ARL-TN-1274 ● SEP 2025

Distributed and Collaborative Intelligent Systems and Technology Collaborative Research Alliance Fall 2024 Integration Testing Results

by Craig Lennon

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TN-1274 ● SEP 2025

Distributed and Collaborative Intelligent Systems and Technology Collaborative Research Alliance Fall 2024 Integration Testing Results

Craig Lennon
DEVCOM Army Research Laboratory

REPORT DOCUMENTATION PAGE

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
		START DATE	END DATE
September 2025	Technical Note	21 October 2024	1 November 2024

4. TITLE AND SUBTITLE
Distributed and Collaborative Intelligent Systems and Technology Collaborative Research Alliance Fall 2024 Integration Testing Results

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER
5d. PROJECT NUMBER	**5e. TASK NUMBER**	**5f. WORK UNIT NUMBER**

6. AUTHOR(S)
Craig Lennon

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
DEVCOM Army Research Laboratory
ATTN: FCDD-RLA-JC
Aberdeen Proving Ground, MD 21005

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TN-1274

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

13. SUPPLEMENTARY NOTES
ORCID: Craig Lennon, 0000-0002-2229-2919

14. ABSTRACT
In Fall 2024, ARL worked with researchers from the Distributed and Collaborative Intelligent Systems and Technology Collaborative Research Alliance to integrate ongoing research onto robotic platforms at Camp Buckner, New York. This report documents the results of integration testing for four research topics: collaborative planning in uncertain environments, natural language-based goal assignment, language-based task assignment with incomplete specifications, and team movement planning based on visibility and communication.

15. SUBJECT TERMS
autonomy, autonomous systems, robotics, collaborative robotics, distributed perception and control, Military Information Sciences

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT	b. ABSTRACT	C. THIS PAGE		
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UU	43

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
Craig Lennon	(520) 691-6117

STANDARD FORM 298 (REV. 5/2020)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

List of Tables

1. Introduction

The Distributed and Collaborative Intelligent Systems and Technology (DCIST) Collaborative Research Alliance (CRA) conducts basic and applied research in distributed intelligence, heterogeneous group control, and adaptive and resilient behaviors to "enable robotics and autonomous systems operations across all Army-relevant environments; provide better situational awareness; increase Warfighter capabilities and standoff; increase coverage and create dilemmas for the adversary; provide force multiplication; enable faster decision-making; and extend maneuverability" (Piekarski et al. 2022). Researchers participating in DCIST CRA joined the U.S. Army Combat Capabilities Development Command Army Research Laboratory at West Point, New York, from 21 October to 2 November 2024, to integrate their research with robotic platforms and test the integration at Camp Buckner, New York, which is a training area of West Point. This report documents four integration tests conducted during the Fall 2024 event. The research for which integration was performed is presented in the following list, with a project title in italics, followed by a shorter title in parentheses, and then a brief description of the project. The shorter title will be used to refer to the project throughout the rest of this report.

- *Language-Driven Task Planning, Execution, and Monitoring in Changing Environments* (planning with uncertainty): Enable a collaborative multiagent team to efficiently navigate in and map an uncertain environment, given an overhead image as a priori information and commander-specified goals.
- *Language-Driven Task Planning, Execution, and Monitoring in Changing Environments* (language-based goal assignment)*:* Demonstrate collaborative multi-robot perception-enabled task and motion planning using hierarchical maps to ground natural language to the planning problem.
- *Distributed Online Semantic Mapping and Planning with Heterogeneous Robots Given Underdetermined Mission Specifications* (under-specified language commands): Deconstruct missions into manageable tasks for each robot and execute plans generated by language models in unknown and unstructured environments.
- *Communication-Aware Robot Coordination – Planning, Networking, and Control for Resilient Missions in Contested Areas* (exposure-conscious path planning): Plan multi-robot missions in a contested environment while maintaining communication. As part of this task, a robot must plan to maximize line of sight (LOS) coverage of specific regions or agents while minimizing detection risk.

Integration tests were conducted in the areas shown in Figure 1. The tactical operations center, which served as the administrative center of the integration activity, is indicated by a yellow star. The test sites for planning with uncertainty and language-based goal assignment are outlined in red and blue, respectively, although some tests crossed between regions. Exposure-conscious path planning was tested at the site outlined in green. Under-specified language commands were tested at both Camp Buckner and Range 15 (a training area), mostly in the region outlined by an orange box in Figure 2.

Sections 2–5 describe the four research projects, with each section briefly describing the research, the site at which the project conducted tests, and the tests ran in Fall 2024. The original integration tests, planned out before integration began, are included in Appendix A, while Sections 2–5 describe the results of the tests that were conducted, which sometimes differ from the planned tests in Appendix A. Additional range facilities at West Point that are promising sites for experimentation, but which were not used for testing these four projects, are described in Appendix B.

Figure 1. Test regions at Camp Buckner.

Figure 2. Test site for under-specified commands at Range 15.

2. Planning with Uncertainty (Phase 1)

Research was conducted by the Massachusetts Institute of Technology for both planning with uncertainty and language-based goal assignment, and these projects interact as two phases of planning (Figure 3). Beginning with an overhead image and annotations related to the mission, a multiagent plan is developed and executed in Phase 1 (planning with uncertainty). At the end of this phase, each agent has reached its goal, having built its own map. Phase 2 (language-based planning) begins with the fusing of these maps. Sharing a common map allows multiple agents to be provided with mission tasks in natural language, translated into individual goals for the robots that they execute. The remainder of this section explains the planning with uncertainty project (Phase 1), while language-based goal assignment (Phase 2) is explained in Section 3.

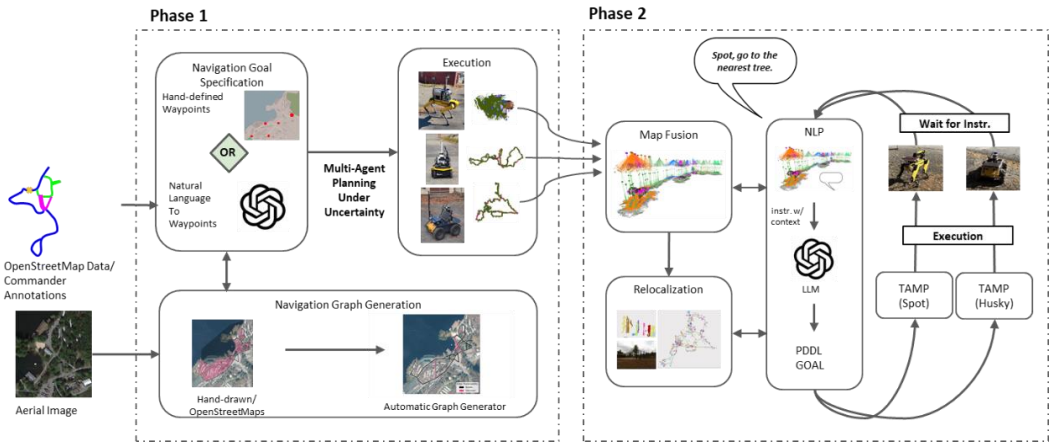

Figure 3. System architecture for planning with uncertainty and language-based goal assignment.

The planning with uncertainty project team researched team route planning over terrain with varying and uncertain traversability. The route planning process used in the project begins with an overhead image, algorithmically segmented into regions based on terrain and distance. These regions are identified with vertices in a graph and connected by edges with adjacent regions. Edges may be traversable by any platform, no platform, or platforms with a certain property (e.g., legged platforms). The edges may have uncertain traversability. A team of robots moves through the environment, possibly starting at different positions and proceeding along different paths to different goal vertices. Robots on the team may have different capabilities, including the ability to traverse different types of terrain and the ability to identify whether an edge (route between regions) is traversable, allowing the robot to resolve the uncertainty associated with planning routes along the graph.

Given the graph and robots with assigned start positions and goals, two methods of planning are compared. In each, an algorithm plans paths for the team to traverse the graph from start to end positions. With the baseline method, each robot independently plans a path that is optimal with respect to time, replanning as needed when its own observations resolve the uncertainty associated with edge traversability. The method developed under the planning with uncertainty project plans jointly over all robots, developing a plan that is optimal with respect to the time required for all robots to traverse from their start to end positions. Detailed descriptions of the collaborative and baseline planning are contained in Kurtz et al. (2024). The Fall 2024 tests compared the time required for all robots on the team to complete their paths from start to goal to measure the benefit of planning under uncertainty. In the Fall 2024 experiments, robots reached their goals within an average of 562 s when using collaborative planning, versus an average of 640 s for noncollaborative planning. This made for an average improvement of 12% over the

three runs. The time savings of collaborative versus noncollaborative planning depends both on the graph to be traversed and the correct function of the algorithm. Thus, the 12% improvement result is valuable as evidence that collaborative planning saves time rather than as an estimate of how much time one can expect to be saved. The example run in Figure 4 (Strader 2025) shows how time savings is caused by changes in navigation. In this run (Figure 4, left), the Spot starts at the blue circle and proceeds directly to the blue star when using noncollaborative planning. In collaborative planning (Figure 4, right), it examines an edge along which the Husky (shown in red) intends to travel. As the Spot observes the edge to be blocked, the Husky does not attempt to proceed along that edge, which results in the overall savings of time.

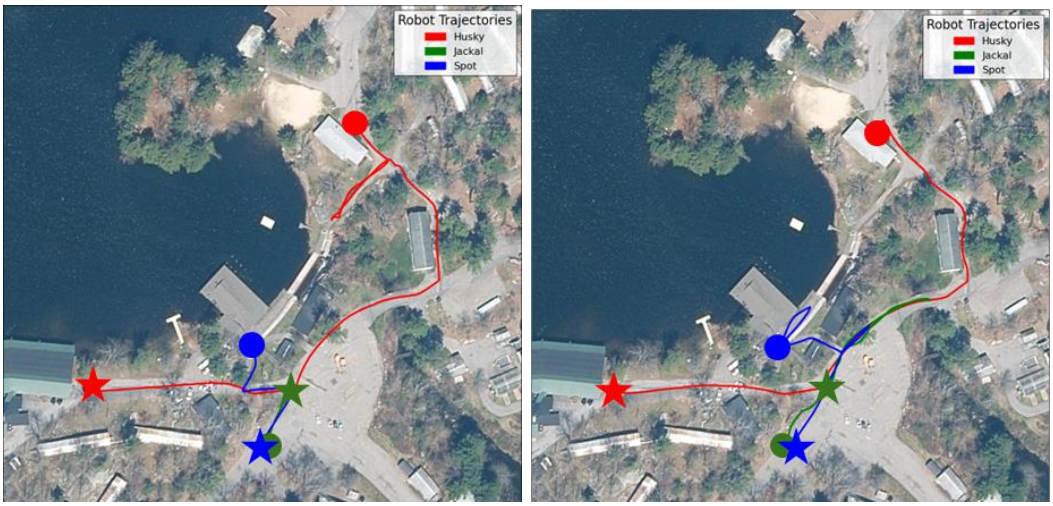

Figure 4. Noncollaborative planning (left) vs. collaborative planning (right).

2.1 Description of the Platforms

The three platforms used in the experiment are shown in Figure 5. The Jackal and Husky are manufactured by Clearpath Robotics, and the Spot is manufactured by Boston Dynamics. All robots use a real-time kinematic GPS receiver to aid in localization, Intel RealSense and light detection and ranging (LiDAR) to aid in navigation, and a Silvus radio network for communication.

Figure 5. From left to right: the Jackal, Husky, and Spot used in the tests (reprinted from Strader 2025).

2.2 Summary of Test Results

Eight tests (some consisting of multiple runs) were used to evaluate the integration of the joint planning algorithm with other robot functions. The tests are listed in Table A-1, but the following list summarizes the functionality that passed integration testing:

- An overhead image was used to produce a graph with regions automatically segmented into vertices and adjacent regions linked by an edge. The planner used the graph as input.
- Each individual robot generated a path along the graph and followed it using a global planner for establishing waypoints and local planner for navigating between waypoints.
- A collaborative plan was generated for up to three robots.

The 12% time savings mentioned previously referred to two pairs of three runs, all following the pattern in Figure 4. All runs used the same routes, with collaborative and noncollaborative planning each getting three runs. Table 1 (Strader 2025) shows their results, with the average time (in seconds) used by the algorithm to plan and execute. Planning time refers to time spent generating global plans before movement starts. Execution time refers to time after movement has started. Total time is the sum of planning and execution. The standard deviation (SD) for each time is listed, and in each case is small relative to the average, indicating consistent performance across runs.

Table 1. Results for planning under uncertainty.

	Noncollaborative			Collaborative		
	Total	**Execution**	**Planning**	**Total**	**Execution**	**Planning**
Average (s)	640.18	635.68	4.5	561.58	531.35	30.23
SD (s)	32.77	32.84	0.15	6	6.68	0.68

3. Language-Based Goal Assignment (Phase 2)

The Fall 2024 tests examined two robots' abilities to 1) build a scene graph using methods described in Hughs et al. (2024) and Strader et al. (2024), 2) localize in that scene graph, and 3) autonomously plan and execute a sequence of actions provided in natural language and translated to a planning domain definition language (PDDL) by GPT-4, a large language model (LLM). Before natural language to PDDL tests were run, three scene graphs were generated of the environment, which were fused into one scene graph. The environment in which they operated is shown in Figure 6. The left side of this figure shows the routes the robots traversed during the experiments. The scene graph decomposes the environment into mesh, objects, places, and regions, all of which may have properties associated with them such as position or semantic labels (e.g., terrain type, object type, and region type). While the full scene graph cannot be shown in a single image, Figure 7 shows key aspects of the graph. A red arrow points to the mesh, which was overlaid with color from the images used to construct it. The yellow arrow points to objects that are represented as nodes with their bounding boxes. The green arrow points to places, which are the polygons resulting from separating the mesh into surfaces based on traversability. Dashed lines between the mesh and objects identify the location (bounding box) of the objects in the mesh layer, while dashed lines between objects and places identify the location of objects in the places layer of the scene graph. The blue lines emanating from nodes within the place-layer polygons join pairs of places between which the robot can traverse. Figure 7 does not show regions, which are collections of places. The red X marks the goal location of the robot, corresponding to the red X on the right side of Figure 6.

The scene graph is transmitted as context to the LLM in JavaScript Object Notation (JSON) format, along with examples of proper PDDL output. A natural language instruction is also provided, such as "go to the vehicle next to the sidewalk." The LLM produces a PDDL goal as output that references locations (e.g., places) in the scene graph near objects or within regions, as specified in the instructions. This goal is then sent to a PDDL planner that plans a sequence of actions to achieve the goal.

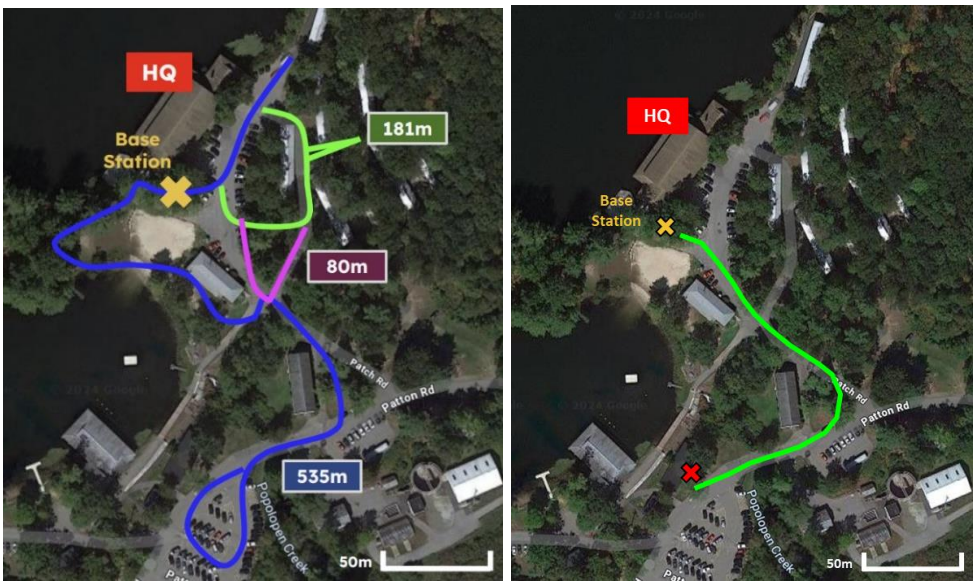

Figure 6. Area of operations for language-based goal assignment (reprinted from Strader 2025).

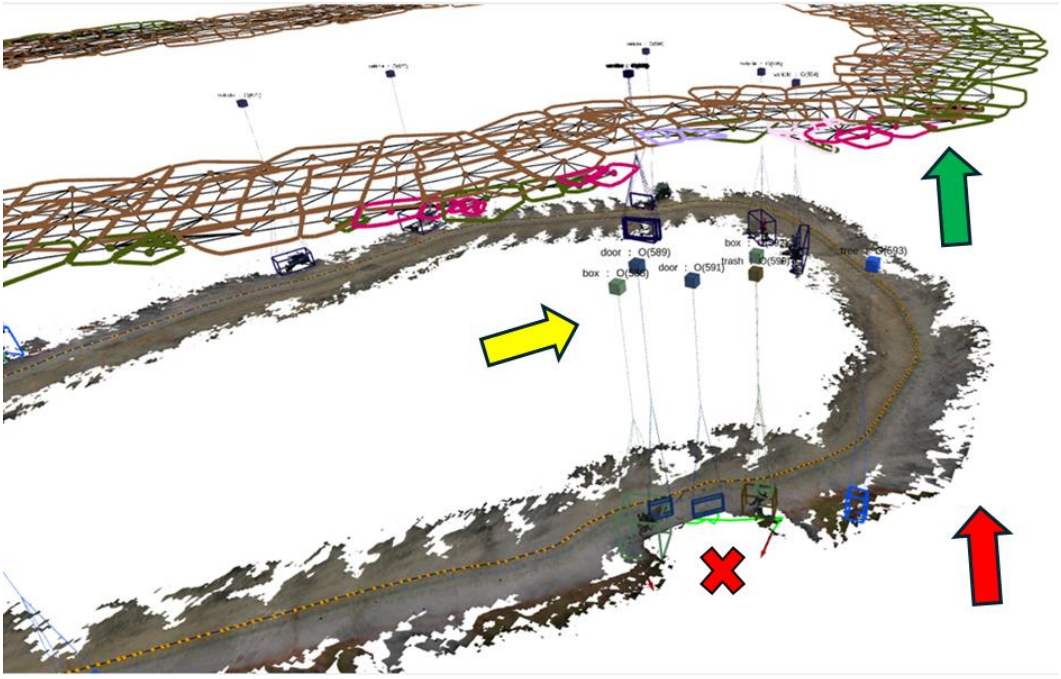

Figure 7. Scene graph with arrows pointing to places (green), objects (yellow), and mesh (red) (reprinted from Strader 2025).

3.1 Description of the Platforms

The Spot and Husky (Figure 8) both carry a laptop with a graphics processing unit for compute, and both use an Intel RealSense D455 for capturing the color and depth images needed to recognize objects and regions in the environment. The

robots use Silvus radios to communicate with a base station laptop that uses the internet to access GPT-4.

Figure 8. The Spot and Husky used in the tests (Strader 2025).

3.2 Test Results

The following functions were successfully tested in Fall 2024:

- Three scene graphs were fused into one scene graph.
- Robots reestablished their positions (relocalized) on their version of the fused map.
- A single robot planned and navigated to execute a command translated by an LLM from natural language to a PDDL goal.
- Two robots planned and navigated to execute a command translated by an LLM from natural language to a PDDL goal.

The test card describing the runs is included in Table A-2. An example of the context provided to the LLM is in Section 3.3. The natural language instructions tested during the experiments varied between easy, medium, and hard, based on the complexity of the instruction for both single-robot and multi-robot scenarios. An example of an easy instruction is “Spot, go to the nearest tree. Husky, go to the nearest vehicle.” An example of a hard instruction is “The bag in the shelter and the trash are both suspicious. I need you two to secure the area. Then rendezvous at the tree in the road.”

For the single-robot tests (eight trials), the robot was evaluated on grounding, planning, and executing. Grounding means correctly identifying the correct PDDL goal in the map, planning means planning a path to reach the PDDL goal, and executing means traversing the path correctly to reach the goal. The success rates for grounding, planning, and executing are shown in Table 2. The improvement in success rate from the single to multi-robot tests was unexpected, and could be caused by errors in the system being corrected throughout the testing, as bugs in the software were being fixed through the integration process. Table 2 summarizes

the results, with the first column showing the number of robots used, the second summarizing the objective to be tested, the third recording the number of trials for that run, and the fourth showing success at grounding, planning, and execution. Each test objective identifies the number of robots used and the difficulty of commands and goals. Grounding success indicates that the correct goal object was identified, planning means a plan was generated to the correct goal, and execution means the goal was reached. Success rate is rounded to the nearest percent.

Table 2. Summary of test results (adapted from Strader 2025).

No. of agents	Objective	No. of trials	Results (average success rate) grounding/planning/execution (%)
1	Single-robot, easy language commands and goals	2	100/100/100
1	Single-robot, medium language commands and goals	2	100/50/50
1	Single-robot, hard language commands and goals	4	75/75/50
2	Multirobot	Easy = 7 Medium = 2 Hard = 4	100/85/28 100/100/100 100/100/75

3.3 Context Provided to the Model

This section contains an example of the context provided to the LLM in the easy single-robot command "Spot, check out the bag." Without editing, the context would be 16 pages, so we provide only examples taken from the context. The instructions sent to the robot are written in a different font and placed within quotes to make distinguishing them easier.

First, the LLM was provided with the following instructions as to what it should do:

"You are a helpful assistant who is an expert at mapping from natural language commands to Planning Domain Definition Language (PDDL) goal predicates to be executed by a team of robots. Given an instruction and a description of the world, your task is to generate PDDL goals for each robot in the team. The robots may have different capabilities. If the instruction does not explicitly assign instructions to specific robots, you are responsible for deciding which robot completes which instruction. The PDDL goal predicates should be grounded

in the world. The world is described using a 3D Scene Graph that consists of Objects, Places, and Regions. Objects are objects in the world and consist of a semantic label, an index, an x-y coordinate position, and a parent Region index. Places are small polygons that represent patches of the world and can have a semantic label. Since there are many Places in a Scene Graph, you will only be told about a subset of important Places. Regions are large parts of the world that represent different kinds of areas and contain Objects. The PDDL domain consists of Actions and Predicates. Actions describe the way that the robots can interact with the world. Predicates are boolean properties describing the state of the world. Predicates can be negated by adding `not` to the front of them. PDDL goals are the Predicates that should be true after completing the instruction. Given an instruction, tell me the PDDL goals that correspond to the instruction and which robots should accomplish each goal. You will be provided with a description of the PDDL domain (Predicates and Actions) and a description of the available robots. Each of these will be delimited by XML tags. You will then be given an example World (Objects, Places, and Regions) and several examples of instructions mapped to PDDL goals for that World. The Objects, Places, and Regions will be delimited by XML tags. After the examples, you will be given a new description of a World and a new instruciton to map to PDDL goals. The output format must be in JSON and formatted as a Python dictionary of the form: {\"robot id\" : \"PDDL goal\", \"robot id\" : \"PDDL goal\"}."

Second, the LLM was provided with descriptions of the roles of the robots on whose behalf it would be planning.

"<Robots>\nspot: spot is a quadraped robot. It is able to move and inspect objects. It has the following unique id: `spot`.\nhamilton: hamilton is a ground vehicle robot. It is able to move, but it cannot inspect objects. It has the following unique id: `hamilton`"

Next, PDDL predicates that could be used in creating goals were introduced. The following are two such examples:

"</Robots>\n<Predicates>\n(VisitedPlace ?P): takes one Place ?P and returns True if the robot has ever been the the Place ?P.\n

(Safe ?O): takes one Object ?O and returns True if the Object ?O is safe"

Natural language instructions were then related to the predicates.

"an instruction does not care about the order of execution, use (VisitedPlace ?P) or (VisitedObject ?O). If the order matters, use (AtPlace ?P) or (AtObject ?O).

Inspect is an action that changes a Suspicious Object into a Safe Object that is not Suspicious"

Examples of mapping robots to PDDL goals were provided. This started with a list (provided in the format shown below) of objects in the world. The number in O(number) is the identification number of the object, type is the semantic class, region R(number) refers to an area in the map, and pos is the position in a metric reference to the object's center.

"(idx=O(2), type=door, region_parent=R(0), pos=(3.335,3.482))

(idx=O(5), type=boat, region_parent=R(1), pos=(1.34,3.28))"

Next, 15 examples of input natural language instructions and correct output PPDL goals were provided in pairs, with 2 pairs shown here:

"Hamilton, move over to the door.

{\hamilton\: \(VisitedObject O(2))\}

"Spot, drive over to the door and inspect the boat.

{\spot\: \(and (VisitedObject O(2)) (Safe O(5)))\}"

A database listing all regions and objects in the robot's map was provided. There were 737 distinct IDs, with 2 shown here:

"<Regions>(id=R(1),type=unknown, pos=(-24.537,20.099)

<Objects>(id=O(2),type=tree, pos=(-0.79,6.112), parent_region_id=['R(7)'])”

Finally, the natural language command was provided.

“Spot, check out the bag”

At this point, the LLM translated the natural language goal into a PDDL goal, which is sent to a PDDL planner to generate a plan for the robot.

4. Under-Specified Language Commands with Incomplete Maps

Under-specified language commands research done by University of Pennsylvania also used GPT-4 to translate natural language commands into a sequence of robotic actions, but at a lower level of specificity than the research in Section 3, thereby delegating more of the responsibility for how to execute the task to the LLM. Language-based goal assignment was part of a two-phase process in which the LLM assigns goals based on an existing map created prior to the command being issued (Figure 3). Under-specified language commands research does not require a full map but incorporates exploration into the set of behaviors the LLM may select to execute after the command is given. In both cases, the process starts with a scene graph provided as context to the LLM in a JSON format, a task to be completed, and instructions about the expected form of output. This research differs from Section 3 because the focus is on tasks in which not all features of the environment are known when the task is provided to the robot. Consequently, the LLM (repeatedly) provides the next action for the robot to take rather than an end goal towards which a sequence of actions can be planned. In turn, this means the LLM must interface with the perception subsystem to be informed of the presence of new objects and also with the control subsystem to direct the robot to new regions. The set of algorithms that handle the interaction of perception, control, and human interface with the LLM is referred to as SPINE and described as “online semantic planning for missions with incomplete natural language specifications” in unstructured environments (Ravichandran et al. 2024; Cladera et al. 2025). A detailed description of how SPINE interfaces with perception and control, as well as the library of behaviors from which the LLM may choose, can be found in Cladera et al. (2025).

The Fall 2024 experiments, held at West Point Ranges 15 and 16, are shown in Figure 9 with the red circle indicating the site of the base station through which robots communicated with cloud-based GPT-4.

The task provided to the LLM was not completely specified, meaning there might be many different things the robot could do to satisfy it. For example, one task was "I lost my robot when it was filling its battery. It is red. Can you find it and tell me its status?" The task's context for the LLM is provided in Section 4.3.

Figure 9. Overhead image of Ranges 15 and 16.

4.1 Description of the Platform

A Jackal (Figure 10) was used as the platform for the tests. An Ouster LiDAR was used for obstacle avoidance and odometry, and a ZED stereo camera was used for object detection. The GPS antenna and Ublox were used to establish a common frame of reference within a map created by aerial robots, and the Rajant wireless mesh node was used to link the robot to the cloud-based GPT LLM.

Figure 10. The Jackal used in under-specified command tests.

4.2 Test Results

The following functions were tested successfully:

- The robot was able to connect to a local computer and interface with cloud-based GPT-4.
- The robot's perception and control worked while the connection with the local computer is maintained.
- The robot loaded a full map of the environment and executed a mission using this map, i.e., it executed actions provided by the LLM.
- The robot received a partial map of the environment and executed a mission using this map, i.e., it executed actions provided by the LLM and updated the map as needed.

The test card describing the runs is included in Table A-3. The runs testing the four functions above were all completed successfully. As the commands were not completely specified, runs were graded against subjective expectations of proper robot behavior. Six specifications were tested and are quoted in the following text, followed by a description of the expected behavior and then a description of what the robot did. The robot was generally successful in the first five specifications, but challenged by the sixth.

1. "I heard of activity near the red houses. Go check." The robot was expected to go to a region known to contain red houses, explore the region, and report three people near the houses. The robot went to the correct area and detected and reported 1 of 3 people in the area.
2. "Go inspect the black vehicle at the end of the driveway." The robot was expected to identify that there was no vehicle at the end of the driveway and explore the driveway until it found a black vehicle at the top of the driveway. The robot went to the end of the driveway and (correctly) found no vehicle there. It returned to the top of the driveway where vehicles had previously been mapped. In an attempt to resolve the mission, it queried the perception subsystem to ask if a mapped vehicle (without recorded color) was black. Upon receiving a response that the vehicle was black, the robot correctly reported the presence of a black vehicle at the top of the driveway.
3. "I got reports that the bridge was blocked. Go check." The robot was expected to identify that a vehicle was blocking the bridge. The robot identified and navigated to a bridge which was known a priori but not known to be blocked. It mapped the region around the bridge and correctly reported a person, car, and barrier as blocking the bridge.
4. "Is there activity near the gate?" The planner was expected to go to the gate, which was mapped at the time the command was issued, and detect a person

by the gate. The robot did go to the gate and detect and report a person by the gate.

5. "Identify the parking lot 30 meters east of the bridge." The robot was expected to explore near the bridge, which had been mapped, until it located a parking lot, which had not been mapped. The robot did locate and report the parking lot.
6. "I lost my robot when it was filling its battery. It is red. Can you find it and tell me its status?" The robot did not have any other robots in its map, but correctly deduced that the charging station, which was in its map, was the relevant spot to search. It produced a plan, but the plan was incorrect, after which it needed assistance from the user to fix the plan. We will use this query in Section 4.3 to show the context provided to the robot, as Section 3.3 did for the language-based planning context.

The first five of the previous examples are described by Cladera et al. (2025), who also describe air–ground teaming experiments conducted prior to the Fall 2024 tests. In these experiments, one air and one ground robot start with an incomplete map and an under-specified task and jointly explore, detect, and inspect to complete the task. Prior to the Fall 2024 experiments, a comparison of SPINE versus alternative applications of LLMs to task planning had been carried out as described in Ravichandran et al. (2024). Section 4.4 summarizes those results.

4.3 Context Provided to the Language Model

This section presents the context provided to the robot, edited for brevity. First, the LLM was provided with the robot's role:

```
"You are an excellent graph planner. You must fulfill a
given task provided by the user given an incomplete graph
representation of an environment. You will generate a
step-by-step plan that a robot can follow to solve a
given task. You are only allowed to use the defined API
and nodes observed in the scene graph for planning. Your
plan will provide a list of actions, which will be
realized in a receding-horizon manner. At each step,
only the first action in the plan will be executed. You
will then receive updates, and you have the opportunity
to replan. Updates may include discovered objects or new
regions in the scene graph. The graph may be missing
objects and connections, so some tasks may require you
```

to explore. Exploration means mapping existing regions to find objects, or adding a new region to find paths."

Second, the format in which the map would be provided was described, and an explanation of how the graph related to the physical environment. For example,

"regions: [{name: region_1_name, coords: [west_east_coordinate, south_north_coordinate]}, ...]

region_connections: [[some_region_name, other_region_name], ...]"

"regions is a list of spatial regions. The regions are traversable ONLY IF they appear in the region_connections list"

Next, the required format and expected output of the LLM was provided:

"Provide you plan as a valid JSON string

primary_goal: Explain your primary goal as provided by the user. Reference portions of graph, coordinates, user hints, or anything else that may be useful.

relevant_graph: List nodes or connections in the graph needed to complete your goal. If you need to explore, say unobserved_node(description). List ALL relevant nodes.

reasoning: Explain how you are trying to accomplish this task in detail.

plan: Your intended sequence of actions."

An application programming interface (API) for modifying the scene graph was provided:

"def add_nodes(new_nodes: Dict[str, str]) -> None: Add nodes to graph. Each node is represented as a dictionary."

Followed by an API for actions that could be used as part of a plan:

"def goto(region_node: str) -> None: Navigate to `region_node`. This function uses a graph-search algorithm to find the most efficient path to that node."

Constraints on how the actions should be applied were provided:

"Before calling goto, consider this: goto uses a graph-search algorithm to find an efficient path, so avoid calling goto on intermediate nodes. For example, if you path is ground_2 -> ground_7 -> ground_10. Call goto(ground_10) instead of goto(ground_7), goto(ground_10)"

Then a task was provided, along with the graph needed for the task. The following is an example task:

"task: I lost my robot when it was filling its battery. It is red. Can you find it and tell me its status?"

The scene graph was then provided, which may have relevant objects, such as a charging station. The robot would produce a sequence of actions, which the user may provide feedback on, by suggesting alternative plans. In this example, the robot states its goal and plan, and the user proposes an alternate action:

"primary_goal: Find the users lost robot. The user provided contextual details that will help me find the robot. relevant_graph: charging_station_1

reasoning: The graph does not contain any robots, but the graph does contain a charging station, charging_station_1. The robot may be charging its battery there. I will determine if the nearby regions are connected to the booth.

plan: [goto(example_node_2), goto(example_node_3), goto(example_node_4), goto(example_node_5)]"

The following is the user's feedback:

"Feedback: this is a bad plan. example_node_4 and example_node_5 take you farther away from the goal, charging_station_1. Instead, try to get as close as possible to your goal, then expand the map via extend_map()."

4.4 Prior Evaluations of Under-Specified Language Commands

Ravichandran et al. (2024) compared SPINE against two alternative LLM-based planning applications, "explicit tasking" and "LLM-as-planner." A full map was

provided in both alternatives, as well as step-by-step instructions in the case of explicit tasking and a single complete specification in the case of LLM-as-planner. The planners were compared on efficiency of completing the mission in terms of time and distance, success in completing the mission, the number of human–machine interactions, and the number of queries made to the LLM. Results are summarized below (see Ravichandran et al. [2024] for complete results).

- Efficiency (of distance and time): SPINE and explicit tasking were approximately equal, and both better than the LLM-as-planner.
- Success of mission: explicit tasking and LLM-as-planner were 100% successful in simulation and real life, while SPINE was 94% successful in simulation and 100% in real life.
- Number of human–machine interactions: SPINE and LLM-as-planner were superior to explicit tasking.
- Number of queries to the LLM: LLM-as-planner was superior to both SPINE and explicit tasking.

The experimental conditions, in which a full map was provided and a complete specification was possible to provide at the time at which the task was given, were favorable to LLM-as-planner and explicit tasking. Under-specified language commands would be most useful when either a full map is lacking or complete specification is not possible.

5. Exposure-Conscious Path Planning

The exposure-conscious path planning research created an algorithm that plans a path based on consideration of the regions from which the path would be visible (Hamzezadeh et al. 2024). The algorithm divides an environment into a set of discrete regions and then generates graphs for traversability and visibility. The traversability graph connects two regions with an edge if the robot can traverse from them. The visibility graph connects two regions if one region is within LOS of another. The algorithm then finds a path through the traversability graph that minimizes exposure, defined as the cardinality of the set of regions from which the path is visible. More formally, if $\{R_i : i = 1 \cdots n\}$ is a path on the traversability graph, and we use $v(R_i)$ to denote the set of regions from which region R_i is visible, the exposure along the path is $|\cup_{i=1}^{n} v(R_i)|$. A variation of this calculation would increase cost if the path is visible more than once from a region. The set of regions in the traversable path are a corridor along which any path is equally visible from regions outside of the corridor, which provides the opportunity to navigate so as to avoid obstacles or move in formation without reconsidering visibility.

The Fall 2024 tests were conducted at Camp Buckner within a set of tennis courts that had shipping containers on them. Figure 11 shows the life cycle of the exposure-conscious path planning in action. It starts with an overhead image, for example from a satellite (image no. 1), which is used to create a 3D model of the environment in Unity (image no. 2). The original satellite image (image no. 1) showed a shipping container that was not present at the time of the experiment. In Figure 11, the missing container is covered by a black box in image no. 1 to make it and image no. 2 easier for the reader to match. This was done after the test for presentation and is not part of the processing done for the algorithm. After the Unity environment is created, scripts partition the environment into regions; in this case, square regions represented as pixels in image no. 3. The exposure between each region and traversability between adjacent regions are shown in the top and bottom of image no. 3. In the exposure map, each pixel is colored by its relative visibility for ease of visualization, with lighter colors being more visible. In the traversability map, brighter colors indicate nontraversable regions. The planned path is shown in image no. 4 of Figure 11, with a successful execution performed on a Jackal in image no. 5.

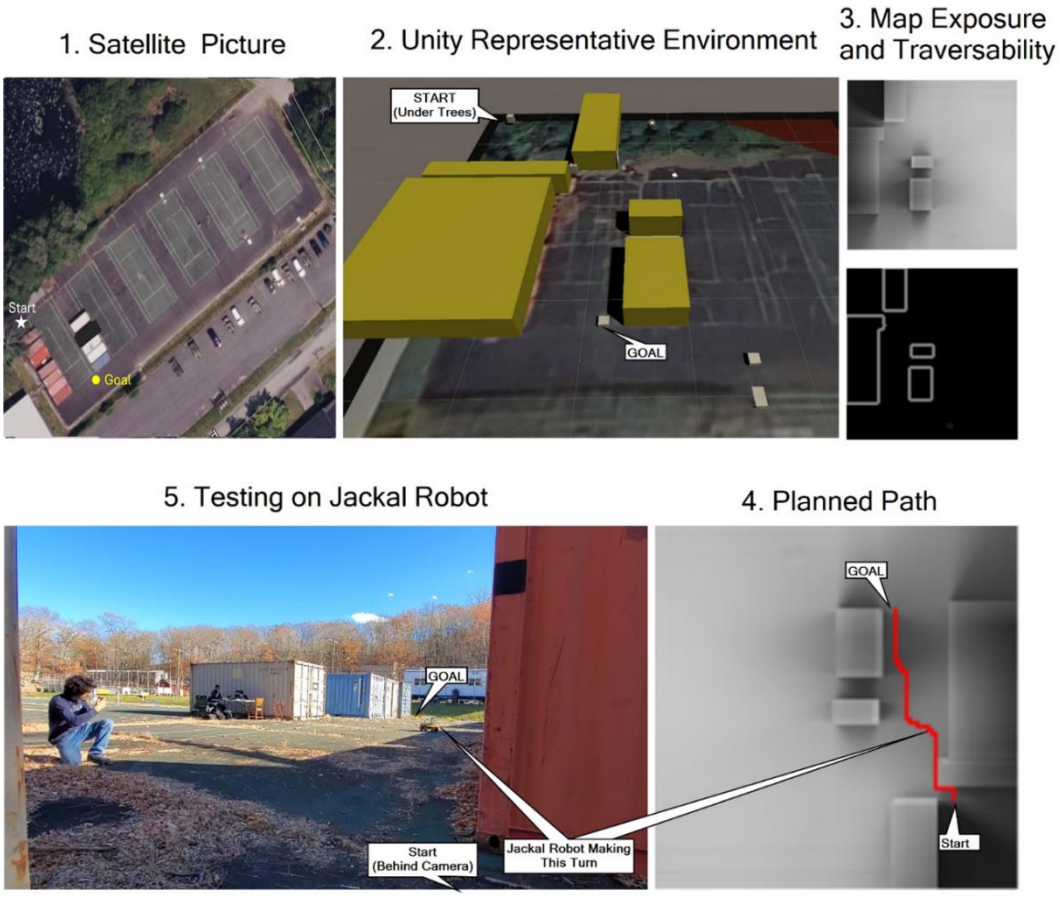

Figure 11. Life cycle of a test run (image courtesy of Eugene Hamzezadeh).

5.1 Description of the Platform

Tests were performed on the Jackal shown in Figure 12. The robot was equipped with an Intel t265 visual-inertial tracking camera and an Intel d455 depth camera and an Ouster OS1 LiDAR sensor that were used for localization and obstacle avoidance.

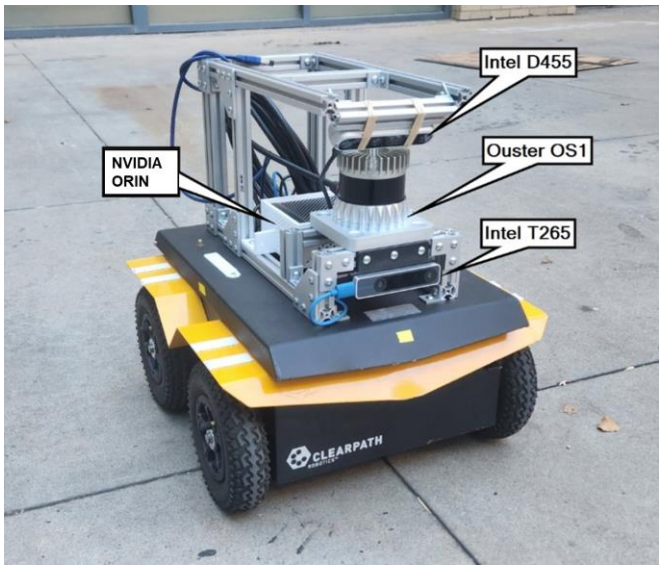

Figure 12. Hardware platform used for experiments (image courtesy of Eugene Hamzezadeh).

5.2 Test Results

The following functionality was successfully tested:

- The robot created exposure information for a real-world environment that matched expected results.
- The robot calculated paths in the environment that minimized exposure to surroundings. Paths made sense intuitively from a tactical movement perspective, i.e., they included tucking corners and minimizing time in open areas.
- The robot executed paths according to plan.

Although multiagent runs were originally planned (Table A-4), the choice to integrate a new controller led to expanded testing of a single robot rather than multiple robots. The run shown in Figure 11 is the basis for concluding that integration was successful.

6. Conclusion

This report documents progress towards integrating DCIST CRA research—including LLM-based research—onto robotic platforms for multiagent experiments. Sections 2 and 3 showed how collaborative planning, based on an overhead image, can save time and reduce distance when compared with noncollaborative planning. The resulting maps can be fused into one map (scene graph), and natural language can be used to designate goals for a robot within that map. Section 4 demonstrated how an LLM can be integrated into an iterative perception and action loop, allowing robots to complete tasks even with incompletely specified criteria. Finally, Section 5 presented a new approach to team movement planning that considers visibility and communication, although initial integration tests used a single robot.

From an experimentation point of view, the application of LLMs to tactical planning in two different ways is worth further consideration. The research described in Section 3 relies on translating natural language into PDDL. With a PDDL planner, the symbol representing a goal is simply a variable, but with an LLM translating natural language into goals, the type of goal object could become relevant to the planning process through the LLM translation. Since the goal assignment is based on context provided to the model, other model properties may influence that assignment, although it is not clear at this time how to establish if this is the case. An analogous issue arises in using the LLM to choose the next action. Examples of how to do this are provided as context, raising the question of how context, and other model properties, influences action selection. Exploring the evaluation challenges presented by complex models will likely need to be done separately from (though informed by) subsequent DCIST CRA experimentation. That experimentation, for which this report serves as a partial baseline, will focus on demonstrating the value of DCIST CRA research to the Army, rather than on detailed examination of individual research projects.

7. References

Cladera F et al. Air-ground collaboration for language-specified missions in unknown environments. 2025. IEEE Trans Robot. http://doi.org/10.1109/TFR.2025.3584019

Hamzezadeh E, Rogers J, Dantam N, Petruska A. Exposure conscious path planning for equal exposure corridors. 2024 IEEE 20th International Conference on Automation Science and Engineering (CASE); 2024 Aug 28–Sep 1; Bari, Italy. c2024. p. 3660–3666. http://doi.org/10.1109/CASE59546.2024.10711539

Hughes N et al. Foundations of spatial perception for robotics: hierarchical representations and real-time systems. Int J Robot Res. 2024;431457–1505.

Kurtz M et al. Real-world deployment of a hierarchical uncertainty-aware collaborative multiagent planning system. arXiv; 2024 Apr 26. arXiv: 2404.17438. https://doi.org/10.48550/arXiv.2404.17438

Piekarski B, Sadler B, Kumar V, Ribeiro A. Distributed and collaborative intelligent systems and technology (DCIST) collaborative research alliance (CRA). DEVCOM Army Research Laboratory (US); 2022 Apr. Report No.: ARL-TR-9435.

Ravichandran Z, Murali V, Tzes M, Pappas G, Kumar V. SPINE: online semantic planning for missions with incomplete natural language specifications in unstructured environments. arXiv; 2024 Oct 3. arXiv: 2410.03035. https://doi.org/10.48550/arXiv.2410.03035

Strader J, Hughes N, Chen W, Speranzon A, Carlone L. Indoor and outdoor 3D scene graph generation via language-enabled spatial ontologies. IEEE Robot Autom Lett. 2024;9(6):4886–4893.

Strader J. Department of Aeronautics and Astronautics, Massachusetts Institute of Technology. Personal communication, 2025 Feb 20.

Appendix A. Test Cards

This appendix contains the tests planned out before the integration event began (Tables A-1 and A-2). Not all planned tests were executed. As noted in Table A-3, the final test in that table was skipped to give a demonstration to a visitor. In Table A-4, which shows the plan for exposure-conscious planning, only one agent was used, so all tests were related to environmental exposure optimization for a single robot. Sections 2–5 in the main body of the report above describe what happened at the event, while this appendix documents what was planned before the event.

Table A-1. Test card for planning with uncertainty.

No. of agents	Mode	Experiment goal	Outcome
2	Large language model (LLM)-based goal specification, auto-generated route graph, noncollaborative multiagent planner	Longer length scale planning 1 configuration (3 runs)	Trials done at Magazine Beach; heterogeneous agent capabilities led to more interesting configurations.
2	LLM-based goal specification, auto-generated route graph, collaborative multiagent planner	Longer length scale planning 1 configuration (3 runs)	
2	LLM-based goal specification, auto-generated route graph, noncollaborative multiagent planner, heterogeneous agent traversability capabilities	Longer length scale planning 1 configuration (1 run)	1 trial for each configuration was performed. Additional trials focused on 3-agent planning.
2	LLM-based goal specification, auto-generated route graph, collaborative multiagent planner, heterogeneous agent traversability capabilities	Longer length scale planning 1 configuration (1 run)	
3	Hand-drawn graph and auto-generated route graph/hand-drawn graph, noncollaborative multiagent planner	Planning with more agents 1 configuration (3 runs)	Hand-drawn graph trails were not run in favor of auto-generated graphs. Trials completed with quantitative results. Heterogeneous agent traversability capabilities used. These were the runs from which the 12% difference in time was calculated.
3	Hand-drawn graph and auto-generated route graph collaborative multiagent planner	Planning with more agents 1 configuration (3 runs)	
3	[Stretch goal] LLM-based goal specification, Auto-generated route graph, Noncollaborative multiagent planner, heterogeneous agent traversability capabilities	Planning with more agents 1 configuration (1 run)	LLM commander integrated and tested in collaborative trial (planner required minor operator intervention). Noncollaborative trial attempted but not completed due to planner issues.
3	[Stretch goal] LLM-based goal specification, auto-generated route graph, collaborative multiagent planner, heterogeneous agent traversability capabilities	Planning with more agents 1 configuration (1 run)	

Table A-2. Test card for language-based.

Completed	No. of agents	Objective	Notes
Yes	1 (robot)/ ≥2 (maps)	No. 1a (Husky): Demonstrate scene graph fusion and object-based relocalization in fused scene graphs	Verify quality of scene graphs and communication from base station to robots, uses teleop data, fused from Husky/Spot
Yes	1 (robot) ≥2 (maps)	No. 1b (Spot): Demonstrate scene graph fusion and object-based relocalization in fused scene graphs	Verify quality of scene graphs and communication from base station to robots, uses teleop data, fused from Husky/Spot
Yes	1 (Husky)	No. 2: Single-robot, autonomous planning/execution, easy language commands, easy goals	1 goal proposition; short, direct, unambiguous, prior map from teleop data (to decouple from phase 1), connects to natural language test card, completed in fused scene graphs
Yes	1	No. 3: Single-robot, autonomous planning/execution, medium language commands, medium goals	2–5 goal propositions; long, disambiguation, prior map from teleop data (to decouple from phase 1), connects to natural language test card
Yes	1	No. 4: Single-robot, autonomous planning/execution, hard language commands, hard goals	6+ goal propositions; long, universal quantification, disambiguation, prior map from teleop (to decouple from phase 1), connects to natural language test card
Yes	2–3	No. 5: Multi-robot, autonomous planning/execution for natural language commands	Prior map from teleop (to decouple from phase 1)
No	2–3 (stretch)	No. 6: Multi-robot, autonomous planning/execution for natural language commands, (capability aware role-assignment)	Prior map from teleop (to decouple from phase 1), connects to natural language test card

Table A-3. Test card for incomplete language specifications.

Requirements	Test	Progress
Internet, 1 Jackal, traversability estimation and object vision	Infrastructure test (LLM) via simple explicit tasking	Complete
Internet, 1 Jackal, traversability estimation and object vision	Semantic test (vision, control) via explicit tasking	Complete
Internet, 1 Jackal, traversability estimation and object vision	Test mission execution using 1 unmanned ground vehicle (UGV) with a full preloaded map (semantic route inspection)	Complete (largest map 1 km)
Internet, 1 Jackal, traversability estimation and object vision	Test mission execution using 1 UGV with a partial preloaded map (semantic route inspection + surveillance)	Complete

Internet, 1 Jackal, traversability estimation and object vision	Waypoint tasking of unmanned aerial vehicle via Opportunistic communications network	Skipped to hold a demo

Table A-4. Test card for exposure-conscious path planning.

No. of agents	Mode	Experiment goal	Notes
1–3	High-level objectives testing, incorporating tactical movement and stealth objectives into controller.	Environmental exposure optimization.	Testing at tennis courts.
1–3	Multiagent high-level objectives testing with tactical movement/stealth considerations in the controller.	Environmental exposure optimization with team collision constraints.	Testing at tennis courts.
1–3	Multiagent high-level objectives testing with tactical movement/stealth and no inter-agent communication.	Experimenting with no-communication movement.	Testing at tennis courts.
1–3	To be determined (TBD): follow-up experiments based on previous results and interests.	. . .	. . .
1–3	TBD: follow-up experiments based on previous results and interests.	. . .	. . .
1–3	TBD: follow-up experiments based on previous results and interests.	. . .	. . .

Appendix B. Ranges

This appendix contains locations, images, and brief descriptions of ranges near Camp Buckner, New York, that would be good for experimentation and were not described previously. Each range is referred to by the number West Point Range Control uses to label the range, such as Range 15, 16, etc.

B-1. Range 15

Range 15 offers two sites indicated by white arrows in Figure B-1. We will refer to the center site as "the mansion" and the left-most site in the image as "station 2."

Figure B-1. Overhead image of Ranges 15 and 16.

Figure B-2. The mansion at Range 15.

The Range 15 site referred to as the mansion (41.32455, –74.01768) offers an underground tunnel with two entrances marked by green squares in Figure B-2. It also has a two-story cinderblock building with doors and window covers.

The area referred to as station 2 (41.32768, –74.01645) consists of a row of red metal CONEX boxes (Figure B-3) and a concrete building with many rooms, all uncovered (no roof) with a door and window facing the center (Figure B-4).

Figure B-3. Station 2 CONEX boxes and cleared area.

Figure B-4. Station 2 concrete rooms for room clearing.

B-2. Range 16

Range 16 (41.321829, –74.017607), shown from above in Figure B-5, has five red cinderblock and wood buildings with doors and uncovered windows. One building is two stories and the rest is a single story (Figure B-6). A wheeled robot could go around any of the buildings, and a Husky or Spot could go in if the door was propped open. A road runs through the village, connecting Ranges 15 and 16.

Figure B-5. Overhead image of Range 16.

Figure B-6. Example of buildings at Range 16.

B-3. Range 11

Range 11 (41.36013, –74.03338) is a small-arms range with berms of varying sizes established at fixed distances from a firing line. The orange triangle flag in Figure B-7 shows the left limit of the firing positions. Figure B-8 shows paths traversing the range that allow wheeled vehicles to travel throughout the site.

Figure B-7. Downrange perspective on Range 11.

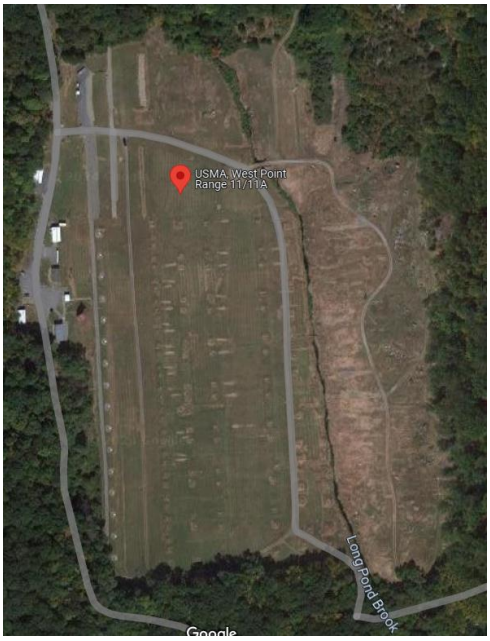

Figure B-8. Overhead image of Range 11.

B-4. Range T2

The Range T2 site (41.33719, –74.04362) offers a site we refer to as the CONEX site. It has two CONEX structures shown in Figure B-9, one with a CONEX box set side by side (left side of Figure B-10) and the other with a CONEX box stacked (right side of Figure B-10), with the two structures separated by approximately 20 m. The range can be entered by a gravel road, but thick brush and drop offs on the sides of the road make the land adjacent to it not trafficable by wheeled vehicles. A gravel path leads from the CONEX site to a bivouac site that has patches of clear area and less dense trees, shown in Figure B-11. In Figure B-12, the CONEX site is marked by a green square, and the bivouac site by a yellow circle. Wheeled robots could start a mission at the bivouac site and then proceed to the CONEX site.

Figure B-9. Military operations on urbanized terrain (MOUT) site.

Figure B-10. The CONEX structures at the MOUT site.

Figure B-11. Gravel path from bivouac to MOUT site.

Figure B-12. Overhead image of Range T2.

List of Symbols, Abbreviations, and Acronyms

3D	three-dimensional
API	application programming interface
ARL	Army Research Laboratory
CRA	Collaborative Research Alliance
DCIST	Distributed and Collaborative Intelligent Systems and Technology
GPS	global positioning system
ID	identifier
JSON	JavaScript Object Notation
LiDAR	light detection and ranging
LLM	large language model
LOS	line of sight
MOUT	military operations on urbanized terrain
PDDL	planning domain definition language
SD	standard deviation
TBD	to be determined
UGV	unmanned ground vehicle

1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
1 (PDF)	DEVCOM ARL FCDD RLB CI TECH LIB