilicom

During the actual mission or large-scale testing, special units called Mobilicom are used. These modules use high-power transmission with dual-channel capability and automatic meshing abilities to construct a network that can send data over much larger distances than Wi-Fi. During runs, robots are coordinated to provide retransmission nodes to maintain communication with all the robots at all times. This makes a network, which is able to dynamically switch to the best possible path between each unit, able to send valuable information for most of the structures we have yet tested. Since the data throughput is limited to 1 Mbit/s, rationing this to robots proved to be a challenging task but allowed us to have a low-quality video for teleoperation, basic environment maps, and object detection pictures to be transmitted through the network. In the "Tunnel" circuit, we used the larger MCU-30 Ruggedised units and MCU-200 with higher transmission power as the base. The size of these modules allowed their use on the wheeled and tracked robots only, which limited the system to a communication range of roughly 300 m. In the "Urban" circuit, the smaller MCU-30 Lite units were used, and this allowed the deployment of the communication nodes on all robotic platforms, including UAVs and hexapods, to help extend the network range.

4.5.3. Long-range link: Motes

To have more redundancy and coverage, one more communication module type is used. These small custom-built modules allow for low bandwidth, which is sufficient to share crucial data from the robots, such as their status, position, and possible detections. The data are not shared only to the base station but also to the other robots to enable low-level coordination and control from the base station. These "Mote" modules are carried by all our robots, even UAVs, since they are compact and small. The larger robots can even carry several of these modules and can drop them to create a network of relays, providing communication infrastructure for the other robots. In the "Urban" circuit, due to the code and bandwidth improvement of these modules, we were also able to send commands to the robots from the base station via the Motes. This helped in cases where the robots went out of the range of the Mobilicom nodes, and the operator wanted them to return to the signal.

5. Software

While appropriate hardware is necessary for successful participation at the challenge, it is the software that brings the entire system alive. To stress the importance of the software, DARPA SubT has a parallel "virtual track" where the contest occurs only in software simulations of the deployment sites. While not all the teams participated in the "virtual track", most of them utilized the simulators as an integral part of the development process since tests in a simulation are much less tedious than real-world testing. Moreover, freely available, open-source simulators can be used to evaluate the performance of developed software modules in isolation as well as in a holistic way. The software modules of all robots had to tackle localization, mapping, navigation, object detection, exploration, and multi-robot coordination.

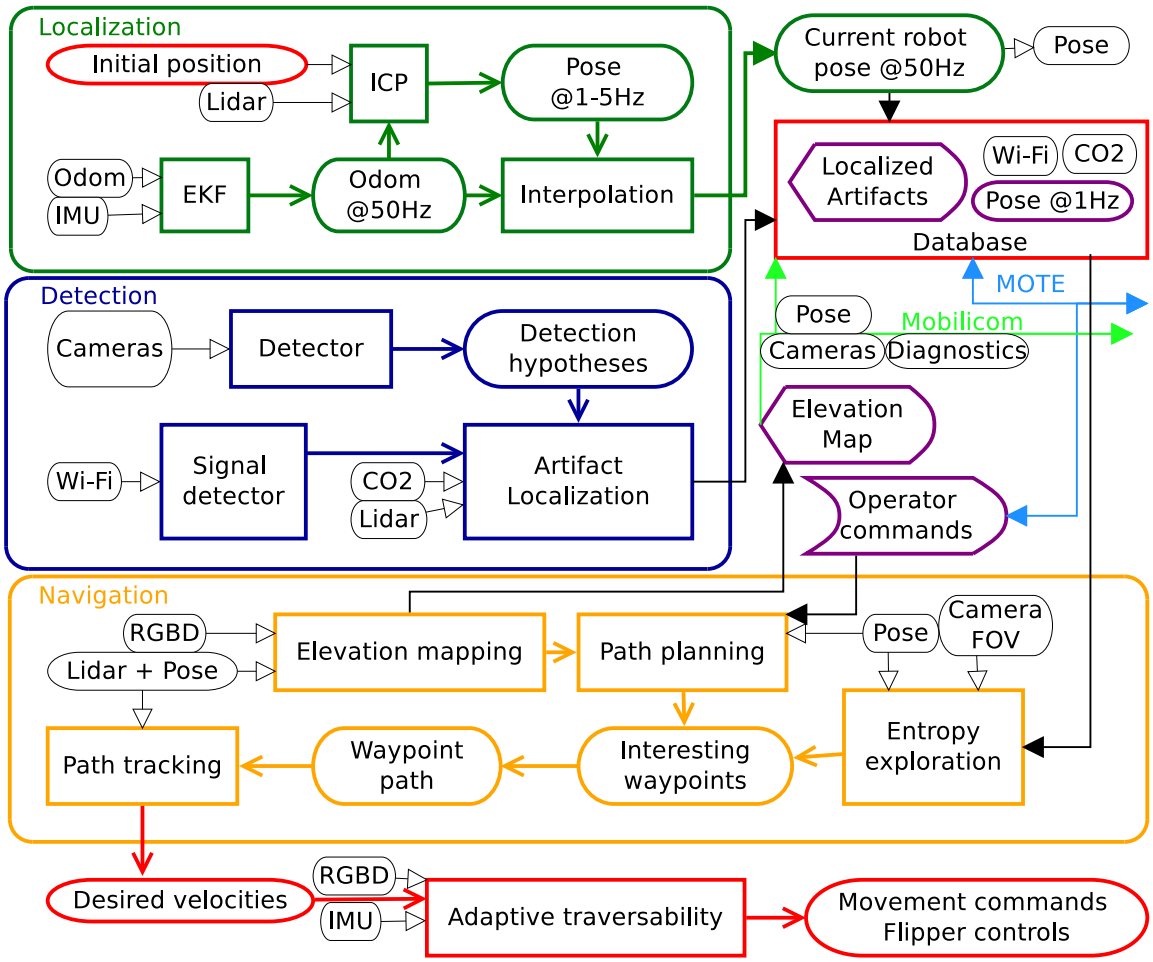


Figure 6. Block diagram of software modules on a single UGV platform with three main subsystems: localization (green), detection (blue) and navigation (yellow). Purple boxes and database indicate the main I/O of a robotic platform that is transmitted to other agents via the MOTE (light blue) and Mobilicom (light green) networks. Internal robot states and sensory information are represented as bubbles while modules are represented as rectangles. The initial position is obtained from the pit crew during setup.

5.1. Software Architecture

All of the systems used in the competition were running the Robot Operating System (ROS) [Quigley et al., 2009] in its Melodic version on Ubuntu 18.04 or its variants (i.e., Linux4Tegra 32 on Jetson TX2 computers). Choosing ROS allowed us the benefit from the multitude of packages developed by the open-source community around ROS; thus, we could concentrate on the software that is specific for our platforms. For example, the whole driver stack of the Husky robot up to EKF-based localization, RGB camera, and lidar drivers was using just the publicly available code from ROS community. The Absolem platforms have a free base drivers (but not open source), but all sensor drivers and a large part of sensor data processing are based on the code provided by the ROS community. The community-provided packages do not, however, have only advantages. ROS does not (yet) provide any terms of software quality measures, so it happens that as we are using a package, we discover it is either buggy or not written in an optimized manner. Thankfully, as all the packages are open source, we can always fix the issues locally, and then offer the fixes to the upstream repositories.

For SLAM, we are using the open-source mapper called "ethzasl_icp_mapper"[1] with customizations done by the mapping experts from our partner Université Laval. Our planning and exploration solution is not open source, and it is built in a way that it can even work without ROS, but of course, it has a ROS interface to be able to communicate with the rest of the software modules.

One of the benefits ROS brings to us is the ease of decision where should some code be running. As mentioned in Section 4, most robots carry two or three computers on board. The various types of algorithms we use and their constant development require high flexibility in terms of computational power, and with ROS, running the CPU- or GPU-intensive programs on different computers is just a matter of a few configuration lines and some one-off provisioning. ROS also provides high-quality visualization tools like RQt and RViz, which we utilized in our system both during development and for the operator console (described further in Section 5.9).

5.2. Network Architecture

To get a resilient and distributed communication architecture where some links may disappear during the mission, we decided not to use the standard single-master mode ROS supports. Instead, we use a multimaster solution based on the "nimbro_network" package with customizations, and also a RocksDB database. Each robot runs a ROS master on its main computer, so on the robot's local network, all computational nodes and other supporting programs are running as in a standard single-master ROS system.

However, to connect with the base station, the multimaster solution allows us to (a) explicitly state, which data can be transmitted via the limited-bandwidth connection, (b) use UDP protocol where appropriate (which helps to prevent network congestion), and (c) be resilient to network failures. As most connections between the robot and the base station are made via UDP protocol, the connection state is not affected by the actual wireless link status – it just transmits data whenever the link is available.

The multimaster solution is only suitable for streaming-type data like camera images or 3D information; for example, the artefact detections and history of the robot's movement need to be also transmitted for the times when the link was not available (after reestablishing it, of course). For this kind of data, all robots and the base station have a RocksDB database where they store the mission-critical data, and there is a replication protocol between each robot and the base station, via which they synchronize their databases. This synchronization assures that if the robot was operating out of the reach of the wireless link, we would not miss any artefacts making him send the collected data back after the link is reestablished. For the future rounds, we would like to implement peer-to-peer database also between the robots, which would allow us to better utilize the wireless network link and make the setup more resilient.

5.3. System resilience

One of the key features of our software is that it is written with resilience in mind. We know that we do not have the human resources to test all software we write. We also know that we work with cutting-edge hardware that is high stressed during the mission, and it either is not tested for our use-case (like the Realsense T265 cameras) or is just prone to fail in many different ways (screws getting loose, cables getting loose, motors getting overloaded, sensors being blinded or misreporting, and many more).

So instead of going the way industry chooses – which means endless tests, verifications and certifications – we try to write the software in a way that allows for failures and tries to mitigate them. If a vital software node crashes, we restart it hoping that it will not crash again (and we investigate the crash cause after the mission). Some nodes (like the SLAM) require some state to be

[1] https://github.com/ethz-asl/ethzasl_icp_mapping

Table 4. Dataset details with the number of annotated objects. Negative labels are the images containing only the background.

Environment	Images	Negative labels	Survivors	Cellphones	Backpacks	Drills	Extinguishers	Vents
Tunnel	10454	2205	1010	1010	1191	2317	1869	0
Urban	10123	2790	1559	1109	1316	12	247	4024

restored when we restart them after a failure, so we log the state to the filesystem and let the nodes read this state and reinitialize them in the last known working state. From the sensor point of view, we try to duplicate some of the mission-critical sensors, or, better, have several different sensing methods for each sensing modality. For example, the flipper control algorithm needs to know the local terrain shape, so it utilizes data from an RGBD camera but also takes data from the slower lidar. When the RGBD camera fails (which it does because it is connected via USB), the algorithm still has at least some data it can work with. One of the best achievements of these resilience techniques was that during the Urban circuit, on of our robots' main computers got completely restarted during the mission after a hard hit, and the operator was able to connect to the robot and get it up and running again in a very short time, and the robot even kept the knowledge about where it was in the map.

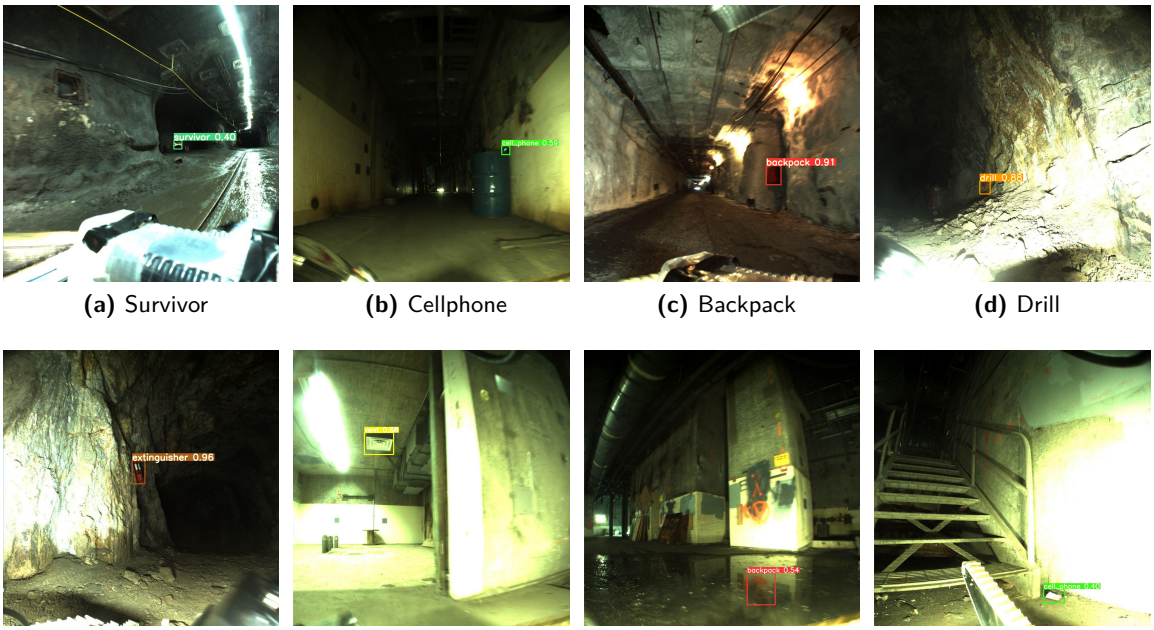
Some parts of the system need a more delicate approach than just restarting when it fails. For example, if the motor that rotates the Sick lidar on Absolem platforms gets stuck, we know that we need to stop the robot immediately; otherwise, the mapping would not recover with the robot going further without any 3D data coming. After we stop the robot, we know we can initiate the motor restart procedure, which hopefully restores the motor function. Another example is the failsafe watchdog, which is running on the robots and watching their pitch and roll angles, and if navigation fails and gets the robot to an unsafe pose, these watchdogs perform recovery actions (either help the robot with flippers or stop it). Knowing that all these automatic recovery procedures are useful, we also know that we cannot anticipate all error conditions and that some will probably falsely trigger these procedures. So we give the operator the right to turn these protections off and try to get the robot to a better state manually.

5.4. Object detection

The purpose of the entire system is to detect and locate artefacts that represent potential victims or provide a cue of their location. Therefore, every robot of our team performs the object detection task. Due to the fact that a color camera is the only sensor common to all of our robots, the base of the object detection is a neural-based computer vision method called YOLOv3 [Redmon et al., 2016, Redmon and Farhadi, 2018].

Since the rules explicitly prohibit training the neural network at the places where the challenge occurs, we gathered our dataset from various underground environments and trained the detector on more than 20 000 images. These environments include university hallways and basement, STIX deployment, the Josef experimental mine, Tunnel deployment, and the Prague subway station. All data was collected though cameras of our robots while other testing was being done on the platforms or through cameras that were temporarily dismantled from the robots. Dataset annotation was done manually. See Table 4 for numbers of annotated objects.

If the robot has a 3D map of its surroundings, the bounding box provided by the YOLOv3 detector is projected into the 3D map. In case that the depth of the object in the image is not known, we save the vector direction between the camera and detected object and estimate the position later from multiple measurements using triangulation. Then, the detections are processed by the final position estimator [Vrba and Saska, 2020], where each new detection creates a new hypothesis (described by state, position estimation covariance, and the number of corresponding measurements) or is assigned to the already existing hypothesis. The final position of the hypotheses is established by the application of Kalman filtering. Only if the hypothesis is certain (covariance lower then threshold



(a) Survivor **(b)** Cellphone **(c)** Backpack **(d)** Drill

(e) Fire extinguisher **(f)** Vent **(g)** False Backpack **(h)** False Cellphone

Figure 7. Detection of all object classes during the challenge. Two false positives are depicted, Backpack as the red sign in water reflection and cellphone as a piece of paper.

and the satisfying number of measurements), the robot sends position, established class and RGB snapshot of the detected object to the database. This prevents flooding of the communication network with image data and helps to lower the number of false positives detections send to the operator. All the objects that were in the field of view of the robots during DARPA SubT scored runs were detected, and the number of false-positive detections was manageable (Table 5), which indicates the good performance of the detection method (see Figure 7 for the qualitative results).

To ease the RGB detections and to improve the chance of finding phones, the Jetson TX2 was utilized to search surrounding for WiFi signal and using weighted trilateration estimate the position of the artefact [Tarrío et al., 2011]. Since in the Urban circuit, it was necessary to detect gas in the form of elevated CO2, we used Sensorion SDC30, which is sensor dedicated to measuring just CO2. Using sensor data, we extrapolated and showed, not only vales to the operator but also eventual guesses where the CO2 artefact might be. Both methods are explained in [Tomáš, 2020].

5.5. Localization and mapping

Detection of the artefacts is not enough. One has to determine their position with 5-m accuracy within a global coordinate frame provided by the contest organizer. Therefore, after the object is detected, a robot has to determine the artefact position relatively to itself and then transform the position to global coordinates. For that, a robot, which detects the object, has to know its position with sufficient accuracy. Nowadays localization methods report [Zhang and Singh, 2015, Mur-Artal et al., 2015, Shin et al., 2018] errors around 1%, which, in theory, is sufficient for robots to venture more than 500-m deep into the underground environments before losing the ability to score any point by detecting objects. However, as discussed before, many of these tests are reported in ideal conditions only and operation in adverse conditions is investigated only rarely [Santos et al., 2015]. Our preliminary tests indicated that there is no universal solution that would suit all of the robots deployed by our team. The problem of self-localization , i.e., reliable estimation of the robot position, is tackled differently depending on the given platform.

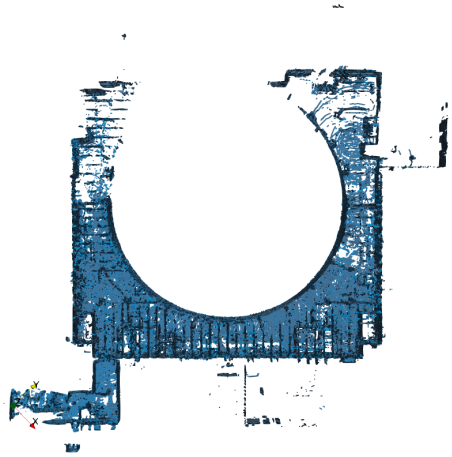
The wheeled and tracked robots employ EKF-based fusion and complementary filtering [Kubelka and Reinstein, 2012, Simanek et al., 2015] to provide dead-reckoning localization and an initial guess to the SLAM method [Pomerleau et al., 2013, Pomerleau et al., 2014] based on the iterative closest point (ICP) method [Chen and Medioni, 1992]. During SLAM, 3D scans from the lidar sensors are being aligned to an incrementally built global map, separately for each robot (see Figure 8 which presents the Urban Circuit maps from the Husky robot). Since the maps created during the Tunnel Circuit showed bending effects in the initial long corridor, we applied gravity-vector-based constraints in the Urban Circuit. They are an extension of the ideas presented in [Babin et al., 2019] and they constrain the ICP algorithm in the roll and pitch angles. Accuracy of dead-reckoning localization of the tracked robots, which suffers when traversing vertical obstacles, is improved using a combination of explicit modelling of robot kinematics and a learned support-vector machine model [Kubelka et al., 2019].

By a design choice, we do not perform loop closure. Instead, the localization pipeline is tuned to stay within the required accuracy to successfully report the detected artefacts. In the ICP algorithm, the chosen cost function for matching point clouds is the distance between the lidar points and the map planes (*point-to-plane* minimizer). We have found this to be robust for the environments we observe in the SubT challenge. The main concern is the localization drift which manifests itself in map bending and warping. To demonstrate the character of mapped environment, we ran the A-LOAM[2] algorithm, an advanced implementation of the original Lidar Odometry and Mapping (LOAM) [Zhang and Singh, 2014], on data recorded by our Husky robot in the second Alpha and Beta sorties. Figure 9 shows the A-LOAM and ICP maps of the Beta sortie. The robot entered through the long south corridor and followed the circular reactor wall. In this situation when loop closure is not possible, it is hard to avoid localization drift. The drift can be only reduced, not eliminated, by tuning the algorithm, by adding complementary measurements or by profiting from some *a priori* knowledge (e.g., that the floors are usually level). An example of a complementary measurement for a LOAM-like localization is the work of Ye [Ye et al., 2019]. In our approach, we can constrain directly the ICP algorithm to align the point clouds with the measured gravity vector.

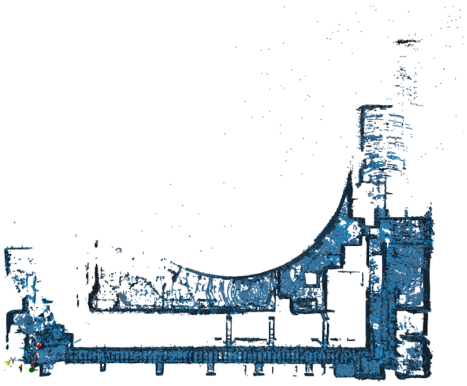
Moreover, thanks to ground-truth position computed by localizing in the maps provided by DARPA after the Urban circuit, we are able to evaluate localization error. The plot in Figure 10 demonstrates that the main problem is the drift in the Z axis caused by the map bending. In the case of the Beta course, the highest error around the time 2000 s corresponds to relative error lower than 1%. This result supports our decision not to strive for loop-closure mechanisms in this application. However, monolithic point-cloud maps also yield some disadvantages. Computational complexity grows with space explored and mapped, but this can be solved by appropriate memory management. A bigger problem that we encountered were erroneously aligned point clouds that damaged the map. These misalignments usually occurred when navigation over rubble and introducing high rotational speeds, or when crashing into obstacles. To cope with this problem, we are currently investigating available point-cloud motion-deskewing methods. Finally, there were several instances where the rotating 2D lidar on the tracked robots would stop working, mainly because of crashing into obstacles and stopping the servo drive. This mechanical liability will be solved replacing the 2D lidars with modern multibeam, 360-degree ones.

The UAVs were initially localized by a Kalman-filtering-based state estimation system [Petrlík et al., 2020], which was fusing measurements from IMU with position corrections obtained from Hector SLAM [Kohlbrecher et al., 2011]. Hector SLAM is a motion estimation algorithm explicitly developed for SAR scenarios. The 2D laser scans generated by the RPLidar A3 laser scanner are first transformed into a local stabilized coordinate system using the UAV orientation measured by IMU. After preprocessing the laser scan by discarding the endpoints that could potentially originate from the ground or the ceiling, the rest of the points are aligned into the online-estimated occupancy grid map.

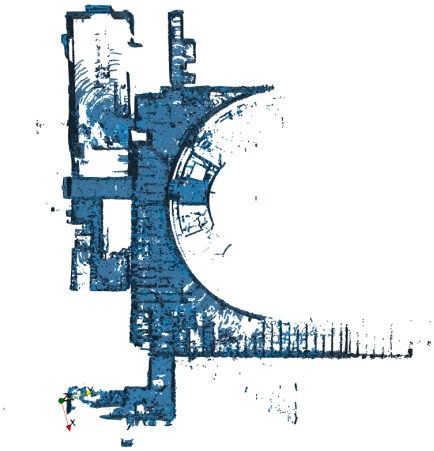
[2] https://github.com/HKUST-Aerial-Robotics/A-LOAM



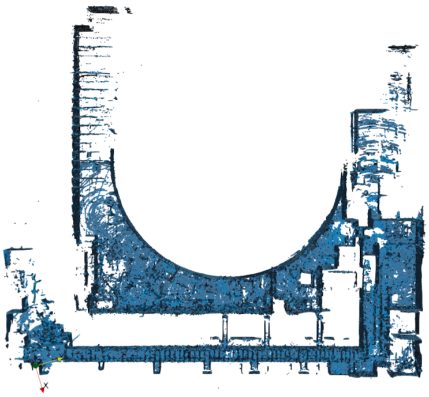
(a) Alpha course, first run.



(b) Beta course, first run.

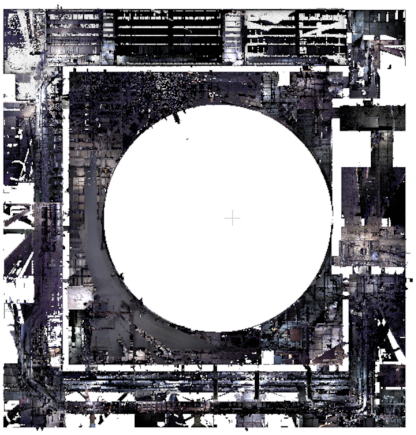


(c) Alpha course, second run.



(d) Beta course, second run.

(e) Ground truth map of the Alpha course.



(f) Ground truth map of the Beta course.

Figure 8. Mapping results of the Husky robot (top-down view) from the Urban Circuit along with the ground truth maps for the Alpha (8e) and Beta (8f) courses.

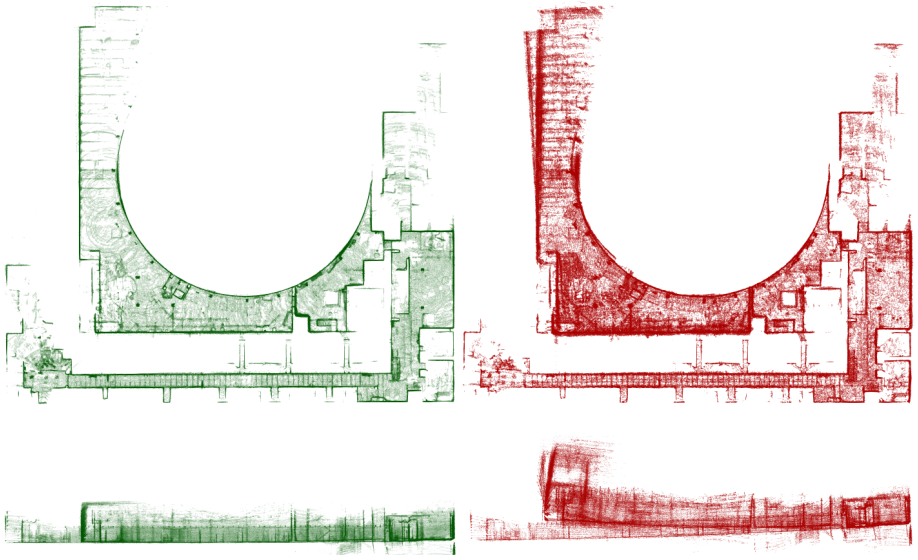


Figure 9. Maps of the Beta course obtained from our localization pipeline (left, green) and from the A-LOAM algorithm (right, red). The sensor data come from the Husky robot during the second Beta run. The top row presents a top-down view, the bottom shows side-views of the resulting maps.

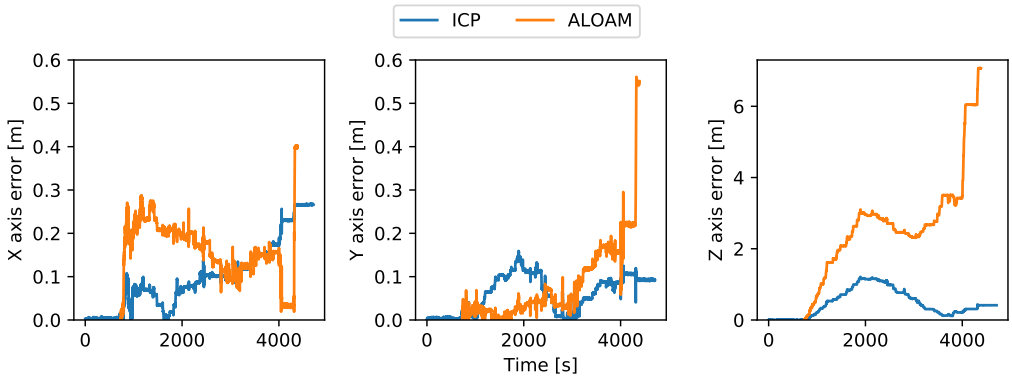
The localization method of the UAVs was changed after the Tunnel circuit due to a more vertically diverse environment of the Urban circuit. The assumption of similar cross section among different heights that was used in the Tunnel circuit was no longer valid. To be able to navigate the more complex building interiors, the 2D lidar was replaced by the 3D (multiplane) lidar Ouster 0S1-16 with 16 measurement planes. Since Hector SLAM was not developed for full 6-DOF pose estimation, the position corrections for the UAV state estimation are provided by the A-LOAM. A-LOAM is based on extracting edge and planar features from the input point cloud, which are then matched into the global map represented by a gradually built point cloud. Contrarily to the localization pipeline run in the UGVs, A-LOAM does not require prior motion estimates from odometry and it is therefore a more suitable choice. To assist the localization and mapping pipeline, an adaptive intensity-based filter removes irrelevant data from the Ouster OS1 lidar in order to filter out clouds of whirled dust, which emerged in both the circuits due to the aerodynamic influence of the UAVs.

The localization of the hexapods is based on the Intel T265 camera module, which provides a position estimate based on a proprietary grey-scale visual-inertial SLAM algorithm. The robots are equipped with a secondary RGBD camera (Intel D435) which builds a 3D map [Bayer and Faigl, 2019], and uses the map to guide the robot to unexplored areas.

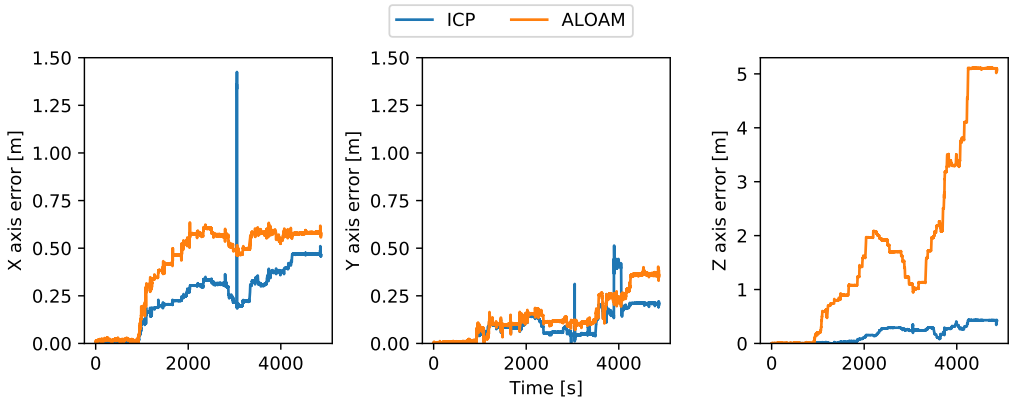
5.6. Navigation

5.6.1. UGV

Besides the map for the localization, each ground robot builds a separate map for navigation and exploration purposes. The navigation map is built from sensors available on the particular robot; walking robots used depth camera Intel RealSense D435, tracked robots and UAVs used 2D LIDAR, and wheeled robot fused range measurements from the 3D lidar with two depth cameras Intel RealSense D435. More in depth principles of navigational maps are based on the work of [Bayer and Faigl, 2019]. The map representation for walking robots used the elevation map with the underlying quad-tree structures, and cache enhancement described in [Bayer and Faigl, 2020]. For tracked robots and the wheeled robot, the map representation has further been extended to 3D by applying a custom speeded-up oct-tree data structure. The navigation maps were periodically analyzed for terrain



(a) Beta course, second run. Distance driven: 260 m



(b) Alpha course, second run. Distance driven: 247 m

Figure 10. Localization error of our ICP-based solution and of the A-LOAM algorithm. The error is prominent the most in the Z axis; note the different scale in the third column.

traversability to determine how to navigate the robot to the specified location. The traversability is estimated from the roughness of the terrain. When the roughness exceeds threshold k_d, the terrain is considered untraversable. Where k_d has been estimated for each robot type based on the robot kinematics and experimental results. In the opposite case, the terrain is traversable, and a path generated by the planner can be planned over the corresponding map cell. Plans for the robot position in the map to the single given goal location were produced by the A* algorithm with Euclidean distance heuristic. Moreover, the cost field generated using a generalized version of the Distance transform algorithm [Felzenszwalb and Huttenlocher, 2012] for 3D maps have been utilized by the planner to keep a safe distance from the obstacles or to avoid rough but traversable terrain if possible. We have dedicated the extra module for the following of the generated path. The module itself smoothes the path using a sliding window filter, and the combination of the two control laws ensures that the robot follows the path. The first control law steers the robot based on the displacement of the robot orientation, and the second control law sets the forward velocity, based on the distance to the closest obstacle, and distance to the next navigational way point.

The motion planning method used proved to be universally applicable, and it works well on our crawling, wheeled, and tracked robots.

5.6.2. Adaptive terrain traversal

To improve their ability to overcome adverse terrain, the tracked robots incorporate the information from their RGBD cameras and position their auxiliary tracks accordingly [Pecka et al., 2016].

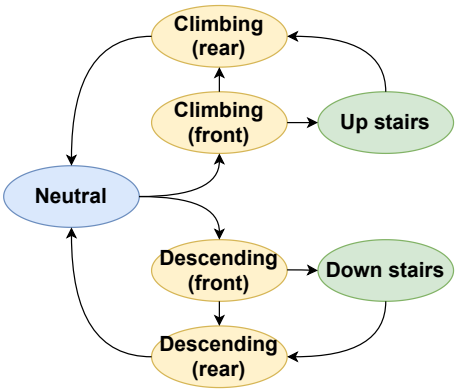


Figure 11. Finite state machine for controlling flippers of the Absolem platform, capable of traversing difficult obstacles and stairs. Blue and green states signify longer lasting modes, whereas yellow states are short lasting.

Thus, flipper control was done autonomously most of the time. This is because manual control would cause excessive cognitive load on the operator, especially when operating several platforms simultaneously. Manual control is also time-consuming as there is a several second delay between issuing the commands and receiving visual feedback of the flipper position. The flipper control algorithm is reactive and uses information from the IMU and a depth input from a front-mounted D435 camera. Our approach consists of a state machine, shown in Figure 11, where we define four primary states of obstacle traversability. The neutral state means folded flippers in such a way to minimize blocking the Lidar. Yellow states define configurations of the flippers that allow the robot to safely climb and descend obstacles by using the front and back flippers as support. These are triggered when the robot is moving forward on the flat ground, and a significant height increase or decrease is sensed in front of the robot in a designated region. For example, in the *Climbing rear* state the rear flippers push down to raise the rear while keeping the front flippers extended forward to shift the weight balance maximally forward and dampen the robot when it hinges forward. Whenever the robot finishes the ascent or descent on to flat ground the state resets back into neutral. The same state machine is used for stair traversal with a modification that the robot cannot turn while on a stairs. When setting flippers for any action, we make use of the natural tactile feedback of the ground by simply pressing down with low torque. This is better than strictly conforming the flippers to a height map which can be prone to errors.

5.6.3. UAV

As the aerial robots do not move on the ground, they do not need to consider the traversability of the terrain. Thanks to that, the UAV can potentially explore areas unreachable by the ground robots. The areas accessible by the UAVs are, however, limited by the narrowness of the corridors leading to these areas. With decreasing width of the narrow passage, the probability of a collision that would render the UAV incapable of continuing the mission rises due to inaccuracies in the localization and wind gusts caused by the propellers in tight areas. To achieve high reliability while simultaneously avoiding deadlocks caused by narrow passages, an exploration method that favours paths further from obstacles was developed.

For the Tunnel circuit, the navigation is based on the 2D occupancy grid produced by the Hector SLAM. First, the grid is downsampled, the obstacles are inflated, and then the distance of each cell to the closest obstacle is obtained by a distance transform. A modified A* algorithm is run at 2 Hz to generate the path from the current UAV pose to the goal using the heuristic:

$$h(\mathbf{x}) = dist(\mathbf{x}, \mathbf{g}) + o(\mathbf{x})p, \tag{1}$$

where $\mathbf{x}$ is the current evaluated point in the grid, $\mathbf{g}$ is the goal, $dist(\mathbf{x}, \mathbf{g})$ is the Euclidean distance, $o(\mathbf{x})$ is the distance to the closest obstacle, obtained from the distance transform, and p is a tunable

parameter, which controls how much the planning algorithm avoids obstacles. The used heuristic forces the generated paths to keep a distance margin from obstacles and pass through narrow passages only when another path cannot be found. The operator sets the goal, and with prior knowledge of the tunnel, the layout could be used to bias the exploration in a specific direction. During the competition, the goal was set to 300 m in the x axis of the SubT reference frame to explore deep into the mine without any bias in the y axis. Depending on the chosen exploration strategy, the UAV can search for artefacts its whole flight time and relay the positions of artefacts over the mesh network formed by other robots. The other option is to return to the initial position after reaching half of the flight time by setting the initial position as the new goal.

Due to the complexity of expected environments, the same approach could not be used for the Urban circuit. Therefore for complex 3D environments, the UAVs identify the next goal from the set of exploration way points. Paths to the individual locations are generated with the A* algorithm with the Manhattan distance heuristic applied on the volumetric map produced by the ALOAM mapping. To overcome problems of trajectory-tracking control errors and possible dynamic obstacles, the planning process is sped up by utilizing only the six-neighborhood and adaptive map resolution, and the current A* path is frequently replanned. The initial or a replanned path is then postprocessed with a fast iterative algorithm which smooths the path and increases an obstacle margin, providing safe navigation in a complex dynamic 3D environment in real time.

5.7. Exploration

Taking into account two circuits of the DARPA Subterranean competition, the two different exploration strategies were used for the ground robots. The first strategy is the frontier based exploration [Yamauchi, 1997], which navigates the robot to the closest edge between free space and unknown space. We have employed the improved frontier based exploration strategy described in [Bayer and Faigl, 2019], which proposes to lower the computational demands of the next navigational goal selection from the possible goal candidates by frontier cell clustering.

The disadvantage of the frontier based exploration is that the environment coverage by the spatial map does not correspond with covering the environment by the robot cameras to search for the artefacts, especially when the robot is capable of detecting obstacles from greater distances than to recognize the artefacts. Therefore, we have also introduced the exploration strategy that used the model of the robot cameras to generate such exploration way points, so the robot covers the environment by its cameras. The navigational way point, followed by the robot, was selected from the exploration way points based on the entropy, which estimates the information that the robot obtains by observing a particular location.

Plans to the close goals during the exploration were generated by local A* planner, or in case of multiple goal candidates by the Dijkstra algorithm. On the other hand, the homing that often required a very long plan took advantage of the already known map covered by the sparse graph connecting the selected traversable way points similar to the approach proposed in [Dang et al., 2019]. The human operator did coordination between the robots since he had complete information about maps of all robots.

To utilize the capabilities of UAVs, their exploration strategy aims at the exploration of further areas of the environment instead of a thorough exploration of nearby locations. In contrast to the ground robots, they utilize the map generated from the 3D lidar also for the generation of exploration way points. A navigational way point for the UAV is chosen according to a custom cost function, which reflects the deviation from a specified direction, and horizontal and vertical distance which need to be travelled in order to reach the way point.

5.8. Coordination

Coordination of the robotic team is performed primarily through the powerful data link provided by the 4G Mobilicom mesh transceivers. This allows the operator to operate the robots directly or to

set the goals where they should navigate. Moreover, the robots are aware of each other's positions using a low-bandwidth link established by the droppable "Motes" described in subsection 4.5.3. By remembering the shared positions, each robot can reconstruct the trajectory of each other robot in the team and avoid exploration of already visited areas. History of the positions also allows us to detect situations where a robot is mislocalized or stuck.

5.9. Interface

As the challenge rules allow only a single human supervisor to manage all the systems deployed in the mission, it was necessary to design a reliable and easy-to-use user interface (UI) to supervise and control the robot's behavior and report back found artefacts. With an increasing number of deployed systems, the UI plays a crucial role in the effectivity of the whole mission; therefore it is a part of the system that has undergone substantial modifications since the STIX event to increase its reliability, to cope with the modifications in the autonomous behaviors of the robots, and to simplify the work of the human supervisor. The current UI is composed of two major parts including an interface for robot control and interface for artefact reporting and several diagnostics utilities that allow the pit-crew to visualize the telemetry data from the robots and simplify the deployment process of the robots during the preparation phase before the scored run. The two major parts of the UI that are used to control the robots and report the artefacts are realized in the base station that is visualized in Figure 12.

Figure 12. The base station and its user interface used in the Urban circuit event.

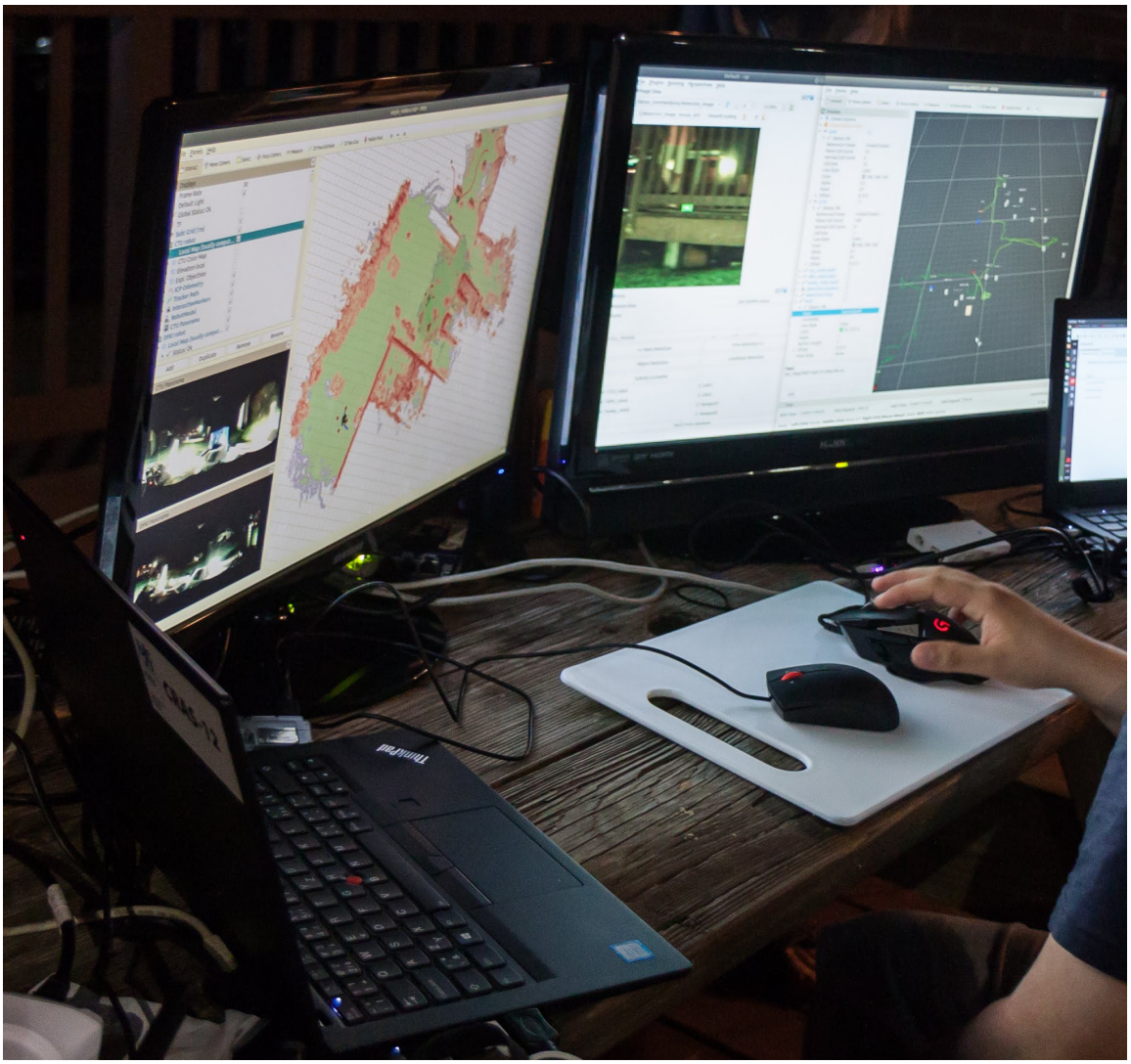


Figure 13. Block/ diagram of base station modules with status monitors for pit crew.

The base station UI is built on top of standard ROS visualization components, namely custom rqt plugins and rviz tool. The robot control UI is divided into two screens. The first screen shows the video feed and current status (e.g., localization status, battery level, emergency stop state) from individual robots using customized rqt plugins fitted in a single noninteractive rqt perspective. The second screen of the robot control UI uses interactive marker server and multiple custom plugins, e.g., to capture keyboard events, in rviz for control of individual robots. It allows to show and hide the constructed maps of individual robots and their horizontal cross sections to deliver the human supervisor the spatial awareness, and assign high-level commands like follow the target or explore using an interactive marker and keyboard shortcuts. Note that so far only the four large robots are controlled using this interface, the drones and hexapods are entirely autonomous, and they interact only with the artefact reporting UI.

The artefact reporting UI serves the purpose of accumulating the detections from all the robots and presenting these detections to the human supervisor that verifies them and sends them to the DARPA scoring server. The UI is realized as a custom rqt plugin and a ROS node that visualizes the

individual detections using the interactive marker server in rviz. As the detections are transmitted to the base station with their image, the rqt plugin presents this image to the human supervisor. The human supervisor may iterate through the detections, confirm or reject them, change any parameter of the detection, and finally he may send the detection to the DARPA scoring server. On top of showing the human supervisor the detection images in the rqt plugin, the detections are visualized inside the rviz as manipulatable interactive markers. The rviz visualizes the individual robots, their path, the detections, and also the strength of the Wi-Fi signal emitted by the cell-phone artefact and the CO_2 levels essential to detect the gas artefact that is measured by the robots along their path. The wi-fi and CO_2 levels are visualized as color-coded point clouds along the robot path. The rqt plugin allows switching the visualization of the individual elements for each of the robots in rviz on and off. Further, the detections visualized as interactive markers are color- and shape-coded based on their status, e.g., detections confirmed and rejected by DARPA are green and red spheres respectively, detections rejected by the human supervisor are small grey cubes. This allows the human supervisor to recognize the state of the detections immediately. Last but not least, the human supervisor is able to manipulate the spatial positions of the detections, and thus manually correct for the localization drift of the robots prior to sending the artefact positions to the DARPA scoring server. This feature has proven to be essential in reporting of many artefacts during Tunnel circuit where the robots had significant map drift.

6. Experiments and deployments

Before the "Urban" circuit at the discontinued nuclear power plant "Satsop" our team deployed the robotic system on four other different occasions: "Edgar" – Coal Mine in Idaho Springs, "URC Josef" – a CTU experimental mine, National Institute of Occupational Safety and Health (NIOSH) – mine facility near Pittsburgh and undisclosed location inside Prague "subway" infrastructure. Additional deployment in the "Bull rock" cave was performed as a preparation for the "cave" circuit which eventually did not take place due to the global pandemic. Our team participated in the virtual track after it had been announced that the systems track is canceled. Several other small-scale experiments were conducted mostly for testing individual parts of a system. Gathered data, including everything collected during our runs and other deployments, is currently available online[3]. This dataset is still not complete as new data is still being added and will be structured in the future after the whole competition is over.

6.1. STIX

The first deployment took place at the "Edgar" mine, where DARPA organized the STIX event. This event focused on testing the teams' readiness and DARPA's infrastructure for later competition circuits. In this deployment, four runs, each lasting three hours, were performed on two "ARMY " and "Miami " tracks. During these runs, the robots could detect and correctly localize three objects, even though they were controlled only by operator joystick. UAVs were deployed only on "Miami " to explore the first 50 meters autonomously before crashing due to a narrow corridor of the tunnel. Track "ARMY " was not favorable for drones due to the enormous amount of dust that hindered all platforms when the UAVs tried to lift off. No proper alignment to the DARPA map was implemented, and we have relied on starting the robots mapping on a specific place in front of the gate to the course. This resulted in improperly aligned maps of each robot as seen in Figure 14. Several problems had arisen during those tests, such as communication issues between robots caused by improper Mobilicom network settings and hardware failures, and issues with overseas battery transport. This lead to a need for faster robots (Husky), hardware improvements of other platforms, improved operator UI, and more frequent in-house testing. After all four runs were completed, the DARPA

[3] https://login.rci.cvut.cz/data/darpa-subt/data/

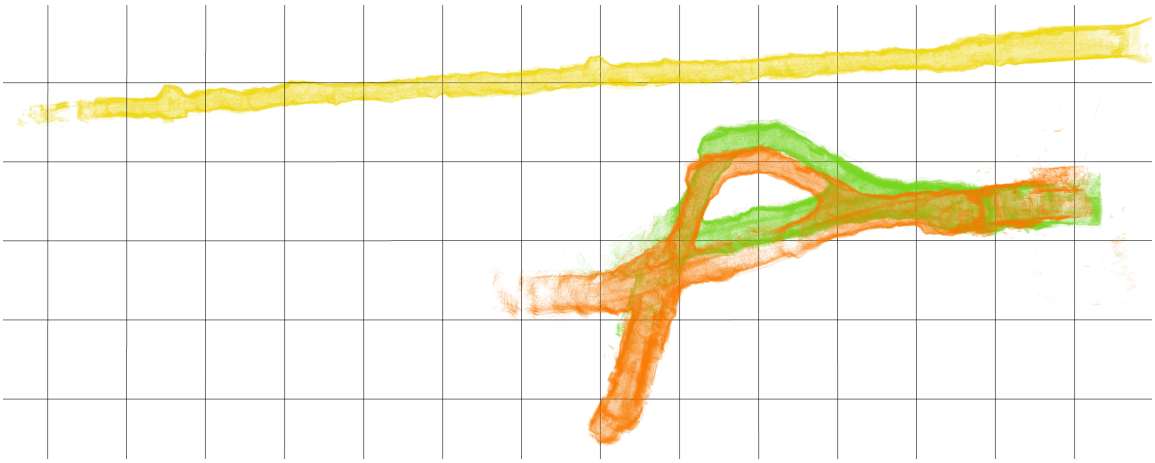


Figure 14. Map of "Miami" (top) and "ARMY" (bottom) courses during STIX event. Notice improper alignment of an angle to the DARPA frame in the ARMY tunnel. 10-m grid.

representative initiated a tour throughout the mines to demonstrate the scale and hazards that our robots will face in subsequent circuits. Those hazards included an area obscured by fog generator, constrained passages, rubble, and masterfully hidden artefacts visible only from specific angles or beneath the metal grating floor.

6.2. Josef

Our team held the second deployment of the robotic team in the "Josef" experimental mine managed by the Czech Technical University in Prague. The team "Robotika" also utilized this site simultaneously since it was one of few options of such tunnels accessible in the Czech Republic. This deployment consisted of several days in an actual mine to test and evaluate the system's features. Deployment at the "Josef" mine enabled us to gather more specific datasets for better evaluation of localization and mapping and training data for the neural network for object detection. We also gained new experience on robot behavior when driving them over gravel, wet railways, or through a light haze. Additionally, we realized the need to implement specialized modules, such as water and wet surface detection, to tackle the unreliability of rangefinding sensors scanning those types of materials. Additionally, we encountered issues with using RGBD cameras that could not reliably be used in dark places with CCTV surveillance cameras since infrared light from them interfered with the projected patterns of the camera.

6.3. Tunnel

A total of three contest deployments in the competition took place as a part of the DARPA SubT "Tunnel" circuit. This setting was divided into two separate tracks called Safety research mine (SR) and Experimental mine (EX), which both took place in the coal mine of the NIOSH. Each track had two runs, each lasting one hour, during which all artifacts that robots had in the line of sight were detected and scored.

6.3.1. Run 1, SR

The first run ended with only one artifact (backpack) found due to miscommunication between the DARPAs emergency stop (estop) system and our robot's receiver, which was wrongly configured. This meant that robots could not venture further than 10 meters into the tunnel due to constantly

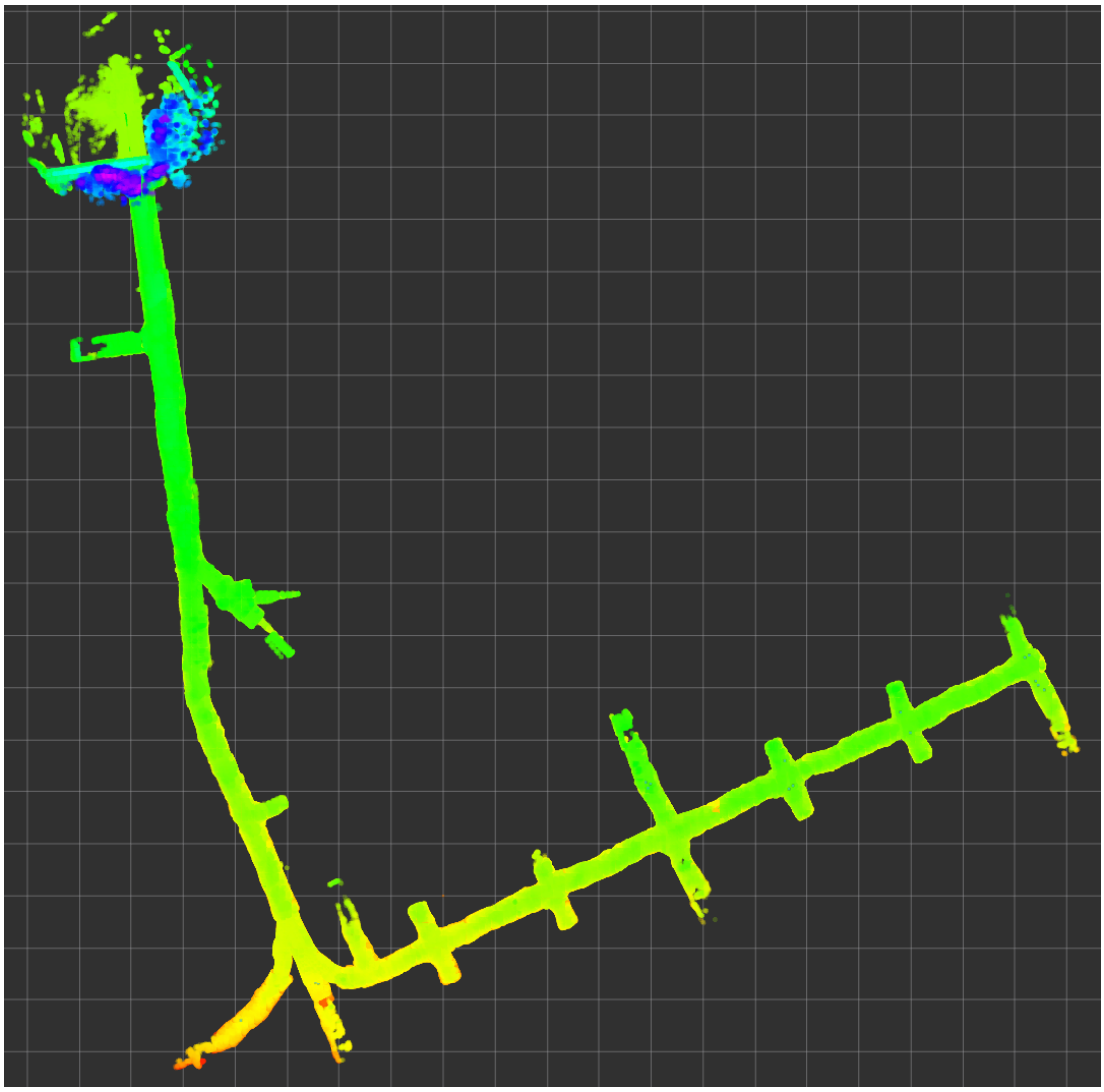


Figure 15. Top view of the Josef mine from a manually driven dataset. At the time, we have focused on planar mapping only. 10m grid.

receiving estop from a DARPA. The operator was able to reprogram one robot on the fly from outside the tunnel with limited bandwidth only. In the end, we had only eight minutes of mission time for a single robot that was able to find one artifact which was visible from the entrance. In this run, there was no autonomy involved on UGVs all drive was done by remote joystick. However, autonomous UAVs were deployed last minute to test the flight capabilities. One UAV could go about 100 m into the tunnel, and while heading back, it hit a wall with its propellers due to low ceiling and crash landed. The other UAV was tasked with going as deep into the mine as possible and having managed to travel 200 m into the tunnel, where it stopped due to the limited map size. The estop issue was fixed for the other three runs.

6.3.2. Run 2, EX

During the second run were able to deploy all robots into the tunnel. The tracked robot reached 190 m into the straight tunnel, where it detected a cellphone. This artifact has cost us ten tries to

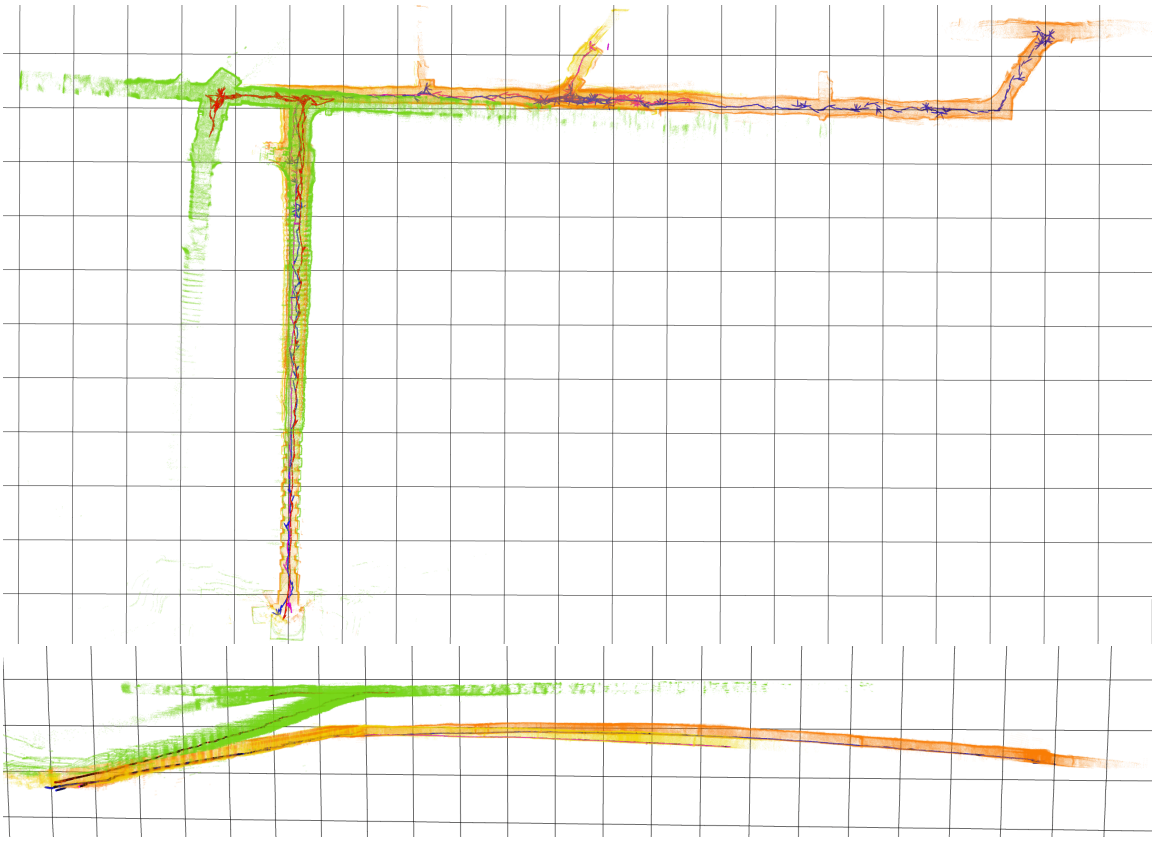


Figure 16. Top and side view of a Run 4 in Tunnel circuit. Slight misaligment of robots at the beginning causes visible error in mapping clearly visible at T juntion. Bottom image shows heavy drift in localization and map bending. 10-m grid

report due to poor robot localization drift, which exhibits itself by map bending, see Figure 16. The mislocalization had to be corrected manually by the operator by moving the estimated detection upwards in the GUI. The error was a 10-m drop on the 200-m distance driven, which was previously unobserved. Autonomy, where the operator is supposed to only set way points had all robots driving slowly due to the width of the tunnel where robots tended to look from side to side, causing a zigzag trajectory. This side-to-side movement and constant turning inevitably caused two robots to either crash into a wall or have their motors die due to high loads. The operator tried to mitigate these problems in at least one robot by driving it using force fallow markers which the robot follows regardless of obstacles by a straight line.

UAVs were again deployed in this scenario, but both were required to return back to the base since they had no wireless link back to the base implemented. One of the UAVs crashed about 60-m in while the other went 100 m in and then 80 m back, landing out of the crew's reach to gather any data. For the subsequent runs, the drones were connected to the WiFi network to send data back to the base if they are close enough to the course entrance.

Due to the map bending on tracked robots, we have recalibrated the localization stack and forced it not to use the gravitational vector from the IMU. The husky platform also showed issues with localization robustness since it did not have enough computational power to run ICP at a sufficient frequency. This was corrected by implementing the T265 Realsense optical odometry camera, which was subsequently mounted with additional lights on the front of the robot to utilize the ceiling of the tunnel for localization to help ICP convergence. Additionally, threshold values for traversability were tweaked, so the robots were not driving zigzag patterns through the tunnel.

6.3.3. Run 3, EX

The third run began with a husky platform driving about 80 m in a few minutes, with only a suggestion of direction from the operator, where it had one of its power fuses overloaded and shut down. This effectively rendered the whole platform immovable in the middle of the tunnel and shut down its communication links and retranslating capabilities. Other tracked robots had issues with traversability due to last-second tweaks, which caused all tracked robots to drive into the walls, where they often got stuck. This again forced the operator to babysit the robots and use operator GUI directly via "force fallow". One of the robots was sent by the operator to the side tunnel, where it lost a signal and then stopped there since it could not receive new commands.

UAVs in this run had issues with a start where they took off and hit the ceiling just after entering the course. Both of those flights tripped safety on the drones where they landed back to the ground safely. This issue happened due to the misconfiguration of the flight level.

Since this run, all our communication infrastructure that is supposed to retranslate any data was powered by its own small lithium battery to avoid issues with robots shutting down.

6.3.4. Run 4, SR

During the last run, no critical hardware issues occurred, making it possible for the robots to traverse more distance than in all other runs combined. The husky platform had an issue with frontier exploration where a few of them stayed outside of the course around the base where the Husky wanted to navigate constantly. This had to be manually mitigated by controlling the robot directly by way points which meant that the robot was driven only to the first crossroads, where it stayed as a retranslation unit for the rest until others could not be driven further. In the end, the platform was a way point driven into the unknown by the operator until the signal was lost. Afterward, the robot switched to frontier exploration, which effectively meant that it tried to return to base. The operator had to fight this issue by periodically sending new way points.

Tracked robots were driven by the operator by providing them with way points. One of the tracked robots got stuck in a small, water body that it could not sense by any means, making it not traversable terrain. The second was driven 250 m into the mine, where it began autonomous frontier exploration two minutes before the end of the run. The operator placed the third along the way to retranslate data back to base.

Throughout the whole tunnel circuit, the fully autonomous behavior (frontier exploration) was rather a hindrance since it was fighting the operators' decisions rather than helping him. About 90% of all driving was done by the operator using the assistive commands "force fallow" and way point driving, which required him to have a vague sense of direction where the robots can be sent. Due to scoring all artifacts that were visible, our team tried to focus more on the reliability of movement and autonomy for subsequent rounds.

Due to issues with aligning the internal robot maps with the DARPA frame, we have introduced a total station that is able to localize the robot in a global frame with submillimeter precision. This also allowed us to gather better ground truth data which was previously impossible to do since we did not own any equipment of precise long-range measuring. Preparing for the Urban circuit, we were allowed to perform tests in undisclosed sections of the Prague subway. This helped obtain a deeper insight into radio communications issues in urban structures and on the traversability of narrow passages, bridges, and stairs.

6.4. Urban

So far, the last official deployment was performed during the "Urban" circuit in "Satsop", which is an unfinished power plant. Both tracks of this circuit were located around the power plant's unfinished reactor. In total, our robots drove about 2.5 Km and flew 500 m, while all artifacts that robots had in the line of sight were detected and scored.

6.4.1. Run 1, Beta

During the first run, most ground robots had severe mobility problems because of the debris scattered on the ground. The Husky platform wrongly assumed drivable terrain over a laying beam on which it gets stuck by balancing on its undercarriage. Husky got freed by another tracked robot that bumps into the stranded robot, making it tip over and eventually break free. Husky then explores the unknown until it loses the link to the base and explores autonomously 20 m in after it returns to the signal. The Husky rescue caused the other tracked robot to lose mapping and get lost. This was caused by the pivoting lidar error, which stopped moving and therefore not returning 3D point clouds necessary for localization. The second robot got stuck in between two concrete blocks, out of which he could not get. The last tracked robot was wrongfully planning over uneven terrain even though it could be driven around. The robot had issues with an exploration algorithm that was often planning back to the base after it was commanded by the operator to travel deeper into the course.

During this run, both drones crashed, entering the course to the door frames. The liftoff also caused the dust to be perturbed, making it impossible for the small hexapods to navigate.

6.4.2. Run 2, Alpha

The second run had a significant problem with the localization of one of the tracked robots. This again happened due to the robot crashing with its pivoting Lidar to a wall which it deemed traversable, making it stop. The operator took full responsibility for this event since he tried to drive the robot using a joystick and the cameras manually. Both of those data streams had a lag of few seconds, and the operator GUI was constantly crashing while this happened.

Two tracked platforms were able to traverse stairs to a lower level of the track. This traversal was done manually for the first robot with the help of a camera and joystick with a good data stream since the robot was still in range of high bandwidth network to the base. The second tracked robot first autonomously drove over our small hexapod platform, which acted as a Mobilicom retranslation unit. Then the robot waited on each mezzanine of the stairwell for approval from the operator to proceed. Otherwise, the descent was autonomous. Both downstairs robots were driven by the operator using way points in the GUI. One lost a localization due to a mapping error, whereas the other was autonomously driving back to base if the operator did not set it a way point in the last 30 seconds. This is especially difficult since the communications allowed the operator to command the robot with about 10 seconds delay. The robot kept returning to the signal because it had an unexplored frontier back at the base.

The Husky platform explored the top floor around the reactor where it once got into a position it could not get out of due to perceived obstacles in its surroundings. The operator's input freed it by issuing the "force fallow" command through GUI two minutes before the end of the run.

The UAVs both entered the course without any issue where one crashed after about 50 m to a wall due to a glitch in localization. The other traveled about 100 m into the course and ended hovering above one place since it could not navigate back to the base. It landed safely before the batteries ran out.

Between runs 2 and 3, several days worth of work was done on robots, such as improving the stability of GUI, implementing a reset function for mapping based on the last failed map, ability to restart any part of the robot for the operator and health status of mapping. Additionally, more problems were discovered in the pipeline, such as a wrong time setting of the robots when they could not connect to the internet after booting. This caused WiFi detection to be unusable since they relied on time synchronization between all onboard computers of robots.

6.4.3. Run 3, Beta

For this run, the traversability of the robots was lowered so they would not drive over debris that was dangerous for them to get stuck on. This caused other issues, such as almost no robots wanted to traverse anything that was not totally flat; therefore, the robots had to be driven using "force

fallow" to a way point in an operator GUI. The majority of this run had to be supervised by the operator in this way.

The pivoting Lidar of one tracked robot had a hardware failure that caused it to lose localization. After several remote restarts, it was deemed unsavable and was manually driven using a joystick by the operator to the crossroads to serve as a retransmission unit.

Another tracked robot tried to climb stairs to the upper level on the operators' command. During the climb, the motor controllers reset due to overload, which was commonly encountered on this aging hardware. In such a situation the robot runs the preprogrammed policy from the tunnel circuit. It begins to restart all software and then hardware by importance. This only causes a robot to stop, but if the actual motor controllers have to be restarted, the flippers cannot support the robot's weight. Since this happened on the stairs, the robot relied on the back flippers to hold it in place, the flippers loosened, and the robot tumbled down the stairs. No parts except the 3D printed mount for depth cameras were broken, and the robot continued to function albeit not movable. The operator shut down the robot remotely since it was turning its flippers indiscriminately due to a pose it has never observed (undefined behavior).

The last tracked robot followed the same path as in Run 1. Drones again did not make it through doors. During this run, a brief power outage caused our operator to lose communications to most of the robots and a display of half the base station. The deployed system had no issue since it primarily relied on battery-powered infrastructure, including our networking.

6.4.4. Run 4, Alpha

In this run, one tracked platform went down the stairs fully autonomously, with only a confirmation from the operator that it can be done. It autonomously explored about 150 m of the lower level where it snapped off all of its droppable communication units due to a turn that was too close to an obstacle. This was caused by not adding the communication units to the robot model, which were protruding out to the back of the robot. The unintended drop did not impact the performance of the network at all since this communication interface was in range all the time.

The Husky and other platforms explored the top floor, where the tracked platform was mostly command driven by the operator using way points since he was afraid that it might get stuck on something with its problematic motor controllers. The Husky drove the whole path in autonomous exploration mode. The last tracked robot was not localized in time by the total station in the base, so an impromptu connection to the robot from the total station to the robot had to be made via wire. Due to this error, we positioned the robot into the track via joystick about 40 minutes into the run with improper localization to the DARPA frame to act as a retranslation unit for other robots.

During this run, two drones were sent in. One was programmed to fly close to the reactor and then land with a Mobilicom unit. This happened flawlessly. The other UAV crashed and burned down during autonomous exploration. The reason for the crash was a collision with a beam that forced the drone to drop fast, braking its plastic landing feet. Due to a rough landing, battery lead insulation was damaged, and the battery shorted itself, subsequently catching on fire. The fire destroyed most of the platform, including all sensors except the Ouster lidar. Data was later recovered from the onboard computer drive. Even though all those mishaps, we managed to secure third place overall while being first as a self-funded team, i.e., the ranking was the same as in the "Tunnel" circuit.

6.5. Bull rock

In anticipation of the cave circuit, our team went testing to a "Byci skala" (Bull rock) natural cave located in Moravia. This environment is somewhat modified for easier walking having a pathway that comprises parts of mud, sand, rock, and concrete bridges without a railing. The cave is about 250 m in length with steep inclines and declines, which are on the main path eased up by tall stairs sculpted out of the mud.

Figure 17. Ground truth map of the Beta course in the Urban circuit. Only the top floor since our team was unable to explore any other level. Robot positions overlayed in colors each representing a different robot during a run 3. 10-m grid.

Table 5. Scored artifacts during the urban round of the competition. Any score point that has been detected by multiple robots is indicated by the brackets. False positives (FP) are combined from each robot separately, meaning one falsely detected object from all robots would account for four FP.

Run	Backpack	Vent	Survivor	Cellphone	Co2	Total	FP
1	1	1	0	0	1	3	31
2	2	2	0	0	0	4	71
3	0	1	1	1	1	4	55
4	1	3	1	0	1	6	54

Multiple runs with all robots were made with or without autonomy to find answers to several questions. Firstly, we tested the capability of tracked robots on stairs without distinct ridges made out of mud, which was difficult to traverse due to slippage. Additionally, we have tested autonomous navigation with negative obstacles, which our team did not have a chance to test before except stairs. This led to the flipping of a Husky robot that ran off to the side caused by elevation mapping where negative obstacles were not as inflated as walls were. All parts of the Husky robot stayed intact, including the lidar sensor, which held the whole weight of the robot without any roll cage. First

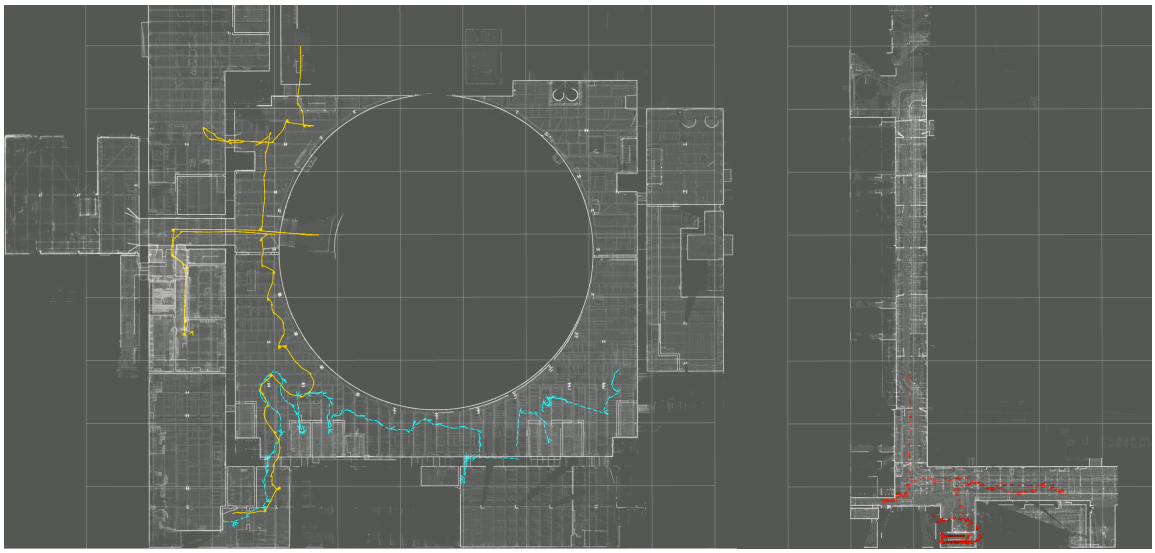


Figure 18. Ground truth map of the Alpha course in Urban circuit with upper (left) and lower (right) floors. Robot positions overlayed in colors each representing a different robot during a run 4. 10-m grid.

tests with a ballonlike device were tested mostly for abrasion against the low ceiling of the caves. Latex-based ballons proved to be useless since they lost most of their helium through porosity. Foil-based balloons showed promises of being able to withstand pulling over rough rocks.

During our trials, we were able to traverse most of the cave autonomously, albeit with close supervision, except the stairs part, which has proven to be difficult for all UGVs. Tracked robots had issues with battery life to a level requiring two or three batteries spare to travel in and out of the cave.

Tests of detecting water for avoidance were conducted using a Husky platform for driving in a shallow (5 cm) water stream that runs throughout the cave. All attempted solutions to this, including usage of stereo cameras, polarized light, automotive radars, and sonars, did not yield any acceptable results.

Additionally, comms propagation tests showed that in the low branching environment we had, it is possible to interconnect the whole cave using only four units of Mobilicom devices if placed optimally. Complex tests with MOTE units were also conducted to improve comm drop predictions as shown in [Zoula et al., 2021].

During our experiments in the cave, some robots suffered from problems with condensation. The cave has a temperature of about 8°C and nearly 100% humidity. This causes all robots to be cooled down to a temperature that can cause condensation on platforms while leaving the cave. We eventually had an issue with a Husky platform not working, most likely due to water condensed on internal components.

6.6. Virtual

Due to the cancellation of the Systems Cave circuit, we could only participate in the Virtual part of the challenge with which we had no prior experience. Before we decided to participate, the only experience we had with simulation was from drones and traversability experiments with Absolem, which made it quite challenging for us to establish a workflow. To make it more complicated, our private computational grid only supports Singularity HPC containers, while the SubT challenge only supports Docker containers. Another significant difference of the competitive environment was the use of Ignition Gazebo, as all of our previous simulations were running in Gazebo Classic, which is very different from Ignition Gazebo.

6.6.1. Cloudsim

SubT Virtual challenge offers the Cloudsim web simulation environment to registered teams. It can run up to three simulations (packaged as Docker images) simultaneously, each with a real-time factor 3% – this means that each 1-hour simulated run takes about 30–40 hours of real time. The teams do not have access to the internal state of the simulation during its run. All interesting data for the teams have to be packed in a 2 GiB ROS Bag file per simulated robot (this especially means it is impossible to store all camera images – not even compressed). We used DRACO compression for point clouds which at least allowed us to store all point clouds collected during the simulation. When the simulation is finished, the teams are given the ROSBag files together with the ground truth positions of the robots (which are not accessible during the simulation).

6.6.2. Porting Docker images to Singularity

The inability to observe the internal state of the simulation and custom code when it is running is very limiting in debugging. Put together with the fact that each simulation runs 30–40 hours and cannot be preempted, using only Cloudsim for debugging would be almost impossible.

We thus put effort into transforming the provided Docker images into Singularity containers. Singularity offers a tool to import Docker images, and it works well. However, there are differences in the philosophy of Singularity compared to Docker, which required further effort.

Most importantly, Singularity mounts the home folder of the user running the container and makes it accessible in the virtual environment, together with most environment variables. So wherever the Docker image assumes that the *HOME* environment variable will always be */home/developer*, it is different in Singularity. Moreover, the SubT virtual simulator hardcodes the */home/developer* path at some places of its code. These places need to be manually found and substituted with the *HOME* variable – but not always. The */home/developer* path is used in two different contexts: (1) to point to prebuilt simulator files, which are stored under */home/developer* when building the Docker container, (2) as a relative location of configuration and log locations used by ROS and Ignition Gazebo (e.g., */.ros/log* for ROS logs). All occurrences of context 1 have to be replaced with */home/developer*, and all occurrences of context 2 have to be replaced with reference to *HOME* variable (to store the logs in the host user's home directory).

Another significant difference is that the Singularity images and containers are immutable. So the only two writable locations are, by default, the user's home directory and */tmp* folder (which is also shared with the host system). This is why writing logs and other generated data in */home/developer* is not possible.

The last major difference is that the user has the same permissions towards the rest of the system inside the Singularity container as he or she has in the host system. So if the user is not allowed to call *sudo* in the host system, calling it inside the container is also not allowed (not even during the build of the container). This is why we have to build the images on personal computers (where we have root privileges) and upload the built images to the computation grid (where we only have standard user privileges).

With all these differences resolved, we can build Singularity images and run them on the local computation grid. When the simulation is running in this grid, we have full access to both the simulator and custom code internal state, we can observe topics, and we can even interact with the simulation, pause it, move the robots, view the raw images from cameras, and so on. This dramatically increases the efficiency of debugging problems with the simulation.

6.6.3. Differences between Real and Simulated Missions

We were looking forward to working with a fleet of robots that do not break or fall apart after every mission in a hardware sense. However, it showed that even the simulated robots have their problems that need to be addressed.

A big difference was the simulation of tracked robots. Ignition Gazebo, unfortunately, cannot simulate tracks in a faithful way, so the main tracks of Absolem were simulated by eight wheels. This results in very different behavior of the robots in complex terrain, and usually, the wheels

have worse traversability – it is, e.g., almost impossible to climb up staircases with the wheel-based tracks. This made the Absolem robots much less useful in the simulation.

On the other hand, the Husky robot models provided in simulation worked better than in reality, and as the robot cannot break any expensive sensor, we were able to run it much faster than we dared in the Systems track. Each Husky robot was thus able to explore up to 3 km of caves in a simulated run. Our record from Tunnel and Urban circuits was about 400 m with Husky.

6.6.4. Path Planning Internal Competition

As the use of a simulator allowed us to do much more tests than what can be done with real robots, we took the opportunity and developed two separate path planners based on different principles.

The first one was the *RDS planner*, an evolution of the path planner used in Tunnel and Urban circuits (described in Section 5.6).

The second path planner, called *Naex*, used an unstructured point-cloud map incrementally built from available RGBD and lidar depth measurements as the primary representation. The map is accompanied by an approximate nearest-neighbor graph constructed using the FLANN library [Muja and Lowe, 2009]. Each point in the map is assigned a reward based on its closest distance to any robot so far and the number of frontier points nearby. Dijkstra's algorithm, namely the implementation from the Boost Graph Library (see [Siek et al., 2001]), is then used to plan the shortest paths to all reachable points, and the point with the highest ratio of reward to distance is selected as navigation goal.

As we did not know which path planner would perform better in simulation, we set up an internal competition of the path planners. We ran dozens of simulations with the whole fleet of robots with each of the path planners. When the simulations were finished, we evaluated the number of points scored, explored area and other metrics, and chose the Naex planner for use in the scored runs.

We have even considered randomizing which path planner would be used, but the idea was rejected in the end as it was not possible to change the robot roster for the scored runs, and Naex worked better with Husky, whereas RDS planner worked better with Absolem.

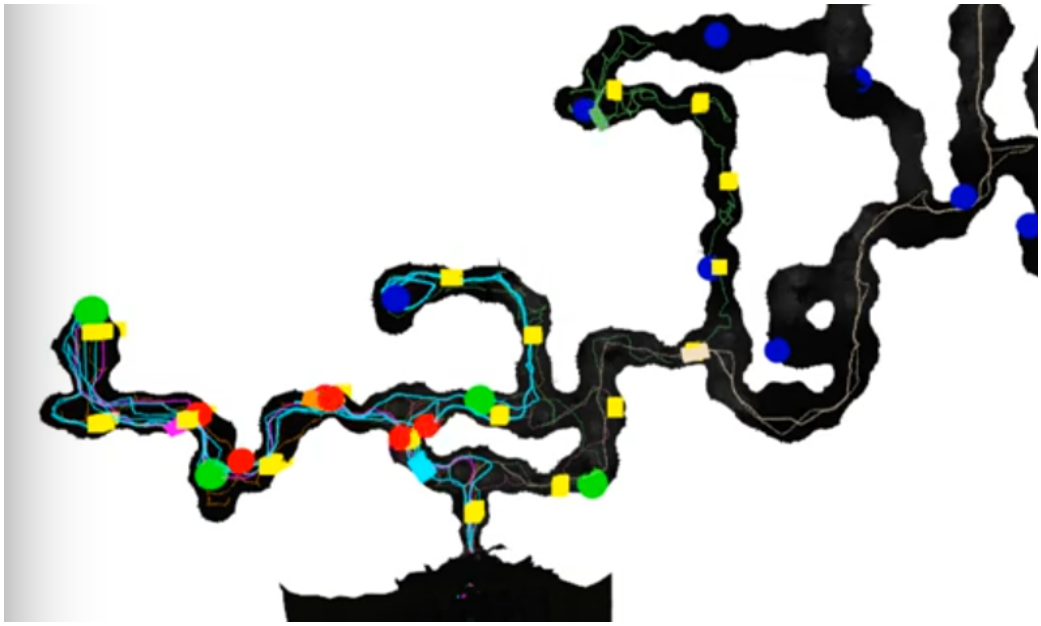


Figure 19. Path trace of a scored competition run. Colored lines are traces of individual robots. Blue circles are ground truth artifact positions. Green circles are scored artifacts. Red circles are unsuccessful artifact reports (wrong position or class). Yellow rectangles are dropped communication motes.

6.6.5. Deployment

Converting our system to be deployable as a Singularity and Docker image helped to straighten our workspace build process and catching many minor deployment errors like missing declarations of dependences, undefined build order, missing install directives, and so on.

One of the tricky parts to handle was the detector weights. They come as a large binary file (about 500 MiB) and change often, so it is not possible to store them as a part of a Git repository as the rest of the system. So we have them stored in a shared network location via HTTP protocol and add them to the images in a special build step. However, this deployment process seemed to have problems with local caching of the detector weights, so the UGVs were using some older version of the weights in the competition, which hindered their detection capabilities.

6.6.6. Cave Virtual Competition Results

The scored runs of the virtual competition were run in eight different virtual worlds, and there were three independent runs of the system for each world. This totals 24 scored runs. The ranking of teams was done by summing up the number of artifacts their robots correctly reported.

When data from the scored runs were made available to use, we ran numerous analyses to learn about our system's behavior. This way, we found the outdated detector weights on UGVs. An example graphical evaluation of a run is shown in Figure 19.

Table 6. Scored points of our robots in scored runs. Robots *UAV2* and *UAV4* correspond to models *SSCI_X4_SENSOR_CONFIG_1*, *UAV5* is *EXPLORER_DS1_SENSOR_CONFIG_1*, and *X1, X2* and *X3* are *EXPLORER_X1_SENSOR_CONFIG_2* Husky-like ground robots. Run name is composed of numeric ID of the world and index of the run (1-3).

World-Index	UAV2	UAV4	UAV5	X2	X3	X1	Total
1-1	1	0	2	1	0	0	4
1-2	0	0	3	0	0	0	3
1-3	0	0	2	0	0	0	2
2-1	0	1	3	0	0	0	4
2-2	1	1	2	0	1	0	5
2-3	1	2	1	0	0	0	4
3-1	1	1	1	0	0	0	3
3-2	3	0	0	0	0	0	3
3-3	2	2	1	0	0	0	5
4-1	0	3	0	0	0	0	3
4-2	0	2	0	0	0	0	2
4-3	1	1	2	0	0	0	4
5-1	1	0	1	0	0	0	2
5-2	0	0	3	1	0	0	4
5-3	0	1	1	0	0	0	2
6-1	3	4	0	0	0	0	7
6-2	5	2	1	0	0	0	8
6-3	1	4	2	0	0	0	7
7-1	3	0	1	0	0	0	4
7-2	2	0	0	0	0	0	2
7-3	1	0	0	0	0	0	1
8-1	0	1	1	0	0	0	2
8-2	0	2	3	1	0	0	6
8-3	1	2	1	0	0	0	4
Total	27	29	31	3	1	0	91

Table 7. Correct hypotheses of artifact locations. This table shows the number of correctly placed artifact location hypotheses each robot created during a scored run. Some of the hypotheses were not reported and scored because the robots did not return to comms until the mission end (e.g., because they flipped over or there was a path planning error).

World-Index	UAV2	UAV4	UAV5	X2	X3	X1	Total
1-1	1	0	2	1	0	0	4
1-2	1	0	3	1	1	0	6
1-3	0	1	3	1	1	1	7
2-1	2	1	3	0	3	3	12
2-2	2	1	3	2	1	0	9
2-3	2	3	2	1	1	3	12
3-1	3	1	3	2	3	1	13
3-2	4	2	2	1	2	1	12
3-3	3	3	1	2	2	2	13
4-1	0	3	2	0	0	0	5
4-2	0	2	2	0	0	0	4
4-3	1	1	2	1	0	0	5
5-1	2	0	2	1	1	3	9
5-2	0	1	4	4	2	7	18
5-3	0	2	1	2	1	6	12
6-1	3	5	0	3	0	0	11
6-2	5	2	2	0	3	0	12
6-3	2	4	4	0	0	0	10
7-1	3	0	1	0	2	2	8
7-2	2	0	0	6	1	0	9
7-3	2	0	0	0	0	0	2
8-1	0	1	1	1	1	1	5
8-2	0	3	4	1	1	1	10
8-3	1	2	3	1	1	1	9
Total	39	38	50	31	27	32	217

We have composed a table of points scored by each robot. There is not a unique link between a scored point and a particular robot, as each artifact can be seen and reported by multiple robots. In this case, we have assigned the point to the robot that saw the artifact first. The overall scored points are in Table 6.

Table 7 shows how many artifacts were detected and localized at all by each robot. Some of the artifacts were not reported because the robots did not always return to broadcast their positions in time.

7. Conclusion

Team CRAS-CTU-NORLAB has deployed a multi-robotic system in diverse subterranean environments during the DARPA SubT. Our team approach for the contest was primarily focused on single-agent robustness to mitigate possible issues with a small robot fleet. This resulted in most advances being made in localization, communications, and detections. Robust ICP based localization allowed for rather rough handling of the robots during the drive, which is typical when small platforms traverse harsh terrains. The implemented object detection pipeline based on YOLOv3 showed its strengths mainly in a way that no object has been missed. Most of the runs were either teleoperated or heavily influenced by the operator who was able to control robots via way-point navigation most of the time due to our multi-network-based communications.

The first round of the competition called "Tunnel" was held in a coal mine where robots searched for artifacts in repetitive tunnels and long hallways with low ceilings and water-soaked ground. In the second "Urban" scenario in February 2020, robots had to operate in the urban underground with service stairs, metal bridges, small rooms, metal fixtures, and holes in the ground. This circuit took place at an unfinished nuclear power plant. In both official deployments, our team managed to secure the third rank while being the winner among non-DARPA funded teams [DARPA1, 2019] [DARPA2, 2020].

In addition to the systems competition rounds, our team participated in the Virtual Challange as well after it was announced that the system track was being canceled due to a global pandemic. After a long struggle, we managed to secure sixth place. We consider it a reasonable result given that we had only four months to port our pipelines into the simulated environments, with which we were not familiar.

In the future we plan on expanding our knowledge in multi-robot cooperation, a better understanding of the signal propagation as well as the implementation of new platforms such as doglike robots or large hexapods.

Acknowledgment

The presented work has been supported by the Czech Science Foundation (GAČR) under reseach projects No. 18-18858S, 19-20238S, 20-10280S, 20-27034J, and 20-29531S, and by OP VVV MEYS RCI project CZ.02.1.01/0.0/0.0/16_019/0000765.

ORCID

Tomáš Rouček https://orcid.org/0000-0003-3598-1630
Martin Pecka https://orcid.org/0000-0002-0815-304X
Petr Čížek https://orcid.org/0000-0001-6722-3928
Tomáš Petříček https://orcid.org/0000-0002-3136-5673
Jan Bayer https://orcid.org/0000-0003-1190-1085
Vojtěch Šalanský https://orcid.org/0000-0003-2977-0976
Teymur Azayev https://orcid.org/0000-0003-2267-5722
Daniel Heřt https://orcid.org/0000-0003-1637-6806
Matěj Petrlík https://orcid.org/0000-0002-5337-9558
Tomáš Báča https://orcid.org/0000-0001-9649-8277
Vojtěch Spurný https://orcid.org/0000-0002-9019-1634
Vít Krátký https://orcid.org/0000-0002-1914-742X
Pavel Petráček https://orcid.org/0000-0002-0887-9430
Dominic Baril https://orcid.org/0000-0002-7283-8406
Maxime Vaidis https://orcid.org/0000-0002-1749-7207
Vladimír Kubelka https://orcid.org/0000-0001-8393-9969
François Pomerleau https://orcid.org/0000-0003-1288-2744
Jan Faigl https://orcid.org/0000-0002-6193-0792
Karel Zimmermann https://orcid.org/0000-0002-8898-4512
Martin Saska https://orcid.org/0000-0001-7106-3816
Tomáš Svoboda https://orcid.org/0000-0002-7184-1785
Tomáš Krajník https://orcid.org/0000-0002-4408-7916

References

Atkeson, C. G. et al. (2018). Achieving reliable humanoid robot operations in the DARPA robotics challenge: Team WPI-CMU's approach. In *Springer Tracts in Advanced Robotics*, pages 271–307. Springer.

Babin, P., Dandurand, P., Kubelka, V., Giguère, P., and Pomerleau, F. (2019). Large-scale 3D mapping of subarctic forests. In *Proceedings of the Conference on Field and Service Robotics (FSR). Springer Tracts in Advanced Robotics*. Springer.

Bayer, J. and Faigl, J. (2019). On autonomous spatial exploration with small hexapod walking robot using tracking camera intel realsense t265. In *European Conference on Mobile Robots (ECMR)*.

Bayer, J. and Faigl, J. (2020). Speeded up Elevation Map for Exploration of Large-Scale Subterranean Environments. In *2020 Modelling and Simulation for Autonomous Systems (MESAS)*.

Bjelonic, M., Sankar, P. K., Bellicoso, C. D., Vallery, H., and Hutter, M. (2020). Rolling in the deep–hybrid locomotion for wheeled-legged robots using online trajectory optimization. *IEEE Robotics and Automation Letters*, 5(2):3626–3633.

Bouman, A., Ginting, M. F., Alatur, N., Palieri, M., Fan, D. D., Touma, T., Pailevanian, T., Kim, S.-K., Otsu, K., Burdick, J., et al. (2020). Autonomous spot: Long-range autonomous exploration of extreme environments with legged locomotion. *arXiv preprint arXiv:2010.09259*.

Burgard, W., Moors, M., Fox, D., Simmons, R., and Thrun, S. (2000). Collaborative multi-robot exploration. In *ICRA*, pages 476–481.

Chen, Y. and Medioni, G. (1992). Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155.

Čížek, P. and Faigl, J. (2019). Dynamic building of wireless communication infrastructure in underground environments. In *Modelling and Simulation for Intelligent Systems (MESAS)*.

Dang, T., Khattak, S., Mascarich, F., and Alexis, K. (2019). Explore locally, plan globally: A path planning framework for autonomous robotic exploration in subterranean environments. In *2019 19th International Conference on Advanced Robotics (ICAR)*, pages 9–16.

DARPA (2019). Competition rules tunnel circuit.

DARPA1 (2019). Rolling, Walking, Flying, and Floating, SubT Challenge Teams Traverse the Tunnel Circuit.

DARPA2 (2020). Teams CoSTAR and BARCS Take Top Spots in DARPA Subterranean Challenge Urban Circuit.

Dillmann, R. (2004). Ka 1.10 benchmarks for robotics research. *European Robotics Network (EURON), IST-2000-26048*.

Faessler, M., Fontana, F., Forster, C., Mueggler, E., Pizzoli, M., and Scaramuzza, D. (2016). Autonomous, vision-based flight and live dense 3D mapping with a quadrotor micro aerial vehicle. *Journal of Field Robotics*, 33(4):431–450.

Faigl, J. et al. (2016). On localization and mapping with rgb-d sensor and hexapod walking robot in rough terrains. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 002273–002278. IEEE.

Felzenszwalb, P. F. and Huttenlocher, D. P. (2012). Distance transforms of sampled functions. *Theory of Computing*, 8:415–428.

Huang, Y.-W., Lu, C.-L., Chen, K.-L., Ser, P.-S., Huang, J.-T., Shen, Y.-C., Chen, P.-W., Chang, P.-K., Lee, S.-C., and Wang, H.-C. (2019). Duckiefloat: a collision-tolerant resource-constrained blimp for long-term autonomy in subterranean environments. *arXiv preprint arXiv:1910.14275*.

Huber, D. F. and Vandapel, N. (2006). Automatic three-dimensional underground mine mapping. *The International Journal of Robotics Research*, 25(1):7–17.

Hwangbo, J., Sa, I., Siegwart, R., and Hutter, M. (2017). Control of a quadrotor with reinforcement learning. *IEEE Robotics and Automation Letters*, 2:2096–2103.

Iša, J. and Dlouhý, M. (2010). Robotour-robotika. CZ outdoor delivery challenge. In *Proceedings of the 1st Slovak-Austrian International Conference on Robotics in Education, Bratislava, Slovakia.* Citeseer.

Kohlbrecher, S., Von Stryk, O., Meyer, J., and Klingauf, U. (2011). A flexible and scalable slam system with full 3d motion estimation. In *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, pages 155–160. IEEE.

Kruijff, G.-J. M., Kruijff-Korbayová, I., Keshavdas, S., Larochelle, B., Janíček, M., Colas, F., Liu, M., Pomerleau, F., Siegwart, R., Neerincx, M., et al. (2014). Designing, developing, and deploying systems to support human-robot teams in disaster response. *Advanced Robotics*, 28(23):1547–1570.

Kruijff-Korbayová, I., Colas, F., Gianni, M., Pirri, F., de Greeff, J., Hindriks, K., Neerincx, M., Ögren, P., Svoboda, T., and Worst, R. (2015). Tradr project: Long-term human-robot teaming for robot assisted disaster response. *KI-Künstliche Intelligenz*, 29(2):193–201.

Kubelka, V. and Reinstein, M. (2012). Complementary filtering approach to orientation estimation using inertial sensors only. In *2012 IEEE International Conference on Robotics and Automation*, pages 599–605.

Kubelka, V., Reinstein, M., and Svoboda, T. (2019). Tracked robot odometry for obstacle traversal in sensory deprived environment. *IEEE/ASME Transactions on Mechatronics*, 24(6):2745–2755.

Lier, F., Hanheide, M., Natale, L., Schulz, S., Weisz, J., Wachsmuth, S., and Wrede, S. (2016). Towards automated system and experiment reproduction in robotics. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3298–3305. IEEE.

Maity, A., Majumder, S., and Ray, D. N. (2013). Amphibian subterranean robot for mine exploration. In *2013 International Conference on Robotics, Biomimetics, Intelligent Computational Systems*. IEEE.

Miller, I. D., Cladera, F., Cowley, A., Shivakumar, S. S., Lee, E. S., Jarin-Lipschitz, L., Bhat, A., Rodrigues, N., Zhou, A., Cohen, A., et al. (2020). Mine tunnel exploration using multiple quadrupedal robots. *IEEE Robotics and Automation Letters*, 5(2):2840–2847.

Morris, A., Ferrguson, D., Omohundro, Z., Bradley, D., Silver, D., Baker, C., Thayer, S., Whittaker, C., and Whittaker, W. (2006). Recent developments in subterranean robotics. *Journal of Field Robotics*, 23(1):35–57.

Muja, M. and Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. In *Proc. Internat. Conf. on Computer Vision Theory and Applications (VISAPP'09)*, pages 331–340.

Mur-Artal, R., Montiel, J. M. M., and Tardós, J. D. (2015). Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163.

Murphy, R. R. (2014). *Disaster Robotics*. The MIT Press.

Murphy, R. R., Kravitz, J., Stover, S. L., and Shoureshi, R. (2009). Mobile robots in mine rescue and recovery. *IEEE Robotics & Automation Magazine*, 16(2):91–103.

Neumann, T., Ferrein, A., Kallweit, S., and Scholl, I. (2014). Towards a mobile mapping robot for underground mines. In *Proceedings of the 2014 PRASA, RobMech and AfLaT International Joint Symposium, Cape Town, South Africa*, pages 27–28.

Obdrzalek, D. (2010). Eurobot junior and starter-a comparison of two approaches for robotic contest organization. *Robotics in Education, Bratislava*.

Pecka, M., Šalanský, V., Zimmermann, K., and Svoboda, T. (2016). Autonomous flipper control with safety constraints. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2889–2894

Pecka, M., Zimmermann, K., Petrlík, M., and Svoboda, T. (2018). Data-driven policy transfer with imprecise perception simulation. *IEEE Robotics and Automation Letters*, 3(4):3916–3921.

Petrlík, M., Báča, T., Heřt, D., Vrba, M., Krajník, T., and Saska, M. (2020). A robust UAV system for operations in a constrained environment. *IEEE Robotics and Automation Letters (RAL)*.

Pomerleau, F., Colas, F., Siegwart, R., and Magnenat, S. (2013). Comparing ICP variants on real-world data sets. *Autonomous Robots*, 34(3):133–148.

Pomerleau, F., Krusi, P., Colas, F., Furgale, P., and Siegwart, R. (2014). Long-term 3D map maintenance in dynamic environments. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 3712–3719.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. (2009). Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, volume 3.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788.

Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Rusu, S. R., Hayes, M. J. D., and Marshall, J. A. (2011). Localization in large-scale underground environments with RFID. In *2011 24th Canadian Conference on Electrical and Computer Engineering (CCECE)*. IEEE.

Santos, J. M., Couceiro, M. S., Portugal, D., and Rocha, R. P. (2015). A sensor fusion layer to cope with reduced visibility in slam. *Journal of Intelligent & Robotic Systems*, 80(3-4):401–422.

Santos, J. M., Krajník, T., Fentanes, J. P., and Duckett, T. (2016). Lifelong information-driven exploration to complete and refine 4-d spatio-temporal maps. *IEEE Robotics and Automation Letters*, 1(2):684–691.

Santos, J. M., Portugal, D., and Rocha, R. P. (2013). An evaluation of 2D slam techniques available in robot operating system. In *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–6. IEEE.

Saska, M., Krajník, T., and Přeučil, L. (2012). Cooperative $\mu UAV-UGV$ autonomous indoor surveillance. In *International Multi-Conference on Systems, Signals & Devices*, pages 1–6. IEEE.

Schneider, S., Hegger, F., Ahmad, A., Awaad, I., Amigoni, F., Berghofer, J., Bischoff, R., Bonarini, A., Dwiputra, R., Fontana, G., et al. (2014). The rockin@ home challenge. In *ISR/Robotik 2014; 41st International Symposium on Robotics*, pages 1–7. VDE.

Shin, Y.-S., Park, Y., and Kim, A. (2018). Direct visual slam using sparse depth for camera-lidar system. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8.

Siek, J., Lee, L.-Q., and Lumsdaine, A. (2001). *The Boost Graph Library: User Guide and Reference Manual*. Addison-Wesley.

Simanek, J., Reinstein, M., and Kubelka, V. (2015). Evaluation of the EKF-based estimation architectures for data fusion in mobile robots. *IEEE/ASME Transactions on Mechatronics*, 20(2):985–990.

Spurný, V., Báča, T., Saska, M., Pěnička, R., Krajník, T., Thomas, J., Thakur, D., Loianno, G., and Kumar, V. (2018). Cooperative autonomous search, grasping, and delivering in a treasure hunt scenario by a team of unmanned aerial vehicles. *Journal of Field Robotics*, 36(1):125–148.

Tarrí-o, P., Barbolla, A., and Casar, J. (2011). Weighted least squares techniques for improved received signal strength based localization. *Sensors (Basel, Switzerland)*, 11:8569–92.

Tatum, M. (2020). Communications coverage in unknown underground environments. Master's thesis, Carnegie Mellon University.

Thrun, S., Hähnel, D., Ferguson, D., Montemerlo, M., Triebel, R., Burgard, W., Baker, C., Omohundro, Z., Thayer, S., and Whittaker, W. (2003). A system for volumetric robotic mapping of abandoned mines. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Tomáš, R. (2020). Sensor fusion for object localisation in adverse conditions for mobile robots. Master's thesis, České vysoké učení technické v Praze. Vypočetní a informační centrum.

Vrba, M. and Saska, M. (2020). Marker-less micro aerial vehicle detection and localization using convolutional neural networks. *IEEE Robotics and Automation Letters*, PP:1–1.

Wang, W., Dong, W., Su, Y., Wu, D., and Du, Z. (2014). Development of search-and-rescue robots for underground coal mine applications. *Journal of Field Robotics*, 31(3):386–407.

Wong, C.-C., Wang, W.-W., and Lee, Y.-L. (2005). Soccer robot design for FIRA mirosot league. In *IEEE International Conference on Mechatronics, 2005. ICM'05.*, pages 457–460. IEEE.

Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, volume 97, page 146.

Ye, H., Chen, Y., and Liu, M. (2019). Tightly coupled 3D lidar inertial odometry and mapping. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3144–3150.

Zhang, J. and Singh, S. (2014). Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems*, volume 2.

Zhang, J. and Singh, S. (2015). Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 2015.

Zhao, J., Liu, G., Liu, Y., and Zhu, Y. (2008). Research on the application of a marsupial robot for coal mine rescue. In *International Conference on Intelligent Robotics and Applications*, pages 1127–1136. Springer.

Zoula, M., Prágr, M., and Faigl, J. (2021). On building communication maps in subterranean environments. In *Modelling and Simulation for Autonomous Systems: 7th International Conference, MESAS 2020, Prague, Czech Republic, October 21, 2020, Revised Selected Papers 7*, pages 15–28. Springer.

How to cite this article: Rouček, T., Pecka, M., Čížek, P., Petříček, T., Bayer, J., Šalanský, V., Azayev, T., Heřt, D., Petrlík, M., Báča, T., Spurný, V., Krátký, V., Petráček, P., Baril, D., Vaidis, M., Kubelka, V., Pomerleau, F., Faigl, J., Zimmermann, K., Saska, M., Svoboda, T., & Krajník T. (2022). System for multi-robotic exploration of underground environments CTU-CRAS-NORLAB in the DARPA subterranean challenge. *Field Robotics, 2*, 1779–1818.