Inspecta 1.0: Incorporating Spot Robotic Support

Zahi Kakish, Kyle Williams, Shiloh Elliot, Sagan Cox, Heidi Smartt
Sandia National Laboratories, Albuquerque, NM, United States

ABSTRACT

Sandia National Laboratories has designed and is developing *Inspecta* (**I**nternational **N**uclear **S**afeguards **P**ersonal **E**xamination and **C**ontainment **T**racking **A**ssistant), an Artificial Intelligence (AI)-enabled smart digital assistant (SDA) with robotic support. The goal is to provide inspectors an in-field digital auxiliary support capable of performing limited physical functions through an embodied intelligence on tasks identified as tedious, challenging, or prone to human error. Sandia has selected a high-impact inspection task, seal examination, in which an early Inspecta prototype residing on a smartphone could aid an inspector alongside a Boston Dynamics' "Spot" robot. Spot, as part of the Inspecta platform and with human inspector instruction, will initially locate a seal, use optical character recognition to image the seal number and communicate the number back to Inspecta, who will track the overall seal examination task. As the project progresses, Spot will perform more complicated activities associated with seal examination. This paper describes Spot's autonomy capabilities, progress on Spot integration with Inspecta, and how future functionality will be further developed under the Inspecta project.

INTRODUCTION

The International Atomic Energy Agency (IAEA) Department of Safeguards performs inspections at various nuclear facilities around the world [1] in accordance with safeguards agreements under the Treaty on the Non-Proliferation of Nuclear Weapons (NPT). The NPT seeks to provide assurance to the global community that states are using nuclear technologies for peaceful purposes. Inspections may differ based on agreements between the IAEA and states, type of facility, and type of inspection; however, a common theme is that inspection tasks may occur in hazardous environments, require inspectors to wear Personal Protective Equipment, be physically and mentally challenging, tedious, and performed with limited time. These tasks may benefit from human-machine teaming systems that can reduce their repetitive workload and cognitive burden, allowing inspectors to focus on other activities and maximize their effectiveness and efficiency. During a series of interviews with former IAEA inspectors, five common tedious, challenging, and prone to error tasks were identified, including seal examination. For this task, inspectors locate sealed items such as containers of spent nuclear fuel, checking the identification (ID) number located on each seal and compare it against a paper list, pull on the seal to ensure proper attachment, visually examine the seal body and item for signs of tampering, and move to the next sealed item.

We are developing Inspecta [2], an AI-enabled smart digital assistant (SDA) prototype that can assist in a variety of tasks, beginning with seal examination, speech synthesis and speech recognition (for

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. SAND2023-02701C

auditory note taking). Available on a smartphone, Inspecta uses optical character recognition (OCR) to extract a seal's ID using the smartphone's camera, records it, and checks it against known seal IDs ingested into the Inspecta app prior to an inspection. This reduces the amount of time it takes for an inspector to search a paper list for a confirmed seal ID and may reduce errors associated with manual list comparison.

Using robotics as a physical implementation of Inspecta will further help inspectors by having an embodied intelligence perform the task *in tandem* with the inspector coordinated through the Inspecta app. To achieve this, we have acquired a Spot robot [3] developed by Boston Dynamics, and have begun incorporating autonomous behavior and integrating them into the Inspecta app. In short, the Inspecta app will contain the list of seal ID numbers to be examined, Spot will be instructed by the inspector to begin seal examination (under facility escort), Spot will use object detection to locate monitored items (in our case containers), approach the item, locate a seal, grasp the seal, perform OCR, send the seal ID as read by Spot using the OCR back to Inspecta where the seal ID will be marked as confirmed.

In this paper, we give a brief overview of robot autonomy at a high-level and how this pertains to Spot for Inspecta integration. We establish what is needed for an inspection task to be deemed a success and what is required of an autonomous system that works concurrently with human operators to make this work. Next, we discuss how we have approached incorporating that information for our solution. In addition, we discuss object detection and its relation to properly identifying containers within undefined environments such as a nuclear facility. Finally, we summarize our current progress of our system while discussing potential future capabilities and research using Spot.

SPOT ROBOT OVERVIEW

Spot [3] is a quadruped robot developed by Boston Dynamics, a Boston-based robotics company, capable of various functionality suitable for inspection [4], logistics [5], and search and rescue [6]. Each leg of the quadruped is comprised of three actuators that drive hip abduction/adduction, hip flexion/extension, and knee flexion/extension [7]. Together, all four legs drive Spot's motion that uncannily resembles the gait of biological quadruped animals. Spot contains a perception sensor-suite used for path planning, obstacle avoidance, and localization. On each side of Spot's base are a stereo camera and depth RGB camera, except for the front of the robot, which contains two of each. These sensors point in opposite directions and their information is stitched together to deliver a larger frontal field of view.

In addition, Spot is equipped with a front-mounted 6-DoF (degree of freedom) articulated robotic arm with a single-joint gripper end-effector. Its motion around the base is compensated by Spot when carrying an object held by the gripper. Within the end-effector is a 4K RGB camera that can zoom up to 30x for tasks that require more precise image capturing than those taken from the lower-resolution camera sensor suite on Spot's base. While users cannot directly manipulate Spot's individual actuators, Boston Dynamics provides a robust Software Development Kit (SDK) [8] written in Python that abstracts motion, manipulation, and sensing tasks for incorporation into user's preferred client-side applications. Unfortunately, any user-created applications that drive Spot must run on an external computing device and not Spot's internal computer which is restricted by the manufacturer. Figure 1 shows a 3D printed payload mount for an Nvidia Xavier [9] single board computer (SBC) we have added to Spot to run our algorithms and circumvent this restriction. The SBC connects via ethernet to a port located in Spot's rear. Wi-Fi antennas allow an inspector to interface with Spot using point-to-point Wi-Fi through the Inspecta app.

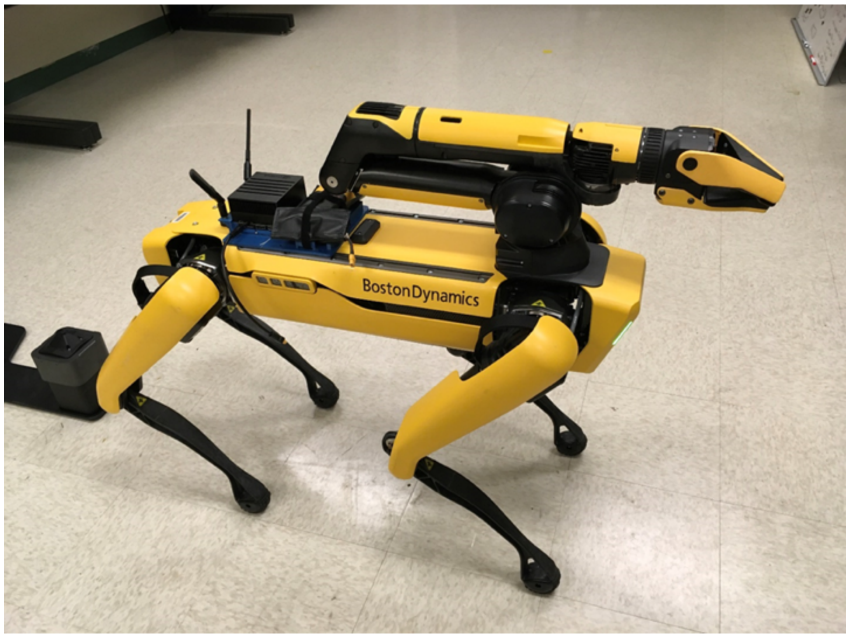

Figure 1: Spot carrying a custom-built Nvidia Xavier mount.

ROBOT AUTONOMY

Using the Spot SDK, we can develop programs to move Spot and engage with the surrounding environment on a per task basis; however, this is not a long-term solution for enabling more complex autonomy that we wish to run alongside a human operator. Instead, we look towards general theories for constructing robot autonomy. Such principles of robot autonomy model the behavior of an agent within an undefined environment as a cyclical sequence of three procedures [10] that a robot undergoes to accomplish a goal: the robot must *see* its surrounding through sensors, after sensing its area the robot must *think* about where it is and what to do next, and finally *act* to move or interact with the world. Within a robot software stack, the *see*-*think*-*act* paradigm translates to programs that fuse sensor information, localize the robot for planning the next action, and actuation for the next step. The robot should arrive to its goal upon repetition of these three procedures all mediated by a *planner*.

To create an autonomy algorithm with Spot, we would need to follow this same procedure with a few caveats given the platform's capabilities. First, mapping and localization is primarily done on Spot's internal computer; therefore, we can only access the information post-processed when needed. Second, object avoidance is a built-in option for trajectory commands. This allows mission planners to function in a global view rather than a local one thereby making waypoint recording and waypoint-to-waypoint navigation easier. Finally, trajectories are built using the SDK and rely on the robot's frames of reference provided by Spot's internal computer rather than ones given by our planner. With these caveats, we devised a mission planner capable of global localization and future trajectory command creation using hybrid frames of reference.

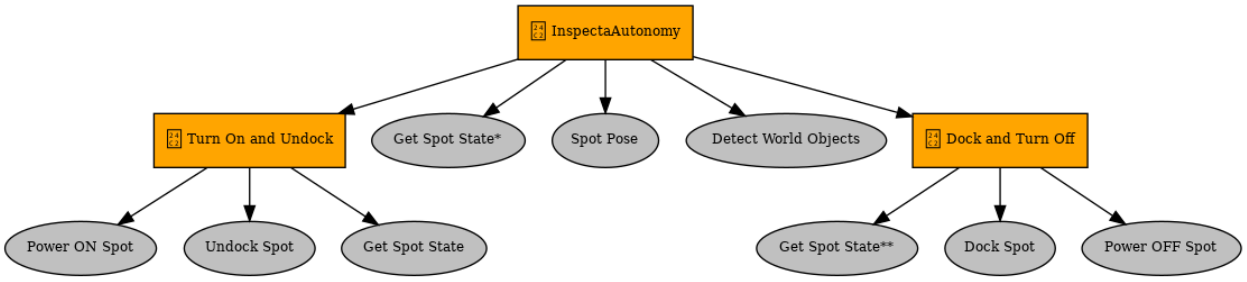

Figure 2: A simple behavior tree composed of multiple actions strung together to form an autonomous system.

While using a *see-think-act* paradigm with a planner may seem to produce sufficient, simple autonomy for Spot, this approach is limited for complex and long-term autonomy that necessitate disparate intermediary goals. For instance, the necessary sensing for a target may require a single or range of different sensing modalities for planning. Sensing obstacles during a trajectory is different than sensing the location of a container. One requires LiDAR while the other requires machine learning (ML) object detection. These different goals require different autonomy that can switch context given a location, state, or goal. Thus, we need a way to abstract this paradigm to a higher-level space where we can link multiple individual autonomies that do a see-think-act step for multiple goals. For this, we can rely on mathematical models that, given certain conditions, switch between a finite set of tasks. The most popular of these models are hierarchical finite state machines [11] and behavior trees [12].

While both are similar with regards to building behaviors, they differ in what guides the main building block of those behaviors. The former relies on what the robot's *state* is, while the latter is the status of a specific task. This reliance on robot state allows for more reactive autonomy since these can be added as conditions following a task's completion. Behavior trees differ by allowing for easier modification of the overall autonomy since adding new or different actions do not change the overall task. For our application, a measure of Spot's success is how well it performs specific inspection-related *tasks* with a human inspector. Therefore, we chose to use behavior trees to supply the autonomy for Inspecta because an autonomy that is more amenable to modification is thereby more beneficial for human-machine teaming.

Behavior trees are composed much like a tree structure shown in the example in Figure 2. The overall "InspectaAutonomy" behavior tree contains sequences of action primitives or state checks from left to right. Certain behaviors are categorized as *sub-trees*, i.e., a smaller or task specific modular subset of the overall autonomy, that help build our whole behavior tree. These are generally a grouping of multiple actions that facilitate a certain behavior. In Figure 2, we see two examples of this. The first is a "Turn On and Undock," which is a sequence of three actions that comprise that sub-tree behavior. The second is the "Dock and Turn Off," which is an opposite version of the previous behavior. These are simple sub-trees, but others can be more complex depending on the behavior. For example, an "Exploration" tree may have a "Search" behavior sub-tree which would potentially be much deeper as certain actions might require other sequence of actions and checks to work robustly [13]. These features of behavior trees mean actions, checks, or sub-trees are easily interchangeable and modular to fit the overall task's requirements. Currently, we are developing numerous actions, checks, and behavior trees using Spot's API to generate our autonomy as a behavior tree.

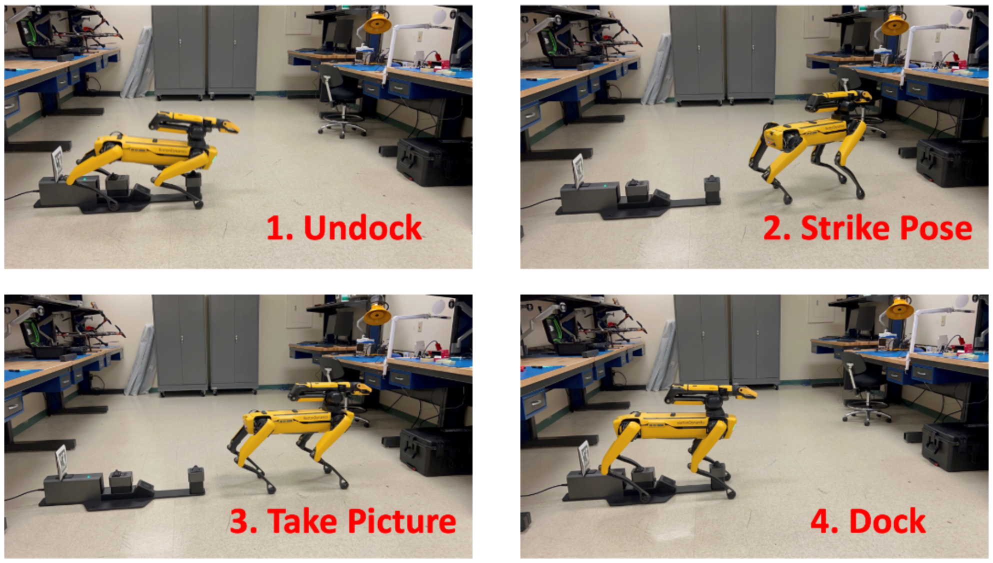

Figure 3: Spot executing the sample behavior tree shown in Figure 2.

In Figure 3, we see Spot going through each sub-tree and action we've developed from left to right using the sample behavior tree algorithm from Figure 2. Spot goes through the undocking sub-tree, strikes a pose, executes the "Detect World Objects / Take Picture" actions, and reverses onto the dock for the docking sub-tree as the "InspectaAutonomy" tree specifies. Certain actions such as "Get Spot State" and "Detect World Objects / Take Picture" actions have no physical indication that they are occurring; however, we do label the latter for clarity since it is only done once.

INSPECTION TASK AUTONOMY

Using the system we discussed in the previous section, we can begin to formulate how Spot may need to operate for a seal examination task. Spot would need to do the following at the inspector's behest:

explore an unknown area (under facility escort), locate a container, move towards the container, locate the seal on the container, extend the arm to grab seal, read the number on the seal, send the information back to Inspecta for data entry, and pull on the seal to test proper attachment. Completing the whole scenario without failure constitutes a successful run of the behavior tree, which it may repeat until an inspector advises Spot to stop. In addition, how well the autonomy would work with a human inspector is a major consideration since Spot and Inspecta should *complement* the inspector, not replace them.

To achieve this, we have begun creating behavior sub-trees of each step in the inspection procedure described earlier. Each of these are complex and require both robust planning for Spot to be able to locate containers in the environment while being safe around equipment, inspectors, and other workers. However, combining all these behaviors into one tree would result in a messy and difficult debugging experience if errors were to arise. We mitigate this through breaking the overall behavior into three separate trees: one for exploration, one for moving into position after identifying a container, and one to use Spot's arm for seal inspection. After the completion of one tree, we go through various state checks, such as battery life or queued input from an inspector via the Inspecta app, before proceeding onto the next one. This provides additional robustness to the system as well as the ability to extend our autonomous system with future behaviors. Furthermore, it allows the inspector to modify Spot's next behavior between the smaller trees since modifying a larger tree could lead to potential failures.

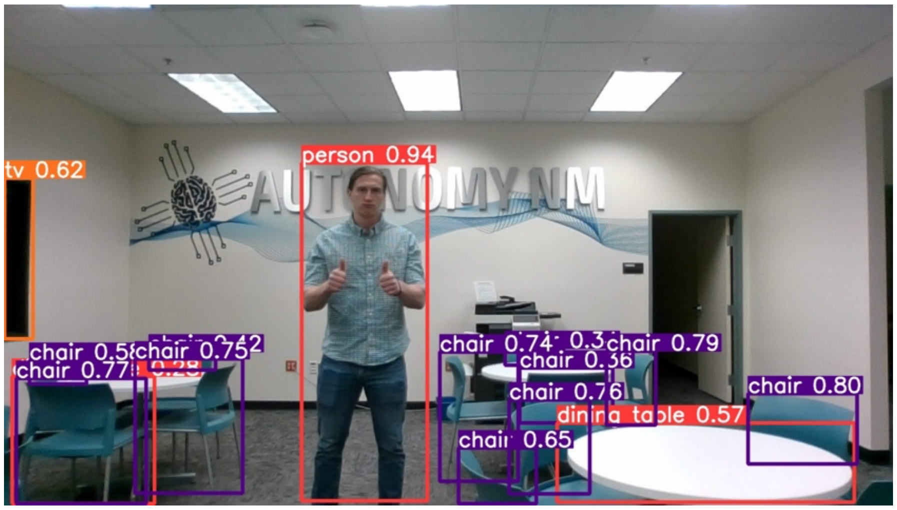

Figure 4: YOLO_v5 object detection running on the Xavier GPU onboard Spot.

Adding to the complexity of an autonomous system, containers and seals are not labeled or easily identifiable within the environment. Therefore, these behaviors will require Spot to classify the equipment it interacts with using vision-based methods. Object detection plays an important part of our system since Spot will not only need to recognize a container in a cluttered environment but estimate its position in space to generate trajectories towards the identified object. Using advanced object detection algorithms like YOLO (You Only Look Once) [14] shown in Figure 4, we can recognize many objects within an image frame for our planner and approximate its location with respect to Spot. However, the default algorithm does not have the ability to classify containers. We thus enhance the algorithm by fine-tuning it with a large dataset of container images provided by the Limbo project [15], a synthetic computer vision training set developed for nuclear safeguards. Figure 5 gives a preliminary look at the classification accuracy of our modified YOLO version 5 for containers. Once fully trained, we intend to test the model alongside Spot's planner to decide which container to select in cases of multiple viable targets and calculate trajectories towards it without colliding with nearby obstacles.

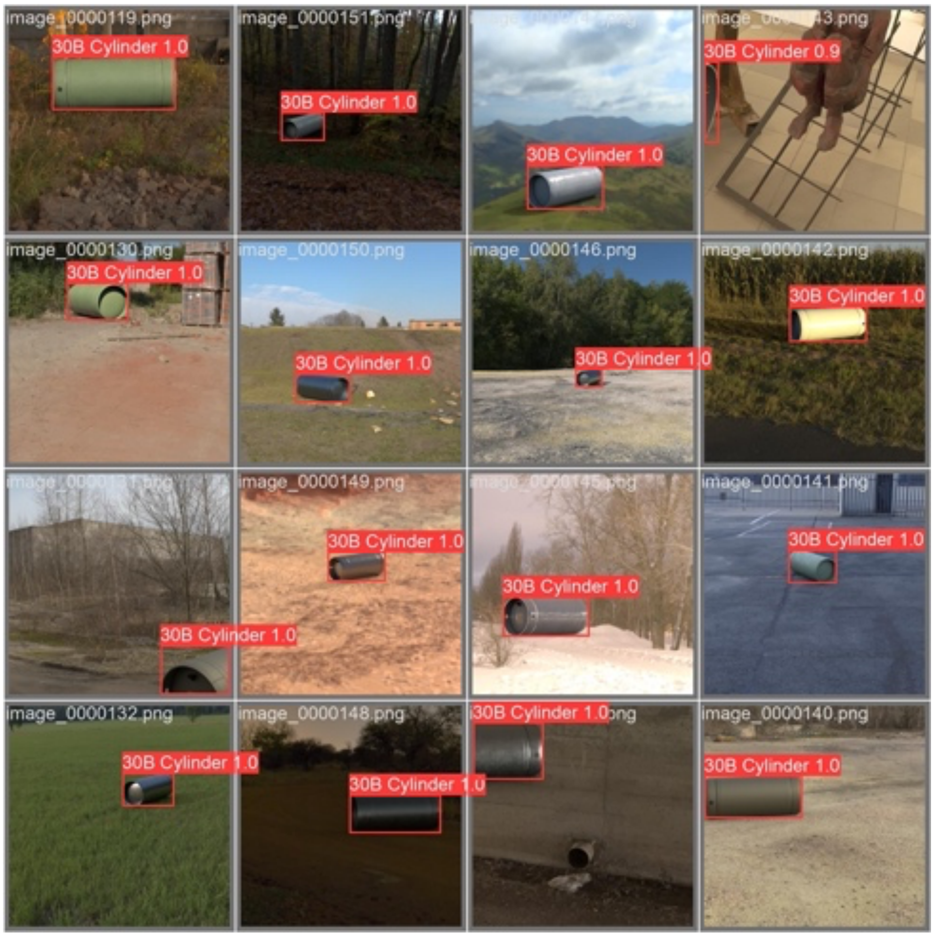

Figure 5: Preliminary training accuracy of YOLO_v5 for containers.

SUMMARY AND FUTURE CAPABILITIES

Human-machine teaming through the Inspecta app has the potential of easing the burden of repetitive or prone to error tasks for inspectors. When adding robotics, this should reduce the workload and cognitive burden as well as reduce an inspector's exposure to environmental hazards. In this paper, we have provided a brief overview of our design and current progress in bringing safeguards related autonomy to Spot that may assist inspectors in addition to the Inspecta app. Given the complexity of inspection-related tasks, we have selected using behavior trees to provide modular and robust autonomy to the system while allowing effortless input from the inspector between seal examination sub-tasks. In addition, we have begun training advanced object detection models on synthetic container datasets for incorporation into Spot. Future capabilities include generating arm trajectories to move towards identified seals on containers and communication between Spot and Inspecta for imaging of the seals and task tracking.

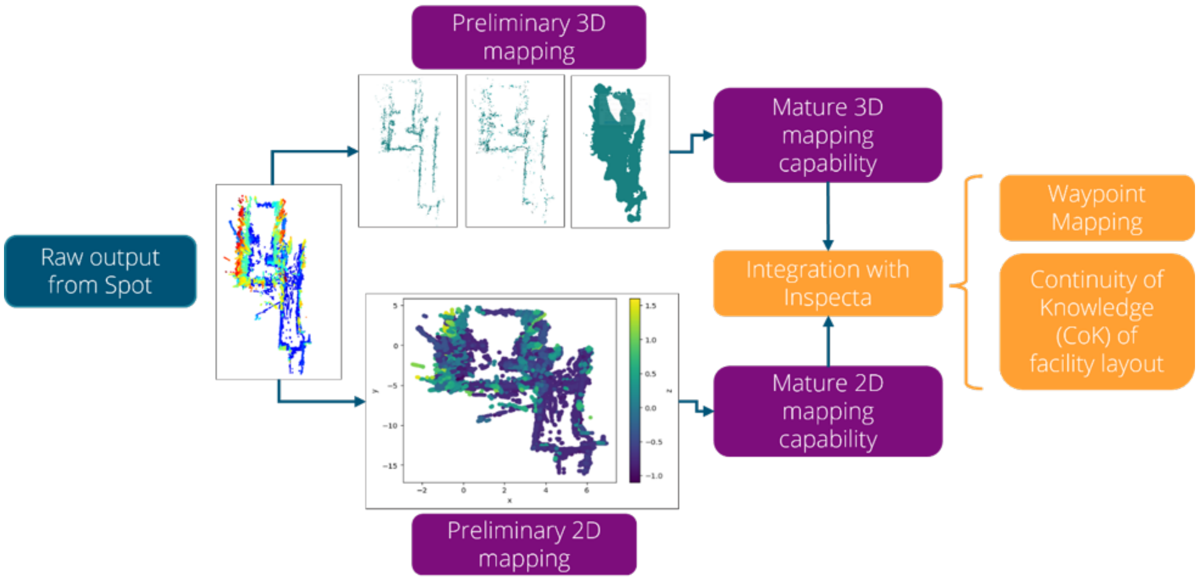
Preliminary 3D mapping
Raw output from Spot
Mature 3D mapping capability
Integration with Inspecta
Mature 2D mapping capability
Waypoint Mapping
Continuity of Knowledge (CoK) of facility layout
Preliminary 2D mapping

Figure 6: High-level depiction of potential avenues for the creation and integration of 3D and 2D mapping capabilities into Inspecta.

Other future research directions we intend to pursue include advanced mapping techniques using Spot's on-board sensors and additional sensor payloads. Specifically, LiDAR sensors can generate 3D point clouds of facilities that Spot navigates through, as shown in the example in Figure 6. These raw point clouds are amenable to 2D and 3D map reconstructions, which can serve as future feature integration into Inspecta such as waypoint mapping of reoccurring inspection operations at a specific facility. These mappings could also prove useful for detecting facility changes (design information verification) or other anomalous activity.

In addition, we plan to rearchitect our behavior tree algorithm to become an open-source project for other organizations to use. This will require removing actions, conditions, and behavior sub-trees related to safeguards while keeping generic and inspection versions for others to build upon. Thus, the Inspecta related abilities and connectivity will become a superset that would run on top of the open-source version providing transparency to the project for future collaborators while maintaining security against potential adversaries' ability to compromise Spot's inspections.

ACKNOWLEDGEMENTS

The authors would like to acknowledge and thank the U.S. National Nuclear Security Administration (NNSA) Office of Defense Nuclear Nonproliferation R&D Safeguards portfolio for funding and supporting this research.

REFERENCES

[1] International Atomic Energy Agency. https://www.iaea.org/topics/basics-of-iaea-safeguards. Accessed 3/17/2021

[2] Smartt, H., Dorawa, S., Hannasch, D., Honnold, P., Shoman, N., Solodov, A., & Spence, K. (2022) Inspecta 1.0: Development of an Architectural Roadmap, In *Proceedings of the INMM Virtual Annual Meeting*.

[3] Spot – The Agile Mobile Robot. Boston Dynamics. (n.d.) Retrieved April 5, 2023, from https://www.bostondynamics.com/products/spot

[4] Huamanchahua, D., Yallia-Villa, D., Bello-Merlo, A., & Macuri-Vasquez, J. (2021, December). Ground Robots for Inspection and Monitoring: A State-of-the-Art Review. In *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)* (pp. 0768-0774). IEEE.

[5] Echelmeyer, W., Kirchheim, A., & Wellbrock, E. (2008, September). Robotics-logistics: Challenges for automation of logistic processes. In *2008 IEEE International Conference on Automation and Logistics* (pp. 2099-2103). IEEE.

[6] Niroui, F., Zhang, K., Kashino, Z., & Nejat, G. (2019). Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments. *IEEE Robotics and Automation Letters*, *4*(2), 610-617.

[7] Yan, W., Pan, Y., Che, J., Yu, J., & Han, Z. (2021) Whole-body kinematics and dynamic modeling for quadruped robot under different gaits and mechanism topologies. In *PeerJ Comput Sci.*

[8] Boston Dynamics (2023). Spot SDK – Spot Documentation. Retrieved April 5 ,2023, from https://dev.bostondynamics.com/

[9] Deploy AI-Powered autonomous machines at scale. *Nvidia*. Retrieved April 5, 2023, from https://www.nvidia.com/en-us/autonomous -machines/embedded-systems/jetson-agx-xavier/

[10] Bohg, J. (2022) Lecture 1: Principles of Robot Autonomy I. In *Principles of Robot Autonomy I Stanford Course*. https://stanfordasl.github.io//aa274a/.

[11] Conner, D. & Willis J. (2017) Flexible Navigation: Finite state machine-based integrated navigation and control for ROS enabled robots. In *SoutheastCon 2017* (pp. 1-8).

[12] Colledanchise, M. & Ogren, P. (2018) Behavior Trees in Robotics and AI. arXiv preprint arXiv:1709.00084.

[13] Dang, T., Tranzatto, M., Khattak, S., Mascarich, F., Alexis, K., & Hutter, M. (2020). Graph-based subterranean exploration path planning using aerial and legged robots. In *Journal of Field Robotics* 37 (8): 1363-1388

[14] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015) You Only Look Once: Unified, Real-Time Object Detection. arXiv preprint arXiv:1506.02640.

[15] Gastelum, Z.N., Shead, T.M., and Rushdi, A., (2021, September) A Large Safeguards-Informed Hybrid Imagery Dataset for Computer Vision Research & Development. In *Proceedings of the Institute of Nuclear Materials Management and European Safeguards Research and Development Association Joint Annual Meeting, September 2021*. Available at: https://esarda.jrc.ec.europa.eu/esarda-43rd-joint-annual-meeting_en