SAND2024-13646

SANDIA REPORT
Tracking #: 1758657/SAND2024-XXXX
Printed October 2024

Sandia National Laboratories

Inspecta Technical Report

Heidi A. Smartt, Shiloh Elliott, Philip Honnold, Zahi Kakish, Adithya Ramakrishnan, Tania Rivas, Nathan Shoman, Kyle Williams

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico
87185 and Livermore,
California 94550

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology & Engineering Solutions of Sandia, LLC.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: http://www.osti.gov/scitech

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Rd
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: https://classic.ntis.gov/help/order-methods/

ABSTRACT

Sandia National Laboratories (SNL) is in the process of creating Inspecta (International Nuclear Safeguards Personal Examination and Containment Tracking Assistant), an Artificial Intelligence (AI)-powered smart digital assistant (SDA) with robotic capabilities, aimed at enhancing the effectiveness, efficiency, and safety of international nuclear safeguards inspections. This innovative tool is designed to assist inspectors on-site by supporting or automating tasks that are typically mundane, hazardous, or susceptible to errors. In 2021, the development team established the specifications for Inspecta by analyzing International Atomic Energy Agency (IAEA) documents and consulting with former IAEA inspectors and subject matter experts. This process involved aligning in-field inspection tasks with existing commercial or open-source technologies to outline a roadmap for the initial prototype of Inspecta, while also identifying areas needing further research and development. From 2022 – 2024, the focus has shifted to integrating a critical inspection activity, the examination of seals, into an early version of Inspecta. This has involved developing both the software and hardware capabilities necessary for this task. This report outlines the ongoing advancements in Inspecta's functionalities, specifically those supporting the seal examination process.

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.

Sandia National Laboratories is a multi-mission 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.

CONTENTS

LIST OF FIGURES

ACRONYMS AND TERMS

Acronym/Term	Definition
AI	Artificial intelligence
AP	Additional Protocol
API	Application program interface
CRAFT	Character-region awareness for text detection
DOF	Degrees of freedom
FVPS	Field verifiable passive loop seal
GPU	Graphics processing unit
IAEA	International Atomic Energy Agency
ICP	Iterative Closest Point
INSPECTA	International Nuclear Safeguards Personal Examination and Containment Tracking Assistant
LiDAR	Light detection and ranging
ML	Machine learning
NERFs	Neural Radiance Fields
OCR	Optical character recognition
ONNX	Open neural network exchange
OSTI	Office of Scientific and Technical Information
PIV	Physical inventory verification
PNNL	Pacific Northwest National Laboratory
PPE	Personal protective equipment
R&D	Research and development
RAG	Retrieval Augmented Generation
RGB	Red green blue
SAM	Segment anything model
SDA	Smart digital assistant
SDK	Software development kit
SLAM	Simultaneous Localization and Mapping
SNL	Sandia National Laboratories
STR	Scene text recognition
TCP	Transmission control protocol
UI	User interface
VAD	Voice Activation Detection
YOLO	You only look once

1. BACKGROUND

The IAEA Department of Safeguards is responsible for verifying international nuclear safeguards agreements. The mission of international safeguards is "to deter the spread of nuclear weapons by the early detection of the misuse of nuclear material or technology. This provides credible assurances that States are honouring their legal obligations that nuclear material is being used only for peaceful purposes" [1]. The implementation of international safeguards is unique for different states, as they are based on sovereign agreements between a State and the IAEA, as well as from facility-to-facility as determined through a safeguards agreement's facility attachment. Safeguards activities at a nuclear facility are also based on state factors and the IAEA's technical objectives as defined in the Annual Implementation Plan.

Despite the variability in their application, inspectors consistently perform a range of common tasks, including auditing facility records, inspecting and maintaining safeguards equipment, taking measurements and samples, examining and verifying seals, counting items, reviewing surveillance footage, confirming design information, and monitoring for any discrepancies. These tasks can be both mentally and physically demanding, making them prone to human error; further, some tasks involve activities in hazardous environments. Moreover, the workload of international safeguards inspectors is on the rise due to several factors: the increasing number and variety of nuclear facilities under safeguards, the growing global inventory of special nuclear materials due to the longevity of safeguards for waste products and spent fuel, and a shift in the inspectors' role from auditing to more investigative duties. Despite these growing responsibilities, inspectors have limited time at facilities and must perform their duties as efficiently, effectively, and safely as possible. Efforts to enforce obligations associated with the Additional Protocol (AP) further compounds their workload.

AI and Machine Learning (ML) are becoming increasingly integrated into daily life, as seen in technologies like automated driver-assistance systems, personalized online shopping recommendations, voice-controlled smart home devices, and AI-powered vacuum cleaners, as well as SDAs like Amazon's Alexa[1]. Applying these advanced technologies to the field of international nuclear safeguards could significantly enhance the efficiency, effectiveness, and safety of inspection activities, particularly those that are repetitive, hazardous, and error prone.

To address the evolving needs of safeguards inspectors, we are developing a prototype AI-enabled SDA with robotic support named Inspecta. Inspecta is designed to assist with general tasks, such as note taking, and more specialized tasks, such as reading seal numbers through optical character recognition (OCR). It will be housed in a compact, portable, and wearable device (with the current version residing on a Google Pixel 6 smart phone), primarily interacting with inspectors through voice commands and, when necessary, displaying information visually. We have assumed that there will be no communication to the Internet or cellular network from within the facility, and thus all software and algorithms run on-device. It is important to note that Inspecta is intended to complement human inspectors, not replace them.

Inspecta is a mission-centric platform to demonstrate what's possible within the current state-of-the-art that's designed to serve as a base for real-world solutions. Large parts of Inspecta are based on ML, which is rapidly evolving, leading to an iterative design process. Inspecta skills see revision as improved approaches within the ML literature are developed. Significantly mature technologies are

[1] https://developer.amazon.com/en-US/alexa

often transitioned to other sponsor spaces for higher TRL development. For example, OCR technologies for seal examination pioneered under Inspecta have been transitioned to a standalone project to deploy these technologies for the IAEA's unified seal reader and the IAEA's Equipment Radiation Monitoring Laboratory.

In this report, we will share details on Inspecta technical capabilities ("skills") that primarily assist inspectors in a seal examination task, selected to demonstrate progress towards a relevant high-impact task. Skills required for the seal examination task include speech recognition, speech synthesis, wake word, OCR (of both seal/container numbers and documents), information recall, robotic autonomy, object detection, segmentation, pose estimation, and mapping. We note that the seal examination task was selected early in the project after a literature review of IAEA inspection tasks and IAEA publications identifying challenges, and from interviews with former IAEA inspectors and subject matter experts.

1.1. Prior Relevant Work

In 2021, we started with precisely defining what Inspecta would be, developing a list of high-impact tasks performed by IAEA inspectors, and creating a list of technical skills an SDA would need to support an inspector in their tasks. In other words, we began with the mission and sought to develop skills around the constraints and tasks of the mission. We drew on three sources of information as part of this process:

1. Safeguards task analysis. We began by collecting a list of tasks that inspectors complete in the field, based on IAEA safeguards reference documents. This list was extracted from the IAEA Safeguards Manual under prior SNL work, and we adopted it to frame the types of activities that inspectors currently perform, so we can identify areas where Inspecta may provide support. The task list is at a relatively general level (e.g., "perform maintenance on safeguards surveillance equipment," rather than detailing each task step).
2. Review of IAEA publications of safeguards challenges. The team reviewed several IAEA safeguards publications to identify challenges that inspectors currently face or expect to face in the future. As the team reviewed the documents, we made notes according to the task analysis described in step 1 regarding where publications were identifying challenges, or opportunities for implementing AI, robotics, automation, etc. The documents reviewed for this step include:
    a. IAEA Safeguards, "Emerging Technologies Workshop: Trends and Implications for Safeguards Workshop Report." February 2017 [2]
    b. IAEA Safeguards, "Emerging Technologies Workshop: Insights and Actionable Ideas for Key Safeguards Challenges Workshop Report." STR-397, January 2020 [3]
    c. IAEA Safeguards, "Research and Development Plan: Enhancing Capabilities for Nuclear Verification." STR-385, January 2018 [4]
    d. IAEA Safeguards, "Development and Implementation Support Programme for Nuclear Verification 2020-2021." STR-393, January 2020 [5]
3. Former inspector challenges elicitation. Finally, we identified former IAEA safeguards inspectors, and individuals with related highly relevant experience in facility operations and nuclear materials control. We interviewed eight experts, and documented their anonymized input regarding:

a. The most difficult or most tedious tasks performed as an inspector.
b. The inspection tasks most subject to human errors.
c. The inspection tasks/activities that other inspectors might most trust to automated system.
d. Perceived challenges of facility operators to meet their international safeguards obligations.

Once we completed data collection from these sources, we documented the potential capabilities for Inspecta and identified the technical capabilities required for each of those. This was done (to some degree) for all inspection tasks, but only a subset of these tasks was further examined and mapped into Inspecta skills and technical capabilities.

We extracted tasks that were identified the greatest number of times during these interviews as (1) surveillance review, (2) Physical Inventory Verification (PIV, in general), (3) transcription, (4) information integration, (5) seals examination and verification, and (6) spent fuel verification.

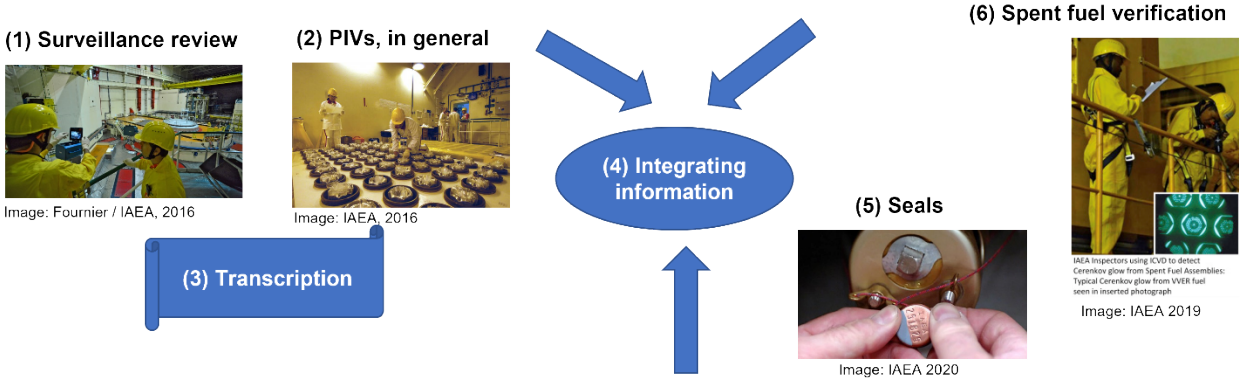


Figure 1: Inspection tasks identified through interviews as most challenging, tedious, or prone to human error.

Transcription and integrating information are often part of the other identified tasks, and PIV is a general term that includes surveillance review, seals, and spent fuel verification – therefore we focused on surveillance review, seals and spent fuel verification as potential Inspecta tasks. Other research and development (R&D) programs utilize ML for surveillance review and are investigating using a floating robotic platform for spent fuel verification [6]. Further, setting up a surrogate seal examination activity to collect training data and perform testing is easier than a surrogate spent fuel pool. Finally, we considered the technology capabilities that would be needed for Inspecta to assist with a task and ensured that the technology needed is relatively mature (TRL 4 and higher). Therefore, the task chosen for the initial Inspecta prototype was examining[2] metal cup seals (hereafter called "seal examination"). The skills developed for this task can be re-applied for other tasks in future iterations of Inspecta.

1.2. Task Detail and Skills Needed

The seal examination task is important but tedious for IAEA inspectors. In one common use case, metal cup seals (Figure 2) with a numeric identifier are attached to labeled containers (Figure 3, Figure

[2] The metal cup seal cannot be fully verified in the field and is removed and taken to IAEA headquarters for full verification. We'll use the term "examination" to include visual and physical inspection and comparing a seal number to a seal inventory list.

4, Figure 5, Figure 6) after the contents have been measured or verified. An inspector will be escorted by facility personnel to the material holding location, find seals to examine, compare the seal number and associated container number on the item with the numbers on a paper list, and mark that the seal has been examined and confirmed. The inspector also physically inspects the seal and seal wire for signs of tampering, pulling on the wire and seal to ensure proper connection to the container. A small set of seals may be selected for removal and verification at the IAEA headquarters; this selection process is performed using a statistical algorithm that informs how many and which seals should be removed and replaced.

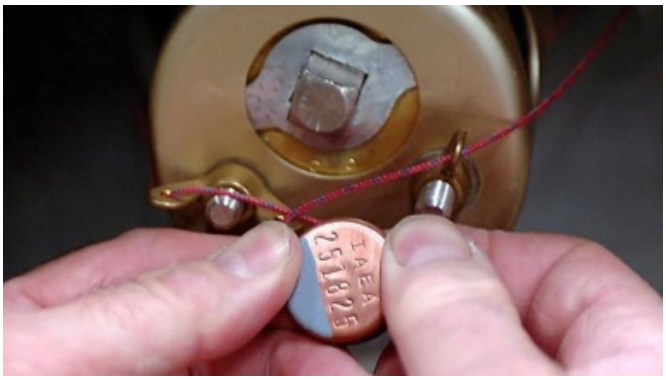


Figure 2: Metal cup seal. Image from [7].

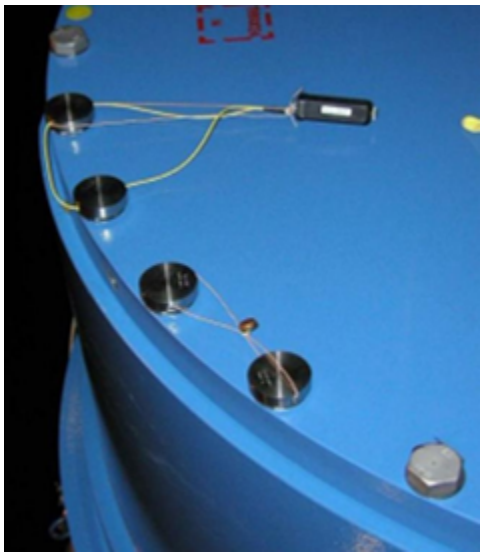


Figure 3: Casks are sealed with two different seal types. Electronic seal is shown on top, and metal cup seal is shown connecting two bolts on the bottom of the image. Image from [8].

Figure 4: Casks can be large, with access to the seal only from the top. Image from [8].

Figure 5: Some facilities have tightly packed casks, and access to seals at the top of casks is achieved via crane or bridge. Image from [8].

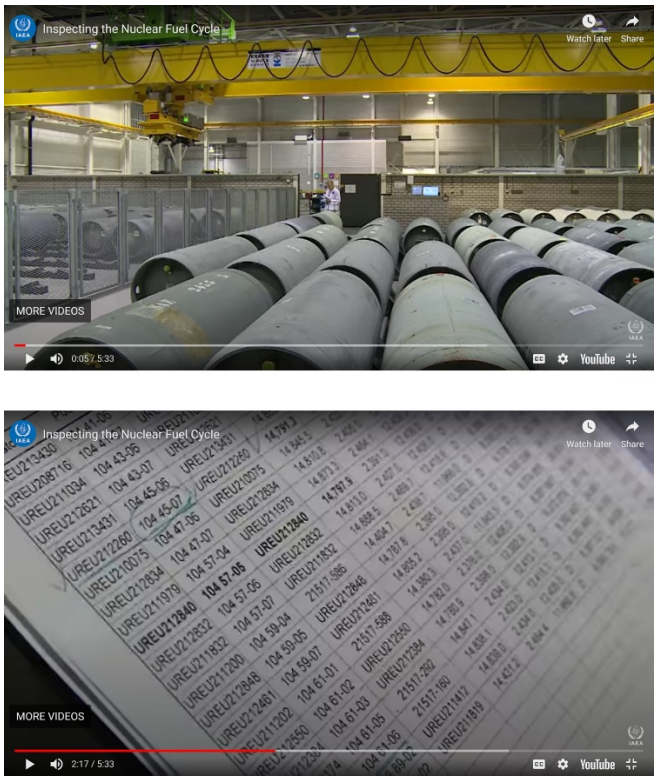


Figure 6: Enrichment facility halls contain large numbers of UF6 cylinders, each with a metal cup seal that needs examined[3].

Inspecta can provide varying levels of assistance to the seal examination task to support inspectors. Examining seals with the Inspecta app on the Google smart phone includes the following steps and skills (in parentheses):

- Ingest seal, container list and documents prior to inspection (OCR)
- Inspector escorted to seal/s location
- Inspector opens Inspecta app and asks to start seal examination (wake word, speech recognition and synthesis) or manually selects the task on the smart phone screen
- Inspecta uses OCR to capture seal and container numbers and they are automatically marked as examined in Inspecta app (task tracking)
- Inspector can ask Inspecta for information, i.e., "Inspecta, what is a significant quantity of nuclear material?" (wake word, information recall)
- Inspector can add a label to the Spot-generated map if desired (to pinpoint seal location, for example) (mapping)

At the other extreme, Inspecta could fully automate the seal examination task alongside an IAEA inspector. This full automation could use robotics and indoor navigation to locate the area with seals (though a facility escort is still required), use a robotic manipulator arm and computer vision to find and grasp a seal, apply OCR to acquire the seal and container numbers and compare to a local database, physically confirm attachment by tugging on the seal and wire, employ anomaly detection to identify

[3] https://www.iaea.org/newscenter/news/new-video-safeguards-inspection-uranium-enrichment-plant

signs of tamper, and communicate with the inspector using speech synthesis, speech recognition, and information recall. Figure 7 illustrates skills that would be applicable in this fully automated case. The latest version of Inspecta includes the following processes:

- Inspector asks Spot to perform seal examination (speech recognition, autonomy)
- Spot locates a container (object detection)
- Spot traverses room to container and locates a seal on the container (autonomy based on location mapping and object detection)
- Spot estimates the pose required for the manipulator arm camera to best align with seal number and acquires images of the seal (segmentation, pose estimation)

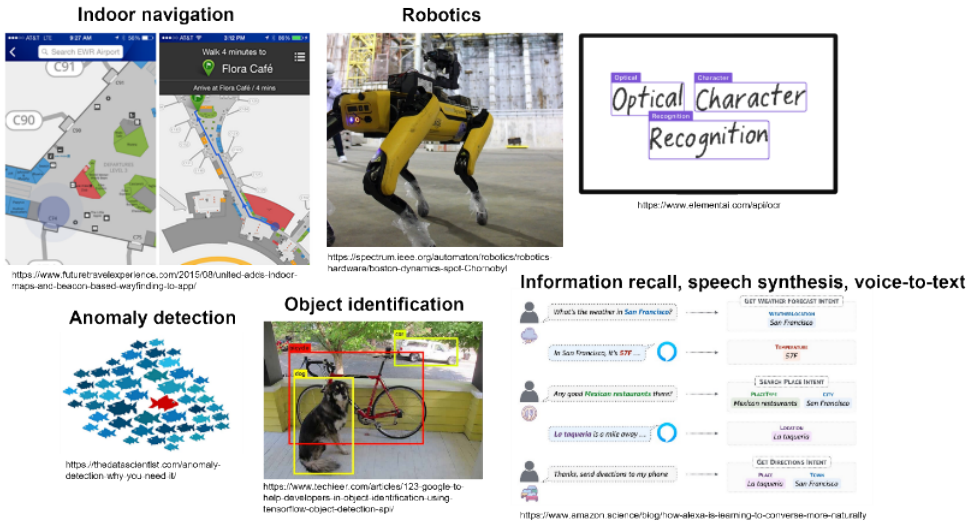


Figure 7: Skills applicable to "seal examination" task.

2. ARCHITECTURE FOR ANDROID PLATFORM

Before development of an Inspecta architecture, we considered high-level requirements that would drive the design based on mission constraints. These were mainly centered on security, privacy, usability, and the ability to use and modify previously developed algorithms, libraries, and software components. Hardware that already included many of the input/output capabilities and device sensors was also considered. An early decision was whether Alexa, Siri, or the open-source Mycroft[4] could be used as a baseline to build from, but project constraints eliminated those platforms as options. Alexa and Siri are cloud-based and require connection to servers to work properly. Wireless connections in nuclear facilities are often not available (or reliable), either due to security concerns from the facility or due to signal dead zones throughout the facility. Further, both platforms are proprietary which would limit access to source code and could also be a security concern for both the facility operator and the IAEA. Privacy would be questionable since data is sent/received through cloud-based servers. While Mycroft is open source, there are few turnkey skills available, and relatively limited documentation available. For these reasons, construction of the SDA platform from the ground up was deemed the best development pathway.

The following sections outline the various aspects of Inspecta implementation – hardware platform, development platform, and application development.

2.1. Hardware Platform

Development of Inspecta requires a development platform, a hardware device, and both a software application and ML algorithms loaded on a hardware device. We sought a hardware device that had integral input/output capabilities – namely, speakers, microphone, camera, and display, and chose a Google Pixel 6 smart phone. While cellular capabilities are often prohibited in nuclear facilities, there are existing seal readers that use modified cellphones where the antenna has been grounded to prevent communication. Smart phones also have various internal sensors for tracking and navigation, a feature that may be beneficial for future Inspecta versions. Smart phones are small, portable, and can be wearable, which is important since inspectors will be carrying the device throughout the day. While we expect the inspector to interact with Inspecta primarily via voice commands (e.g., "Inspecta, take a note"), a backup method of interaction is beneficial, and some information is best provided on screens (list of seal numbers, for instance). Smartphones typically have touch screens or on-device keyboards which provide redundancy. Finally, smart phones have reasonable computing power and extended-life batteries and battery extension packs.

2.2. Development Platform

Code for the Inspecta app was originally developed with Xamarin[5] to allow multiple end-user platforms (iOS, Android, Windows, etc.) since the team was uncertain about which platform might ultimately be deployed. Xamarin is open-source cross-platform code based on .NET that can be deployed as a native application for several platforms: Android, iOS, tvOS, watchOS, macOS, and Windows. Xamarin allowed the team to build Inspecta with a single user interface (UI) code base, which significantly reduced the implementation burden. While there is platform specific code that must be written (i.e., code to specifically interact with device hardware), Xamarin enabled abstraction of the UI and several other components (e.g., device permissions). Xamarin is no longer supported,

[4] https://mycroft.ai/about-mycroft/
[5] https://dotnet.microsoft.com/en-us/apps/xamarin/xamarin-forms

and all code has been migrated during FY24 to .NET MAUI. The MAUI platform is the successor to Xamarin so the capability to deploy to multiple platforms remains.

2.3. Application Development

Initial development of the Inspecta application itself is split into several modular pieces. Generally, the UI development occurs separately from the actual Inspecta capabilities (e.g., speech recognition, OCR, etc.) to allow for quicker testing. Capabilities are integrated into the main application once they have been tested in a controlled environment. Specifics of these modules are detailed in following subsections.

On-device ML has been prioritized for data security and privacy purposes and due to potential lack of communications available within a nuclear facility – this is very important to note as many ML algorithms require desktop-level Graphical Processing Units (GPUs) necessitating cloud connection, and thus we were constrained by the mission. This precludes many more advanced techniques to the problems below as techniques like Retrieval Augmented Generation (RAG) using Large Language Models and end-to-end OCR are not yet computationally feasible on edge compute devices (e.g., phones).

As Inspecta is designed to be cross-platform, a general framework for on-device ML was required. The Open Neural Network Exchange[6] (ONNX) application program interface (API) is being used to that end as it supports a wide range of programming languages (Python, C++, C#, C, Java, JS, Obj-C, and WinRT). This contrasts with popular ML frameworks like PyTorch or TensorFlow which often only support direct mobile applications (i.e., direct Android or iOS development rather than Xamarin development). Generally, ML models for Inspecta are first developed in Python, using common frameworks (e.g., TensorFlow, PyTorch or Transformers) and then converted to ONNX format. Models are quantized [9] where possible and optimized for mobile performance before being added to Inspecta modules for testing.

2.3.1. Inspecta main module

The main Inspecta module is the application envisioned when Inspecta is described in this report. The base application is formed by the Grial UI kit[7], which has a library of common UI elements developed by graphic designers. This UI kit enables the team to develop a professional looking application without needing to hand craft elements (basic widgets in Xamarin are very simple and need additional work to develop if not using a UI kit). FY22 efforts focused on developing UI components relating to the demonstration at the end of CY22. Specifically, this includes elements relating to the seal examination task and a basic inspector note taking interface. Images from the current build of the main Inspecta module are shown in Figure 8. The team performed minor modifications to the UI during FY23 based on user feedback and expanded capabilities. One specific example is the association of specific seals with specific containers, which required a new UI update.

[6] https://onnx.ai/
[7] https://grialkit.com/

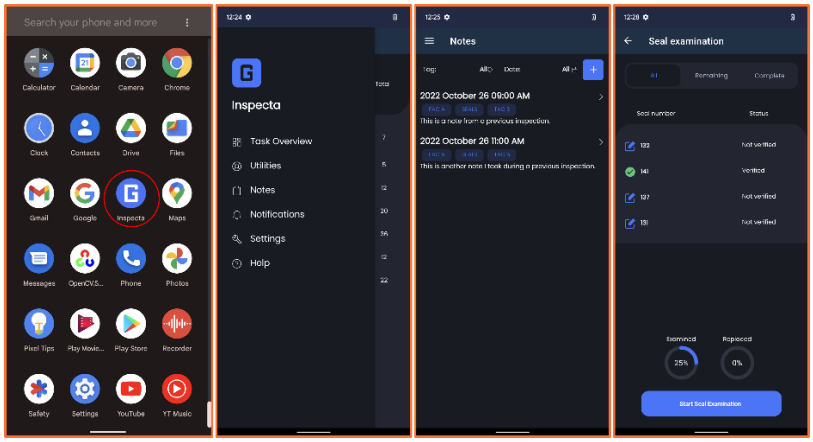


Figure 8: Inspecta app, built in a UI kit. (Left) Google Pixel 6 screen with Inspecta app in center right, (Left Center) Inspecta's main screen, (Right Center) Note taking interface, (Right) Seal examination module.

2.3.2. Speech synthesis module

Speech synthesis in Inspecta is relatively straightforward as most platforms have offline speech synthesis. Consequently, the team is leveraging .NET MAUI's cross-platform call for speech synthesis, and in only a few lines of code, speech synthesis is implemented. This module could be improved in the future, if needed, with on-device models for speech synthesis using open-source models like FastSpeech2 [10], DeepVoice3 [11], or similar.

2.3.3. Speech recognition module

The Inspecta speech recognition model utilizes Meta's Wav2vec2 [12] model to perform on-device speech recognition. While Wav2vec2 is no longer the top state-of-the-art model, it performs well and has lower computational requirements than newer models (e.g., OpenAI's Whisper[8]). Speech recognition with the current implementation requires several steps:

1. Waveform is recorded using on-device microphone
2. Waveform is converted to a different format and preprocessed
3. Wav2vec2 outputs estimated text
4. Text prediction is compared against possible commands using Levenshtein distance. If the distance is too high for any given command, it is assumed that a question has been asked and the information recall module starts.

Future development will investigate methodologies that can improve on Levenshtein distance for improved command matching and more intelligent initiation of the information recall module.

As Inspecta is a foundational demonstration platform, we trained and tested on American English and varying languages and accents are not within Inspecta's current scope. However, Wav2vec2 does have 53 monolingual language models and Whisper has 96 models available that could be used for training and testing in future work.

[8] Recent improvements in the ONNX runtime platform have made the implementation of Whisper easier, but the model is still quite large even using the "tiny" version with quantized weights.

2.3.4. Low fidelity OCR module

Seal examination in Inspecta is primarily focused on identifying metal cup seals and verifying that they are on the correct containers by comparing numbers on both to a paper list (with future work including examination of seals for signs of tamper or damage and ensuring the wire is securely thread through the item). Inspecta can track which seals should be associated with which containers. The general workflow is that a user scans a container identifier that triggers Inspecta to automatically load an internal list of seal(s) associated with that container. Inspecta then expects to perform OCR on individual metal cup seals for the recently scanned container. Ongoing work is considering how Inspecta should react to unexpected scenarios (e.g., seal scanned that should be on different container). In all cases, the human is ultimately in control as the OCR predictions are saved for each seal for human review if necessary. Information regarding seal-container pairs is stored in an internal SQLite database that can be processed at IAEA HQ if needed.

Metal cup seals are challenging to perform OCR on as the contrast between the alphanumeric characters on the seal and the seal body itself can change drastically based on lighting conditions making it difficult even for humans to read in some circumstances. An early design decision concerned how the OCR would be performed. Would a user (1) take a static image of the seal and wait for processing or (2) perform real-time OCR on a camera preview?

Approach (1) was deemed easier to implement, but less likely to be successful. Metal cup seal examination is not performed with a seal reader, as is the case with the Cobra and upcoming Field Verifiable Passive Loop seals (FVPS), leading to a more varied imaging environment. This could result in the user taking an image, receiving a poor OCR result, needing to take another image, potentially still getting a poor result, and so on, leading to a frustrating loop of poor results. A frustrating imaging loop wouldn't improve the user's experience over the current approach of manually documenting the seal status.

Consequently, the team decided to develop a real-time OCR approach that would enable the user to move the seal body in their hand while OCR is being continuously performed. This live OCR approach would lead to a better user experience that would avoid the tedious imaging loop at the price of a more challenging technical implementation. A good user experience in the live OCR approach depends heavily on the framerate of the preview; a slow preview will appear like a slideshow and result in a difficult-to-use solution. The live OCR approach will be referenced as a Low Fidelity approach as lower image resolutions are key to a reasonable preview framerate.

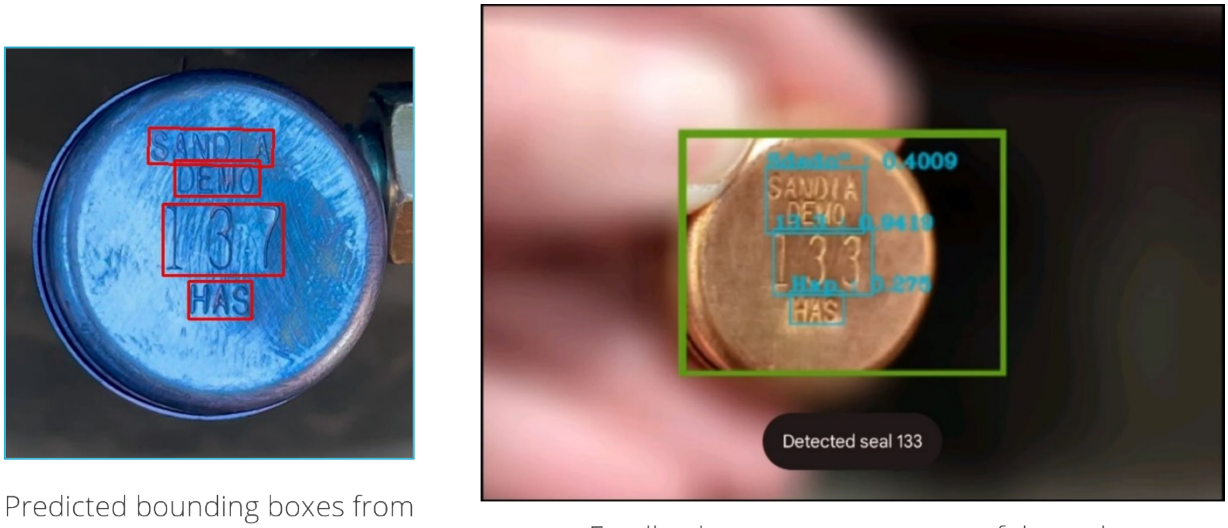
The Low Fidelity (LoFi) OCR module utilizes the on-device camera and ML models to read the alphanumeric designation of a metal cup seal and container. The current state-of-the-art OCR approaches often relies on a single end-to-end model to perform scene text (text with complex backgrounds) OCR. However, due to model size constraints by virtue of running on a phone in real-time, this task is split into two different tasks: text detection and text recognition. Text detection is the task of correctly detecting the text regions against the background of the image or video and setting the appropriate bounding boxes (i.e., the spatial region containing text). Text recognition involves taking the image cropped by the bounding box and estimating the characters within. Both tasks need to be performed effectively to result in a good prediction: poor text detection can result in bounding boxes that exclude some characters, whereas inadequate text recognition can inaccurately predict characters.

This two-step approach is implemented using the Character-Region Awareness For Text detection (CRAFT) and a three-stage Scene Text Recognition (STR) module. Pre-trained models are used for both text detection and recognition. The LoFi OCR module is integrated into Inspecta to show a real-

time camera preview to the user with ML models operating in the background. The streaming pipeline is nonetheless complex as heavy emphasis is placed on performance. In addition to the two-stage OCR model inference itself, there are several low-level image manipulations that must be performed. A preview session must be created such that the user can see the camera output. Image resizing and preprocessing also occurs to better align images with the sizes and scales seen by models in training. Processing speed is the guiding principle for the LoFi OCR pipeline. An overview of the approach is as follows:

1. Pipeline is based on EasyOCR library, heavily modified for the unique Inspecta use case.
    a. Convolutional neural networks (VGG/resnet) are used instead of transformers to reduce computation.
2. OpenCV on Android is used to create a preview (baseline 20 frames per second (fps) at 720x480 resolution).
    a. OpenCV is used here to more easily facilitate image manipulations required for the live ML processing pipeline. While there are some limitations to using OpenCV versus a native preview, they do not impact this OCR module.
3. Image preview frame is processed to provide a narrow analysis window and surrounding region is blurred to provide user feedback.
    a. Analysis window is only 160x120.
4. Image is preprocessed (contrast adjusted, resized, normalized).
5. Image is processed by a quantized text recognition model (CRAFT).
6. Output from the text recognition model, region and affinity scores, are translated to bounding boxes.
7. Small, cropped image patches are created from the bounding boxes of the text recognition model.
8. The small image patches are preprocessed for the text recognition model (resizing, contrast, normalization).
9. Output from both models is drawn on the preview frame.
    a. Both models use dynamic quantization to reduce weights to int8 for further performance gains.
10. Recognized text is compared against a list of seals and provides a popup if a seal is accounted for and the softmax score is above a certain threshold (0.80).
11. Pipeline for the frame is finished. Overall performance results in about a 7-fps preview; the total processing time per frame is about 120 ms.

Note that all processing is performed on-device with no reliance on external hardware or connectivity.



Predicted bounding boxes from Text Detection algorithm

Feedback to user upon successful match

Figure 9: OCR. The left image shows a higher resolution early prototype on-desktop with contrast adjustment whereas the right shows the final, on-device approach.

2.3.5. Wake word

Wake word detection is the process of using a specific phrase (e.g., "Hey Inspecta") to activate a speech recognition module. This capability represents a significant technical challenge as wake word detection is not often seen in the open literature. The most common use case for wake word detection is in digital assistants, which are usually proprietary.

Wake word detection involves several steps:

1. Determining if speech is present in each audio waveform.
    a. This is important as to not have a speech recognition algorithm running frequently on background, non-speech noise which would reduce battery life and consume unnecessary compute resources.
2. If speech is present, then do speech recognition.
3. Determine if speech has stopped, and if so, stop running the speech recognition algorithm.

The original implementation that was attempted relied on a series of hand engineered metrics. This approach was brittle and was only intended as a placeholder. Fortunately, through a series of conversations with ML practitioners at a conference, a much more effective approach was identified.

Wake word detection is now performed using a combination of a Voice Activation Detection (VAD) model combined with a custom wake word detection model. The custom wake word detection model was trained on approximately 100,000 synthetic samples of the phrase "Hey Inspecta" with a mix of background noise and sounds. An overview of the pipeline is follows:

1. Inspecta continually collects audio from the microphone when the application is running. A buffer that holds about 2 seconds of audio is constantly overwritten and is saved only to volatile memory.

2. The VAD model is continually run on the buffer to detect if speech is present. This model is quite small (1-2 MB) and is cheaper computationally to run than continually running the wake word model.
3. If speech has been detected using the VAD model, then the audio buffer is also evaluated by the wake word model to estimate the probability that "Hey Inspecta" is present.
4. If the wake word has been detected, the Wav2vec2 model processes a longer streaming audio buffer that might contain commands or information for Inspecta.
5. The VAD model is continually run while the longer streaming audio buffer is being processed. If no speech has been detected for some time, then the Wav2vec2 model stops processing data and the wake word processor returns to step (1).

2.3.6. High fidelity OCR module

One task that was identified as particularly tedious and error prone during our interviews with former inspectors was document transcription (e.g., copying data from tables, reading through reports, etc.). The team has developed some initial capabilities to recall information from documents through automatic pipelines. The first step is to perform OCR on documents and then store that information for later use. Document ingestion encompasses multiple potential tasks like automatically reading a plan or digitizing facility documents.

Document scanning, which involves converting an image of a document to a searchable, digital representation, is a much more complex task than simply reading characters on a seal body. Layout comprehension, text ordering, and document dewarping are just a few of the challenges involved. There are some end-to-end document scanning approaches like Facebook's Nougat and document comprehension like ClovaAI's Donut, but these models are generally large and challenging to run on edge devices.

The first approach was to use the LoFi OCR approach combined with document dewarping. Document dewarping refers to adjustments made to pages that are warped (e.g., paper is skewed or not flat). This process was complex but did result in an open-source contribution[9] that improved existing approaches through a 30x reduction in computation time. However, this approach was brittle and extremely slow on an edge device due to some of the underlying mathematical operations; while dewarping took 2-3 seconds on a desktop, it took over 3 minutes on the Google Pixel development phone.

A rework of the document scanning activity in FY24 resulted in the creation of a High Fidelity (HiFi) OCR approach. Instead of focusing on speed, as is the case for the LoFi OCR, the HiFi OCR approach focuses on accuracy at the cost of speed. The new HiFi OCR module utilizes the state-of-the-art, end-to-end, DeepSolo model to process text in documents and in arbitrary formats. The module is implemented as an offline approach that doesn't run in real-time. For example, users will take an image intended to be used with the HiFi OCR module, then instruct Inspecta to process these later. The HiFi OCR module powers several capabilities:

[9] https://github.com/nshoman/page_dewarp

- **Document Scanning:** The previous approach using dewarping and LoFi OCR was replaced by only using the HiFi OCR module. Images of documents are segmented into smaller image snippets for more accurate processing, each snippet is processed by the HiFi OCR algorithm, and finally, all outputs are concatenated and returned. Processing a single page takes about 30-40 seconds versus the previous 3+ minutes. An example of document scanning is shown in Figure 10, left.
- **Seal List Ingestion:** List ingestion refers to OCR on a table of seal-container pairs. This has been implemented as a proof-of-concept demonstration as it is unclear if seal list ingestion would occur in-field or as a headquarters activity wherein Inspecta's internal database would be directly updated.
- **Arbitrary Text OCR:** The HiFi OCR module can perform OCR on a variety of text formats in arbitrary shapes. For example, this module could perform OCR of a poster or book cover, as shown in Figure 10, right. The goal for the arbitrary text OCR is to combine these images with notes and/or location in a facility to improve inspection reports.

The processing time varies depending on what task is being performed. Future work will consider automatic processing of the HiFi OCR module based on if the phone is "idle" rather than requiring users to instruct Inspecta to process images.

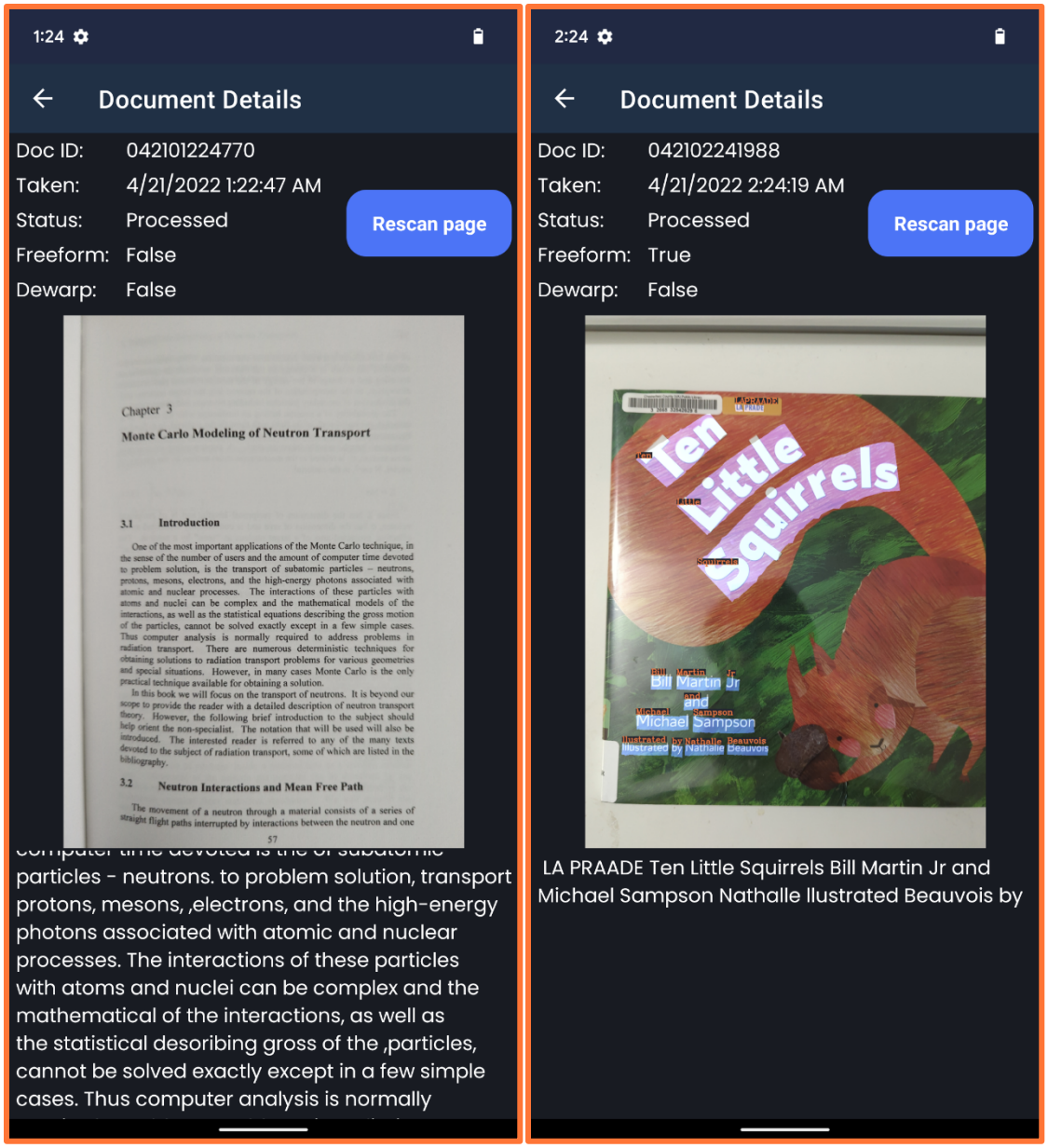


Figure 10: Document OCR (HiFi OCR). Left: Document scanning, right: arbitrary text scanning.

2.3.7. Information recall

Information recall is the process of providing a natural language question with the goal of obtaining a natural language response. Most state-of-the-art algorithms suitable for on-device ML require context which can be reasonably used to provide an answer (large language models such as LLaMa and ChatGPT are not considered as they cannot be run on-device). Information recall may be useful for inspectors in-field since documents might contains hundreds of pages, there might be multiple documents in which information is needed, and inspectors might be wearing gloves – making the process of turning pages to look for information difficult.

With information recall, an inspector can ingest needed documents into Inspecta prior to an inspection (either digitally or via HiFi OCR as described in section 2.3.6), ask natural language questions, and

receive spoken information back. As an example, we ingested the 2022 Safeguards Glossary [12] into Inspecta and then asked "Hey Inspecta, what is the definition of a significant quantity?" Inspecta then verbally responds, "the approximate amount of nuclear material for which the possibility of manufacturing a nuclear explosive device cannot be excluded."

The current implementation uses a two-stage approach; one stage to generate context from potentially many pages, and the second to perform question and answering. The current pipeline is implemented as follows:

1. A DistilBERT [13] model trained on the MSMarco [14] dataset is used to find relevant context across a wide range of documents.
    a. Context is pre-tokenized for use with the model where possible. Documents that are scanned are tokenized on-device.

2. A SciBERT [15] model trained on SquadV2 [16] is used to perform the question and answering task.
    a. Context and query are tokenized and embedded on-device before being used as input.

Currently, Inspecta only uses the IAEA Safeguards Glossary as a proof-of-concept for the information recall capability and is not connected to document OCR due to the current instability in the OCR pipeline (meaning OCR can be used to ingest documents separately, but information recall cannot access those documents). Question/Answer performance on the glossary is mixed due to a variety of factors. First, the safeguards glossary is not written in a "natural" way as one would find in a book or article which affects context. Some efforts have been conducted to reorganize how segments of the glossary are written to increase performance, but this has been met with limited success. After the revision to the structure and wording of the safeguards glossary, the model returned more complete answers (included more detail and acknowledged two-part questions) along with a high confidence proxy but the answers were less correct than when using the original language safeguards glossary.

Second, and the most likely contributor to the inconsistent performance, is a lack of a safeguards/nuclear vocabulary. While part of the pipeline uses SciBERT, an algorithm trained on a corpus of scientific documents, this is likely insufficient for querying more advanced nuclear concepts. Empirical evidence has shown that performance decreases with the amount of nuclear "jargon" present in the query.

There have been efforts by Pacific Northwest National Laboratory (PNNL)[10] to consider performance of language models after improving the model vocabulary with nuclear-specific terms. Further, PNNL has developed a methodology for modifying an existing question/answer dataset with more nuclear-specific terms[11]. The team investigated how the use of one or both strategies might have improved the information recall pipeline by collecting the most prevalent nuclear related keywords using 1,000 Office of Scientific and Technical Information (OSTI) abstracts related to nuclear safeguards to replace words at random for both the training and testing of the SquadV2 datasets. However, creating new datasets with an emphasis on nuclear corpora is insufficient on its own. Currently, the team is

10 https://www.osti.gov/biblio/1861801
11 https://esarda.jrc.ec.europa.eu/publications/artificial-judgement-assistance-text-ajax-applying-open-domain-question-answering-nuclear-non_en

utilizing SentencePiece, a text tokenizer, to create new embeddings with respect to the revised vocabulary files. These embeddings should improve the models' performance when trained on the nuclear vocabulary datasets.

3. ROBOTIC PLATFORM

Boston Dynamics' Spot robot (shown in Figure 11) was acquired in September 2022 to provide a physical implementation of Inspecta, especially for tasks that are tedious or hazardous. The benefits of incorporating robotic support are (1) reduced radiation exposure to inspectors, (2) ability to image difficult-to-access locations via the camera within Spot's manipulator arm, and (3) ability to perform tedious and repetitive tasks. While the commercially available Spot robot comes with a significant level of capability, additional skill development was required for our application, namely, autonomy, object detection, segmentation, pose estimation for seal OCR, and map acquisition.

Spot is a commercial quadruped robot equipped with state-of-the-art sensors, cameras, 30 lbs. payload capability, and a 6 Degree of Freedom (DOF) articulated arm with a single DOF end effector for grasping and image capturing. The robot is controlled with an Android tablet using either direct remote operation or choosing pre-programmed tasks. Operators can string these pre-programmed tasks and create repeatable "missions" (recorded and played back) for the robot to perform. In addition, Boston Dynamics provides a comprehensive Python API with many of these features, pre-programmed tasks, and missions for customers to incorporate into their own applications. As a result, Spot has been deployed for numerous and novel use cases such as law enforcement, warehouse and factory product inspection, and hazardous areas such as Chornobyl as shown in Figure 12.

Using Spot, we hope to create a semi-autonomous robot assistant for IAEA inspectors to rely on when out in the field. This has required formulating an autonomy framework conducive to an IAEA inspector's use-case without obstructing work; ensuring compliance with host facility procedures for safety and security (including escorting requirements); and being safe around humans or equipment in its vicinity. Applied to the seal examination task, Spot required the following modifications and skills:

- Addition/integration of GPU to Spot for processing/autonomy.
- Object detection (locate sealed container, locate seal) – train ML algorithms and test on surrogate objects.
- Autonomy – use behavior trees to string together Spot's tasks (detect container, traverse to container, detect seal, OCR).
- LiDAR to acquire point clouds, create 2D/3D maps of facilities and send 2D map product to Inspecta.

Figure 11: Boston Dynamics Spot robot with manipulator arm [14].

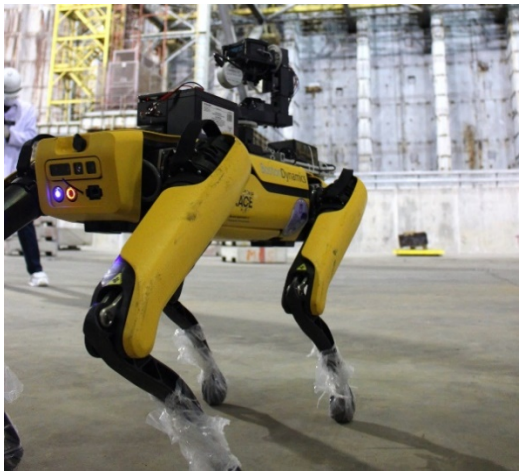


Figure 12: Spot, with a radiation detection payload, deployed at Chornobyl. Image from [15].

3.1. Autonomy

The Spot robot can natively be controlled using (1) a joystick-like tablet with an operator selecting functions like walking, crawling, sitting, and object manipulation or (2) "Autowalk" which enables the operator to record a manually controlled mission and replay it. Neither of these are ideal for safeguards inspections as we don't wish to add tasks for inspectors (operating a robot) and Autowalk requires a static environment (robot performs the same activity for items in the same place repeatedly). Therefore, we are developing a robotic autonomy stack such that an inspector can command Spot to perform a task, such as seal examination, with Spot able to autonomously perform the task. High level tasks, like seal examination, are broken down into sub-behaviors and sub-trees for additional modularity and reactivity, e.g., having Spot undock from its power source, stand, look around a room, traverse the room, perform an action like taking a picture and return to the dock. An informed search behavior using object detection and LiDAR will allow us to explore an unknown area and locate a container after being escorted by an inspector to a starting location. The modular nature of the behavior trees allows for greater *reactivity* to changes occurring around the robot. For example, if a container is moved while Spot is attempting to navigate towards it, this will result in Spot replanning to move towards the container using a new trajectory.

More specifically, the team has created multiple action primitives for the Spot robot using the Spot software development kit (SDK) and has been expanding this suite throughout the project. These primitives are being tested in a simulated inspection environment within the AutonomyNM Testbed[12], using 55-gallon drums with metal cup seals as targets for Spot to identify and process using OCR technology. Additionally, capabilities are being developed for Spot to follow inspectors wearing fiducial markers, with ongoing research into how to securely integrate these markers into clothing to prevent unauthorized control. The introduction of behavior trees has been well-received, leading to the open-sourcing of the "spot_bt[13]" package, fostering transparency and community involvement. Furthermore, in collaboration with Boston Dynamics' efforts to enhance Spot's operability through ROS 2 [12], a widely used robotics middleware, the team has migrated our work to ROS 2 under the

[12] https://autonomy.sandia.gov/autonomynm
[13] https://github.com/sandialabs/spot_bt

name "spot_bt_ros[14]." A significant portion of this fiscal year was devoted to this migration since the original version relies on the vanilla Spot SDK, while the ROS 2 version incorporates the SDK into an open-source robotic middleware popular within the robotic field.

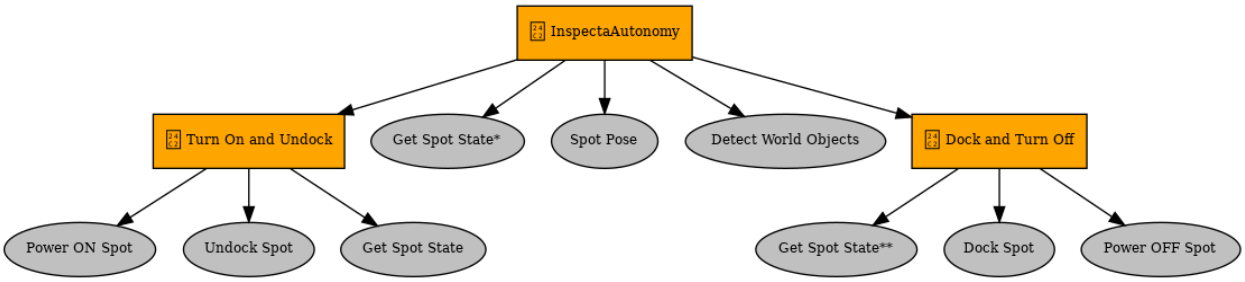


Figure 13: A simple behavior tree composed of multiple actions strung together to form an autonomous system.

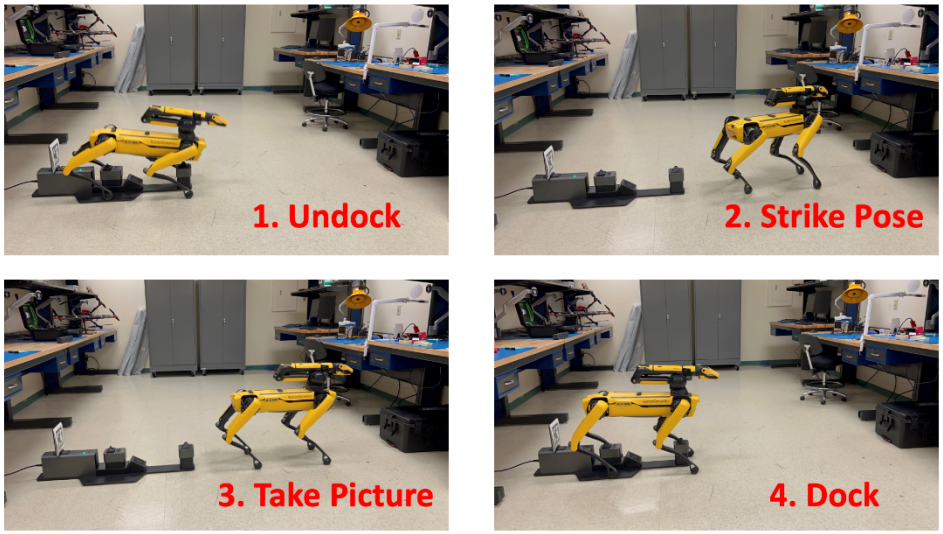


Figure 14: Spot executing a sequence of behavior primitives.

The purpose of expanding our codebase to incorporate ROS 2 support is twofold. Firstly, the robot middleware provides multi-processing support for different parts of our autonomy stack. For example, instead of running everything through a single script as was required by the Spot SDK, the robotic middleware segments processes into individual nodes with specific purposes that may run concurrently (or asynchronously) and intercommunicate with one another using a TCP/UDP network communication protocol. ROS 2 provides a streamlined, well-documented, and mature industry/academia standard implementation used on numerous robotic projects. Figure 15 shows an example search demo running on Spot. Each oval represents a process node and connections between them are defined as Publisher-Subscription connections. Though it may seem like all nodes are not connected, they are connected using Service-Client connections that are not visible in the figure.

Segmenting important functionality into separate processes has the bonus of significantly improved failure recovery. If one node/process of the autonomy stack fails, the system can quickly reboot the faulty node, re-establish its connection to the rest of the nodes, and continue to operate the robot's

[14] https://github.com/sandialabs/spot_bt_ros

autonomy with reduced delay. ROS 2 also provides "Lifecycle" nodes that may start or stop depending on user-defined calls. This is significant for applications where compute may be limited. For example, nodes that require access to GPU resources consume crucial compute resources that may be required by multiple different processes. Making those compute intensive nodes as Lifecycle nodes can reduce our overall compute and power usage by only using those nodes when required and shutting them off when not in use. In addition, this codebase change allows us to reuse nodes for other safeguards related robotics tasks or to easily share nodes with others developing safeguards related autonomy.

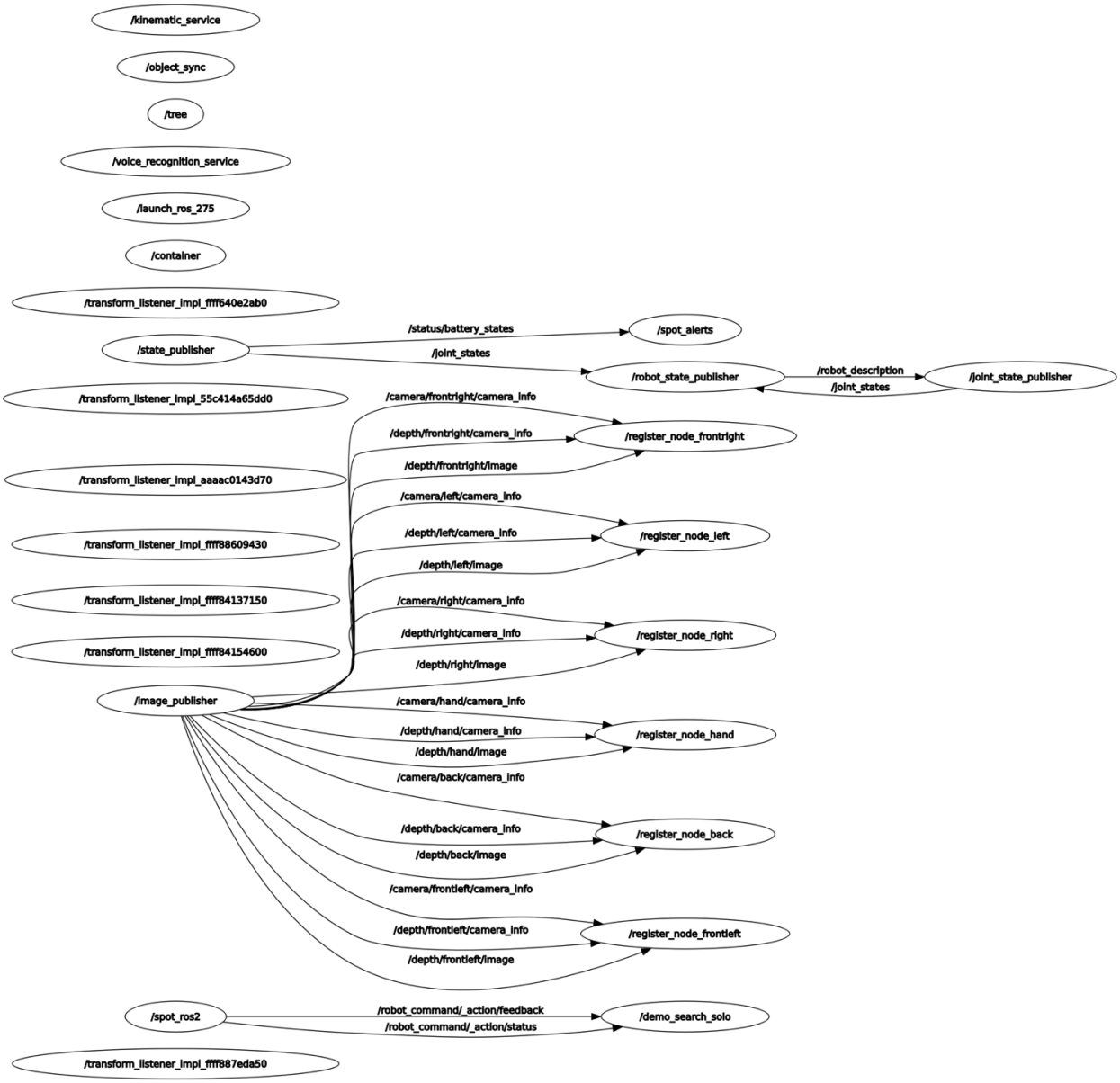


Figure 15: An example computational graph generated by ROS 2 of a single Spot robotic demonstration. Ovals represent process nodes while the arrow connections are Publisher-Subscription connections. All the nodes are interconnected using a Service-Client connection.

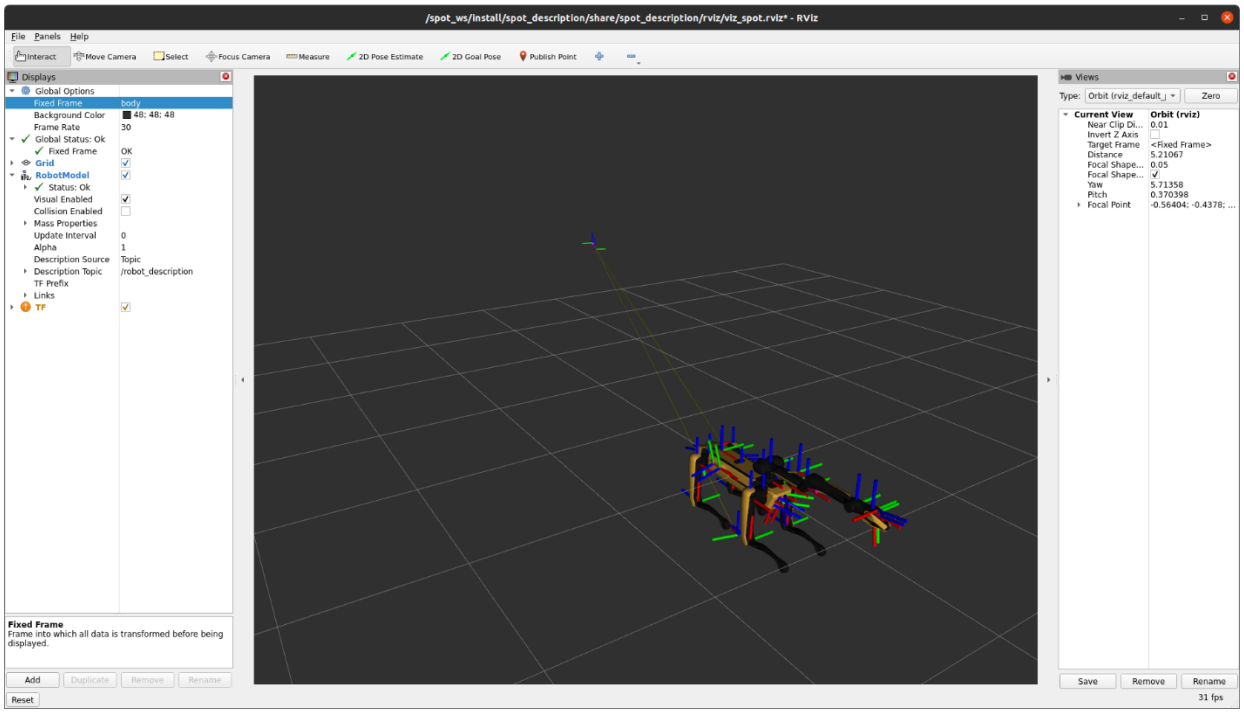


Figure 16: A ROS 2 visualization tool, RViz, used to visualize the robot in different frames of reference, sensor outputs, and other extensible features.

The second major benefit of moving to ROS 2 is that it gives us access to a plethora of useful robotic software that can enhance our autonomy stack without requiring us to write necessary and fundamental robotic software from scratch. For example, "MoveIt 2"[15] and "Nav2"[16] are popular software implementations of robot articulated arm manipulation and navigation, respectively, used in many industry and academic robotic projects. Moving our codebase to ROS 2 has given us access to these necessary software packages for future use in the project. Figure 16 is an example of our autonomy code visualized using an indispensable robotic visualization tool, RViz. Unlike a simulator, this tool helps developers visualize what their robot understands to be their environment, their relation to it, and what they sense around them using their sensors.

In our supplementary material[17], we demonstrate Spot searching for a 55-gallon drum with a fiducial marker placed on the outside. Spot will move in a random search pattern to find the drum (and will soon perform intelligent search by integrating object detection and LiDAR functions such that Spot finds the drum using object detection and plans a path to it based on LiDAR-generated maps). Once found, Spot will move towards the drum and touch it with its arm. This work requires multiple nodes comprising our autonomy stack: a voice command recognition service, a behavior tree autonomy node, a visualization tool instance, a Spot driver to move the robot, and a camera image service. Another added benefit of our ROS 2 port is that we may distribute the nodes among multiple computers. Our demonstration has the behavior trees, Spot driver, and camera services running on an Nvidia Jetson mounted on Spot's back, while the Rviz and voice command recognition service run on a separate computer in a nearby control room. This distributed compute paradigm allows us to

[15] https://moveit.picknik.ai/main/index.html
[16] https://docs.nav2.org/
[17] The project team has produced a demonstration video of Inspecta's capabilities in late 2024, available upon request.

eventually incorporate the Inspecta app into our autonomy stack for greater user control of Spot in the field.

In the second robotic example of our supplementary material, a ROS 2 node running our trained object detection algorithm streams images from Spot's arm camera, classifies whether it detects a 55-gallon drum or metal cup seal, places bounding boxes around the objects and then publishes the resulting image for users to see. We leverage a ROS 2 tool called "rqt" to see the classified images with bounding boxes. This is another example of how our code porting to ROS 2 gives us access to useful software and allows us to incorporate processes (the object detection) as a separate process connected to many others rather than one script.

3.2. Computer Vision

3.2.1. Object detection

Object detection is required so Spot can identify items of interest – for in-field inspections, this might be large containers of spent nuclear fuel with metal cup seals attached. For surrogate testing, we use generic 55-gallon drums and metal cup seals, which are further discussed below. We mounted a GPU to Spot to run these object detection algorithms (as well as enabling computation for autonomy, mapping, and interfacing to Inspecta). The GPU is a Jetson AGX Xavier for ML inference with various neural network models. More specifically, the Jetson GPU is equipped with a 512-core Nvidia Volta architecture chipset capable of running 64 tensor cores at 1377 MHz. Spot has mechanical interfaces for installing custom mounts and external hardware. Therefore, we designed and 3D printed a mount that securely holds the Jetson and an external LiPO battery pack to power the single board computer.

The You Only Look Once (YOLO) algorithm was selected for our object detection applications as it is state-of-the-art for real-time processing. The name contrasts with predecessor algorithms, which required multiple computation steps to detect objects and ran significantly slower. Object detection for this project employs the YOLOv5 algorithm from ultralytics[18], running on the Jetson mounted on Spot. Initially, the model was pretrained using the COCO[19] dataset, which comprises a wide range of everyday objects, to establish a broad visual understanding. To enhance the system's utility for our needs, Limbo[20] synthetic data is then utilized for transfer learning. The Limbo dataset consists of over 1 million synthetic labeled images of various containers, including UF_6 cylinders (30B, 48X, 48Y, 48G types), wine barrels, and 55-gallon drums, among other objects. Finally, to improve the model's performance further, additional datasets specifically focused on 55-gallon drums and metal cup seals were compiled and labeled in our lab. This final dataset was continually trained and tested on to ensure reliability of the object detection skill. This progression in training data from COCO, to Limbo, to in house is shown in Figure 17. Results from the drum and seal object detection system on test data are shown in Figure 18.

[18] https://github.com/ultralytics/yolov5
[19] https://cocodataset.org/#home
[20] https://limbo-ml.readthedocs.io/

Figure 17: Progression of object detection training. (Left) COCO dataset for common objects, (Center) Limbo dataset for UF_6 and other cylinders, (Right) 55-gallon drums and metal cup seals at the AutonomyNM testbed.

Figure 18: Test results for object detection of 55-gallon drums and metal cup seals, with confidence level displayed.

Our team has leveraged several annotation tools while compiling object detection datasets. LabelMe[21] lets the user draw bounding boxes on individual photos. This approach is simple but time consuming. LabelStudio[22] lets the user draw bounding boxes for key frames in a video and interpolate between them, which is much more time efficient than labeling individual photos. However, LabelStudio does not output video data in a YOLOv5 compatible format, so we had to write a conversion script. Lastly, RoboFlow[23] lets the user automatically annotate images with a foundation model such as Grounding DINO[24]. Grounding DINO is a vision-language model that can perform open set object detection of arbitrary text queries zero shot, i.e., without any query specific training. While traditional object detection systems are trained to identify a specific set of objects, Grounding DINO attempts to identify any object associated with a text query. Grounding DINO generated training data is shown in Figure 19. Note this model is too slow to run in real-time, too computationally expensive to run on an edge device, and fails to draw good bounding boxes in many circumstances. Despite these flaws it can powerfully accelerate dataset generation.

[21] https://pypi.org/project/labelme/
[22] https://labelstud.io/
[23] https://blog.roboflow.com/grounding-dino-zero-shot-object-detection/
[24] https://arxiv.org/abs/2303.05499

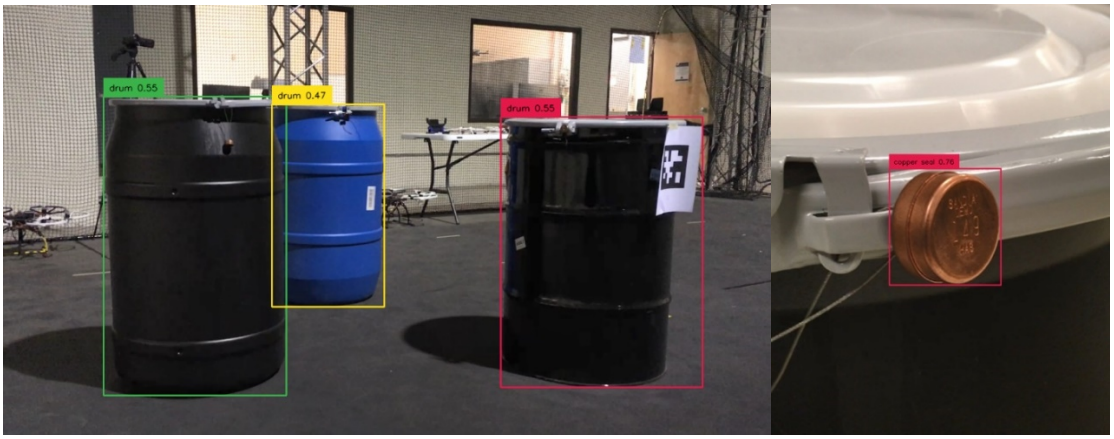


Figure 19: Grounding DINO auto labelled training data.

Finally, RoboFlow allows users to upload their own models and use them to auto label additional data. With this technique we can progressively bootstrap a more and more thoroughly trained object detection system.

3.2.2. Object segmentation

Another visual skill generally useful in robotics is segmentation. Rather than drawing bounding boxes around objects, segmentation seeks to categorize individual pixels into groups or instances. This is useful when trying to visually understand parts of a scene that cannot easily be captured by a bounding box, such as which areas of the visual field are walkable, and which are hazardous (Spot can use this information while planning a path to a detected object, such as a 55-gallon drum). It is also useful for isolating relevant point cloud data from red-green-blue (RGB)-Depth cameras or LiDAR. When RGB data is mapped into the same frame as point cloud data, segmentation can be useful for determining the exact spatial location of drums, seals, humans, or any other object of interest, since it more effectively rejects outliers compared to simpler object detection. Finally, segmentation is a prerequisite for the pose estimation and 3D vision algorithms discussed next.

A common ML algorithm for segmentation is Mask-RCNN[25]. A more recent foundation model for segmentation is Meta's Segment Anything Model[26] (SAM). Some results from these models are shown in Figure 20.

[25] https://arxiv.org/abs/1703.06870
[26] https://arxiv.org/abs/2304.02643

Figure 20: Pretrained segmentation results. Mask-RCNN in the middle. Segment Anything Model on the right.

Pretrained Mask-RCNN successfully segments out the humans and a few odd objects in the scene but struggles to identify Spot as a single object. This is likely because robotic dogs were not in the Mask-RCNN training data. However, Mask-RCNN's performance could certainly be improved by collecting task specific training data and fine tuning. Meta's SAM does very well on this example with no fine tuning. Foundation models, by definition, have been trained on such vast data that they generalize well to new situations, which explains SAM's strong performance here.

Figure 21: SAM segmentation from bounding boxes for drums and seals.

SAM can also segment out an object given only a bounding box around it, such as the bounding boxes produced by YOLO. This is shown in Figure 21. While SAM performs incredibly well, it is too computationally intensive to run locally onboard Spot. However, this YOLO-SAM pipeline can be used to auto generate segmentation training data for smaller models such as Mask-RCNN. Generating segmentation datasets is labor intensive without this auto labeling pipeline, as object perimeters are harder to label than simple bounding boxes.

3.2.3. Pose estimation

As can be seen in Figure 17, Figure 18, and Figure 19, metal cup seals on containers are not always at convenient angles for reading the inscription on their surface. Spot needs to be able to estimate the orientation of the seal and position its manipulator arm perpendicular to the face of the seal. This will enable clear photography of the inscription as well as grasping. Grasping may be implemented in future work to ensure metal cup seals are properly attached to their containers. The team is currently working on "pose estimation" for Spot to perform seal examination most effectively.

However, pose estimation is far from a solved problem. It is a highly active research area, with collections of algorithms that work better in certain situations, but no one size fits all solution. Many factors affect the performance of each algorithm, such as:

- The depth sensor being used and the size of the object relative to the noise and resolution of the depth sensor.
- The uniqueness and identifiability of features in the geometry.
- The visual texture of the object. Note, texture is a metric for the number of identifiable pixel level features in a photo. A smooth water bottle covered in stickers would have high texture, while a smooth, uniformly colored water bottle would have low texture.
- The availability of a CAD file of the object versus the need to estimate it from sensor data and whether the CAD file contains visual properties (e.g. color, reflectivity) or just geometric properties.
- Whether labeled training data exists or can be collected. Note labeled pose data is much rarer and more difficult to collect compared to typical 2D bounding box or segmentation data.
- Whether abundant training time is available prior to deployment or whether the system must quickly onboard new objects.
- Whether abundant calculation time is available prior to pose estimation or whether it must be done real time.
- Whether the object is occluded.

In order to understand these tradeoffs and select the best algorithm for Inspecta, particularly for metal cup seal pose estimation, the team has experimented with many different sets of software, on many different objects and scenes. None of the tradeoffs listed above were understood prior to these experiments, the most important of which are documented below. We do not yet have a fully functional pose estimation system for metal cup seals, but tremendous progress has been made on understanding and therefore eventually solving the problem.

This paragraph presents a quick preview of the experiments. The most classical algorithm for pose estimation is point cloud segmentation with Iterative Closest Point (ICP), which we explore first. However, it seems to be better suited for large objects with unique geometry, unlike our small, round metal cup seals. We then move on to color-based methods, and test variants with and without access to a perfect textured CAD file. With a textured CAD file many algorithms work well. In the case without a CAD file, as is the case for metal cup seals, we begin with a classic algorithm for RGB structure from motion. However, this algorithm relies on pixel level features, which are abundant in highly textured objects, but largely absent in our uniformly colored copper seal. Finally, we experiment with NERFs, which were only invented in 2020. They enable the novel view rendering achievable with a perfect textured CAD file, but do not require the CAD file, and they work better on low textured objects (such as our metal cup seal), compared to RGB structure from motion.

3.2.3.1. Point cloud ICP

This method relies on depth sensor data in addition to traditional RGB images (object detection can identify objects but does not have this depth information). RGB plus Depth (RGBD) sensors based on projected infrared stereo vision are common in this domain. Spot has five such sensors built into its body, and the team is considering mounting an additional RGBD sensor to its manipulator arm.

An example of a depth channel image is shown in Figure 22 on the left. This depth scan contains a 55-gallon test drum with a stuffed animal on top in the foreground in green, a human engineer in the middle-ground in blue, and patches of floor and ceiling in the background in red. It is difficult to do anything useful on the full point cloud. Conveniently, the segmentation techniques presented in the previous section can isolate the point clouds of specific objects. As previously mentioned, segmented out sections of the point cloud can be useful for estimating the exact spatial location of drums and nearby humans, or for estimating the orientation of seals. Once the point cloud of the seal is isolated from the background, ICP can be used to match the point cloud data to a known model for the seal, thus inferring its orientation. An initial implementation of this ICP[27] approach is shown on simulated sensor data in Figure 22 on the right. The blue 'model' point cloud is generated from a CAD file of the seal. The orange 'scan' point cloud is a simulated sensor scan. The green 'scan aligned' point cloud is the sensor scan after running ICP and solving for seal orientation.

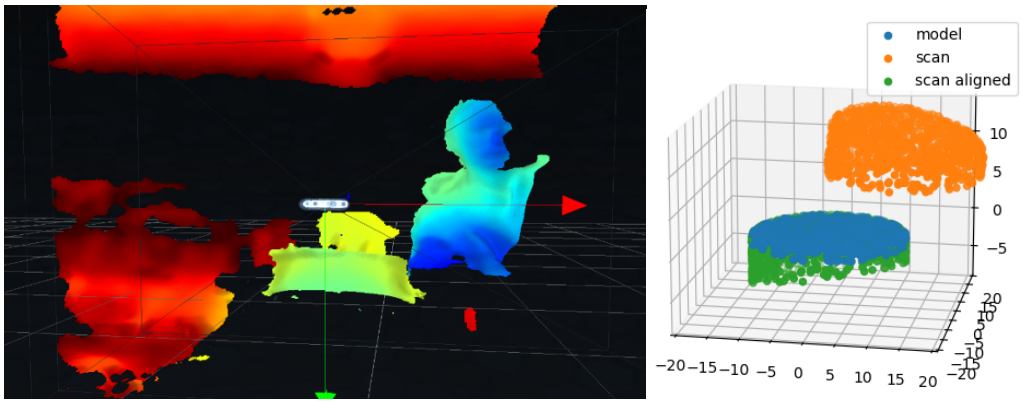


Figure 22. (Left) Depth channel data from an RGBD sensor. (Right) Initial implementation of ICP for seal pose estimation on simulated sensor data.

Point cloud ICP methods are a popular approach to object pose estimation, but they ignore the color properties of the object whose pose they seek to estimate, and they are better suited to large objects with unique geometry. In the case of metal cup seals, geometric properties alone can be ambiguous. Many objects have a similar size and shape, while few objects share the copper color and seal shape combination. Furthermore, the noise levels in many stereo cameras are similar in magnitude to the total size of the metal cup seals. Additionally, the low-resolution levels on these depth cameras mean they capture very few points on the seal compared to larger objects. This makes geometry-only methods prone to error. For these reasons the team has been investigating another class of pose estimation algorithms natively based in RGB or RGBD space, as opposed to the point cloud method which is based only on depth. Being more modern, these methods may also prove more robust, and more easily extensible to any future pose estimation needs for Inspecta, though metal cup seals remain the primary focus.

3.2.3.2. RGB structure from motion

Many RGB-based pose estimation pipelines begin with a textured mesh (interchangeable with textured CAD) of the object of interest. Sometimes these mesh files can be obtained from the original manufacturer of the object, but often they must be generated directly from sensor data, using, for

[27] https://pypi.org/project/simpleicp/

example, structure from motion techniques. This is the case for Inspecta's metal cup seals. The team has seen the best results in this domain using COLMAP[28] photogrammetry.

Note this technological evaluation was conducted on an arbitrary object from the home of the author for convenience, to enable the use of public compute platforms not suitable for SNL data and for the sake of the understanding gained by testing on different types of objects. In this case, the object is a musical device known as a looper pedal. Given a ~30 second cell phone video scanning the object, a textured mesh file can be generated via keypoint feature matching and bundle adjustment for sparse SLAM (Simultaneous Localization and Mapping), followed by multi-view stereo for dense reconstruction. The output of this process is shown in Figure 23.

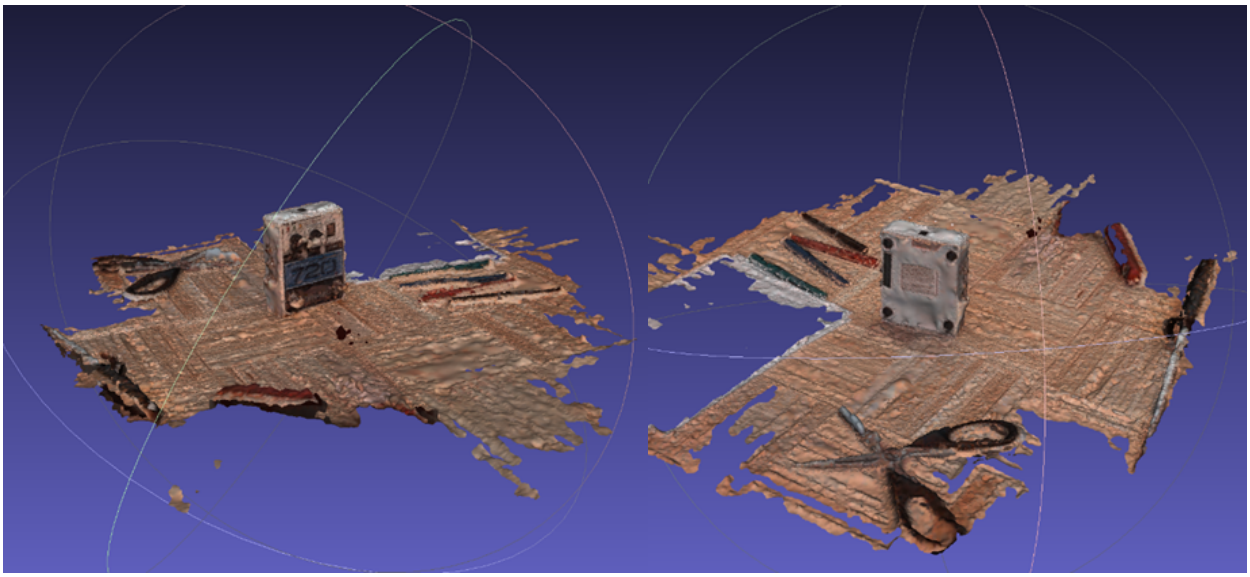


Figure 23: COLMAP Structure from Motion.

Following general scene reconstruction, the specific object of interest can then be extracted manually using CAD editing tools such as Blender or MeshLab. This reconstruction clearly captures the geometry and color distribution of the looper pedal (and recall we are performing this so that we can capture geometry AND the unique copper color distribution of the metal cup seals for pose estimation), but it is only a rough first draft and will need to be refined with further tuning. Consistent lighting, more compute time, and less blurry still photos, rather than extracted video frames, can all improve the result. Note, this takes on the order of 10 minutes to generate.

With the textured mesh in hand, dozens of images of the mesh file can be rendered from discretely varied angles. A few such example renderings are shown in Figure 24. This process of rendering from new angles is also called novel view synthesis. Notice how the more textured regions, such as the area marked '720', are captured well, while the more uniformly colored regions, such as the all-white side panel, appear amorphous. This difficulty capturing low texture geometry is an inherent limitation of the RGB structure from motion approach.

[28] https://colmap.github.io/

Figure 24: Textured mesh renderings.

Finally, when an instance of the object of interest is seen in the wild, it can be segmented out and matched to one of the many available renderings by some similarity metric. An example image with the object segmented out is shown in Figure 25.

Figure 25: Image with object of interest segmented out.

In this case the segmented view of the looper is relatively close to the farthest left mesh rendering, allowing, in principle, for a course estimation of the pose of the looper pedal. However, initial experiments with this render and compare strategy failed to yield accurate results in this case, likely due to the imperfections in the model in Figure 24. More time is needed to achieve the final step of working pose estimation in this case, but many of the building block technologies for the render and compare strategy are in place. This basic pipeline is taken from FoundationPose[29].

An example of the complete use of the FoundationPose render and compare strategy for pose estimation in the case with prior access to a perfect CAD file is shown in Figure 26 below. The photos show the common YCB[30] mustard bottle, which is ubiquitous in 3D computer vision, and for which perfect textured mesh files (shown on right) are available. This test indicates that the overall render and compare strategy in FoundationPose works well for pose estimation. Our team just needs to improve our novel view synthesis, as shown in Figure 24, to successfully apply this method to new

[29] https://nvlabs.github.io/FoundationPose/
[30] https://www.ycbbenchmarks.com/

objects - especially metal cup seals. One approach to this improvement is NERFs, which are explored next.

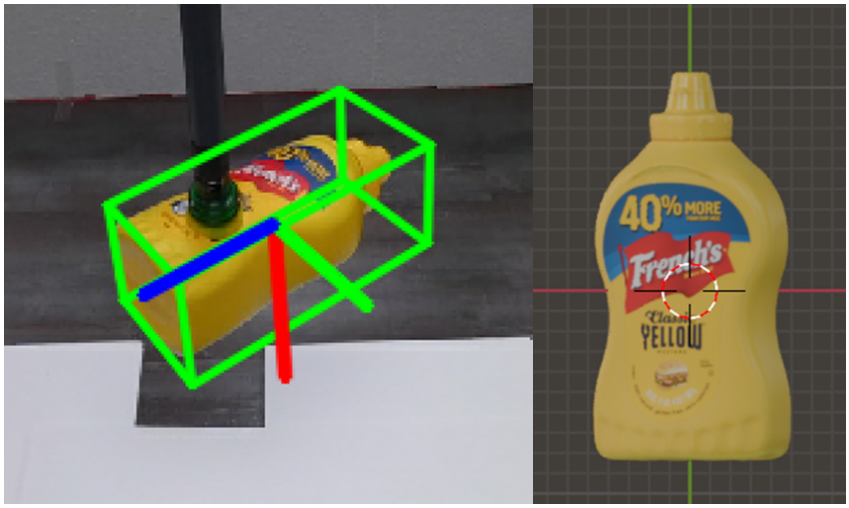


Figure 26: FoundationPose pose estimation for object with known model.

3.2.3.3. NERFs

The render and compare for pose estimation pipeline above critically rely on the ability to model an object and then render that object from many arbitrary angles, without prior access to a texture CAD file. Dense COLMAP style photogrammetry is a classical approach to this problem. A more recent approach is called NERF: Neural Radiance Fields. NERFs take in a set of images with known camera positions, orientations (extrinsic matrices) and 3D-to-2D projection properties (intrinsic matrices) and train a neural network to predict the view-dependent probabilistic radiance at every point in space. This then enables novel view synthesis via classical ray tracing rendering techniques. While more experimental, this approach often produces state-of-the-art results, especially for low texture objects such as our metal cup seals, so the team has been investigating it as well.

In principle any method can be used to obtain the position and orientation of the camera, which is required for NERF, e.g., motion capture system or scanning equipment. Likewise various camera calibration techniques exist to obtain the camera's intrinsic matrices. In this work, facilitated by NerfStudio[31], the initial step is the same as RGB structure from motion: use COLMAP to perform sparse RGB SLAM. This simultaneously obtains the camera position and orientation for each photo in the scan, while also estimating the camera intrinsics.

The results of this sparse SLAM process applied to a scan of the author's desk is shown in Figure 27 below. Each red polyhedron represents the position and orientation of the camera at one point in time. Each point in the point cloud represents a keypoint used by the COLMAP system. Note the bouquet of flowers in the center of the image appears very densely sampled by the SLAM system due to its intricate detail and rich colors, which yield high visual texture.

[31] https://docs.nerf.studio/

Figure 27: Sparse RGB SLAM.

With photos, extrinsics and intrinsics in hand, a NERF can be trained. This is shown in Figure 28 with camera positions and orientations overlayed. Note this bird eye view was not available in any of the individual photos in the scan but is reconstructed from the NERF. Galactic effects in the perimeter of the image are due to lack of data in those regions. This novel view synthesis from data is exactly what is required for the render and compare approach to pose estimation.

Figure 28: NERF novel view synthesis.

Furthermore, Figure 29 below illustrates that depth (shown on the left) is also obtainable from the NERF training process. This illustrates that the NERF training process maps the entire geometric and

visual properties of the scene. Recall that the need to model both geometry and color motivated the original switch from point cloud ICP to the render and compare strategy. Additionally, this process captures the same information that would be available in a textured mesh of the environment but produces higher quality renderings compared to the previously explored RGB structure from motion approach. Observe how the NERF approach has no problem rendering novel views of low texture areas such as the uniform white wall.

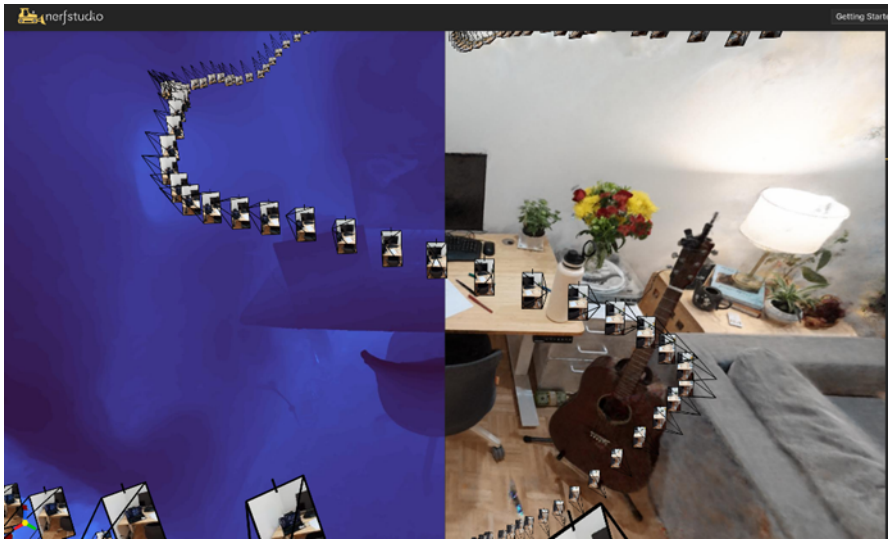


Figure 29: NERF depth estimates and 3D reconstruction.

This NERF represents the whole scene. However, if object segmentation is applied after sparse SLAM and before NERF training, the novel view synthesis can become object specific, making it very useful for pose estimation of metal cup seals. More time is required to implement this final step in the render and compare through NERFs approach to pose estimation for metal cup seals. However, it was successfully tested through a FoundationPose implementation on a generic object in Figure 30 below. The left shows coffee can pose estimation via NERF (tested locally); the right shows a selection of the segmented images used to train the NERF (trained by FoundationPose). Like the YCB mustard bottle previously mentioned, this YCB coffee can is ubiquitous in 3D computer vision research, meaning good implementations of object specific NERFs are already available for it online. This makes it a friendly place for initial testing of the end-to-end render and compare through NERFs approach, but it is only a steppingstone for Inspecta.

Figure 30: FoundationPose pose estimation from NERF for objects with unknown models.

3.2.4. LERFs

As a quick additional note not related to pose estimation, a variation of NERFs embed the 3D radiance field with language embeddings from vision language foundation models such as CLIP[32]. These are called LERFs[33] or Language Embedded Radiance Fields. These can be used to generate heat maps in images given arbitrary natural language queries. One such heat map for the query "water-bottle" is shown in Figure 31. Another heatmap for the query "flowers" is shown in Figure 32. In principle, this approach is not limited to single word queries but could also be used for phrases such as "fire extinguisher in the kitchen." For preplanned queries such as 55-gallon drums and metal cup seals, our existing methods such as YOLO for object detection and mask RCNN for segmentation will outperform this approach. Where this approach shines is in unplanned situations for which hand tuned algorithms have not been implemented. One could imagine sending Inspecta into a hazardous environment to look for "damaged electrical equipment" or "signs of malicious tampering". While LERFs are likely not mature enough to perform such complex operations yet, they are one of the research communities best approaches toward developing truly general visual scene understanding. LERF's are a promising emerging research direction for general autonomous robotic inspection, as demonstrated in VLFM[34], and our experimentation with them was a largely free extension of our NERF pose estimation investigations.

[32] https://openai.com/index/clip/
[33] https://www.lerf.io/

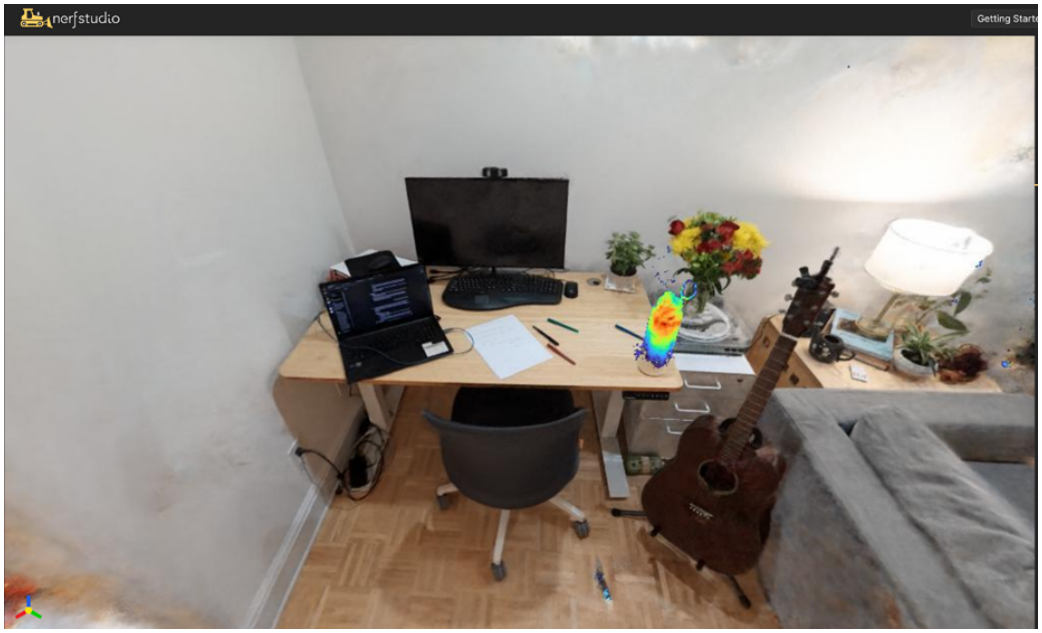


Figure 31: LERF with query “water-bottle”.

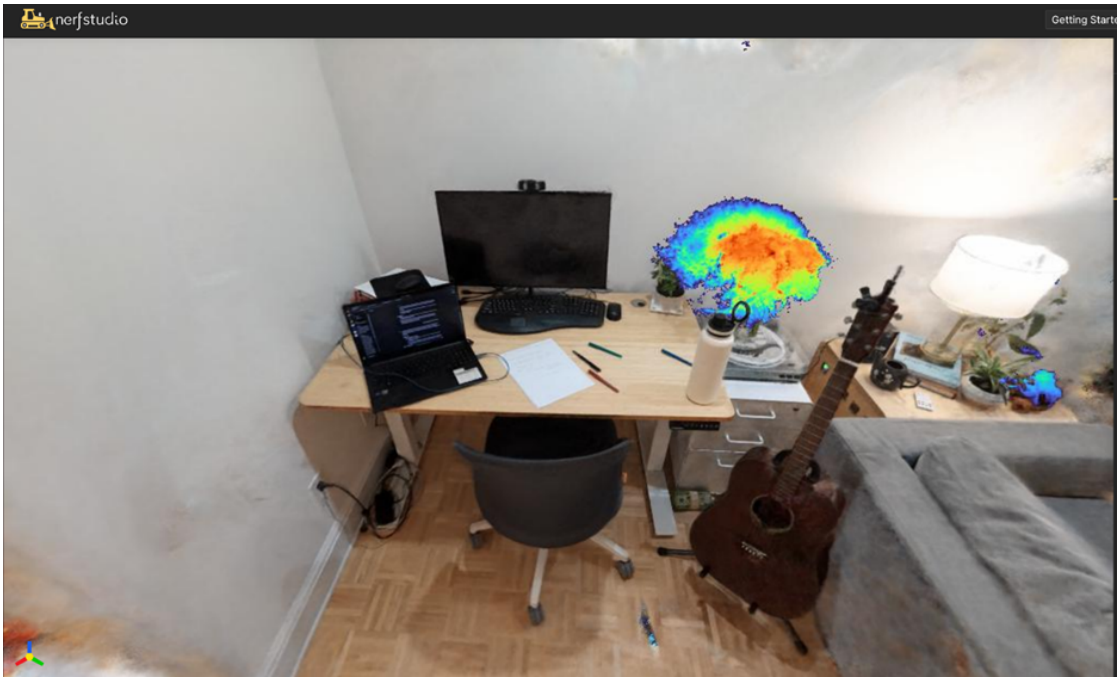


Figure 32: LERF with query “flowers”.

4. MAPPING

4.1. Introduction & Hardware Selection

Another skill researched and developed over the course of the project was the ability to generate 2D and 3D maps with the eventual goal to drop notes or waypoints into the map, which would be viewable in the Inspecta app. During our inspector interviews, we learned that maps are generally not available for most facilities – while inspectors are escorted throughout the facility by a host, a map could be useful to track paths traversed during inspections (and thus note when a path deviates from previous inspections), and to make notes or correlate inspection activities with a location. Though R&D exists in using inertial sensors from phones and/or stitching together images to create maps these capabilities typically require significant user interaction's that would distract not enable an inspector's ability to execute their duties. The team also researched the use of Spot's onboard point cloud acquisition capability. Spot's point cloud acquisition capability is used for navigation and produces point clouds that are unusable for map generation (Figure 33).

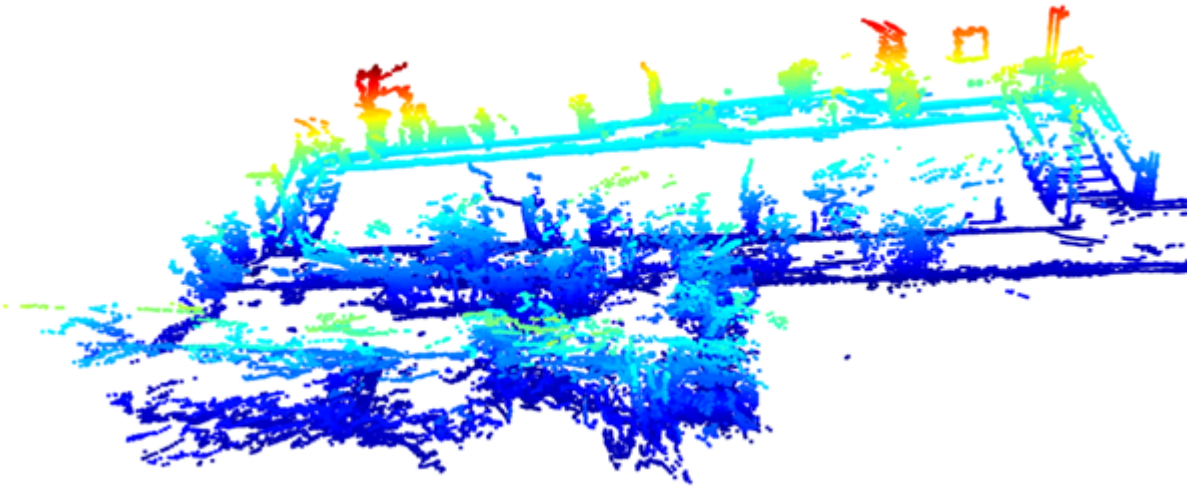


Figure 33: An example of Spot collected LiDAR points.

Boston Dynamics has created an external high fidelity LiDAR sensor. However, it must be mounted in place of the articulating arm, which is not practical for our application. Further, the sensor is designed to improve autonomous navigation and does not generate a point cloud intended for analysis. Instead, the team was able to obtain an Ouster OS1 LiDAR sensor that produces point clouds with a 45-degree view at 128 channels of resolution and possesses a maximum range of 200m, outperforming the former two sensing options. A sample of OS1 point cloud data can be viewed in Figure 34 in addition to near-infrared imagery which is captured simultaneously with the point cloud data.

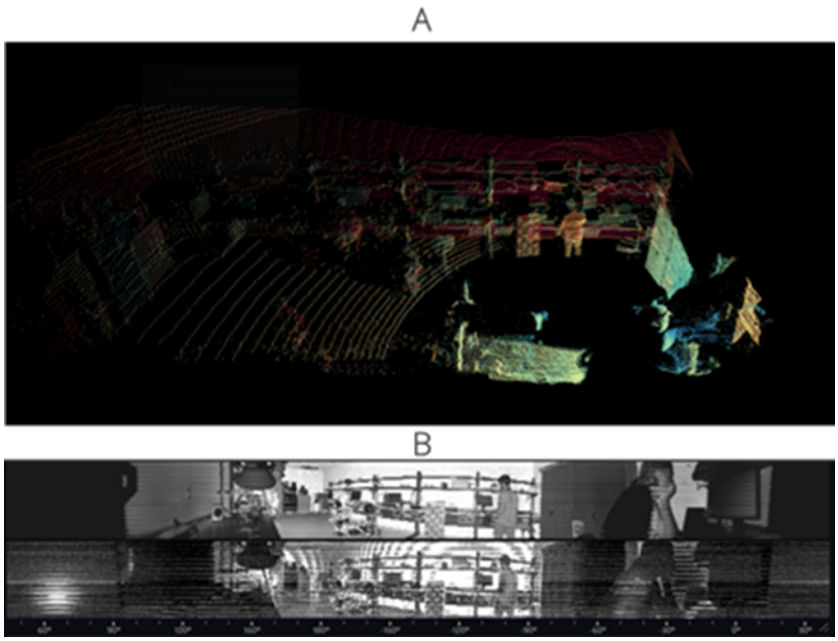


Figure 34: A) Example of the point cloud produced by the OS1. B) Top, image signal produced by the OS1. Bottom, near infrared captured by the OS1.

We designed/prototyped a 3D printed mount for the LiDAR on top of Spot. The LiDAR is connected to the GPU on Spot, with the LiDAR, its power supply and connection to the GPU shown in Figure 35, and the 3D printed mount with the LiDAR on Spot shown in Figure 36.

Figure 35: LiDAR unit with power supply and connection to the GPU.

Figure 36: LiDAR mounted to Spot (GPU not shown but is typically mounted towards the back of Spot).

4.2. Mapping Overview

The major computational steps of the mapping algorithm can be viewed in Figure 37.

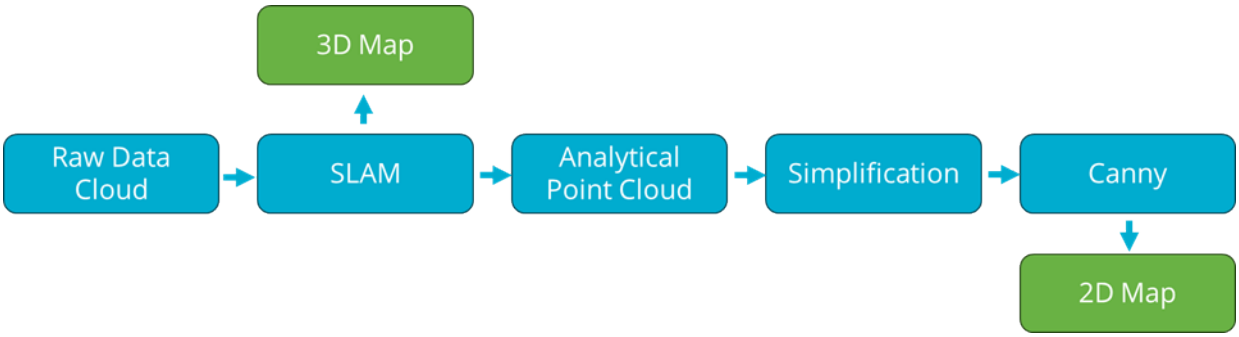


Figure 37: Major computational steps in the creation of 2D and 3D maps.

The derived mapping products presented in the remainder of the section were produced from a single LiDAR collect from the interior of AutonomyNM. Figure 38 should serve as a reference point when viewing these products.

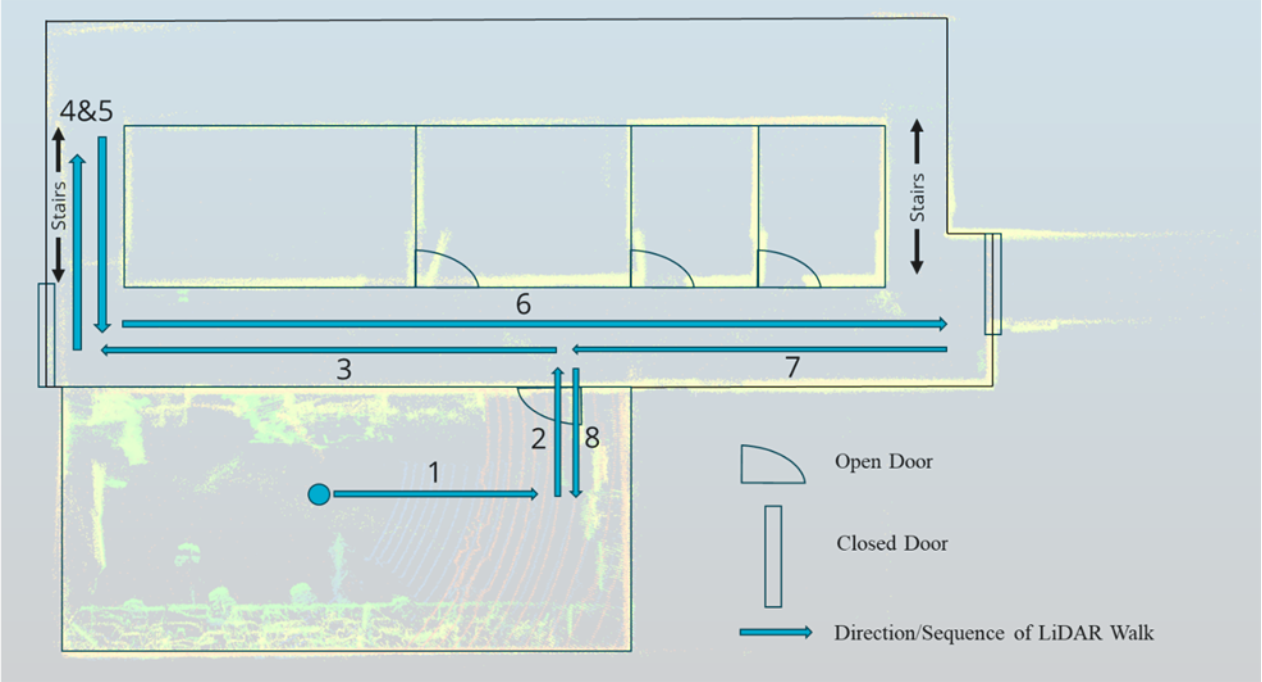


Figure 38: Layout and walk path of the test LiDAR collect.

All mapping for the Inspecta app begins with a point cloud. In operation this will be done when Spot navigates through a facility; after navigation both the 3D and 2D products are generated. During collection each pose (movement of the LiDAR sensor) generates a single point cloud. These single point clouds are packaged by the sensor and delivered to the Inspecta mapping software which further processes the collection of point clouds into 2D and 3D mapping products.

4.2.1. 3D maps

For 3D maps the collection of point clouds are extracted and combined. This is done using Ouster's SDK implementation of the SLAM algorithm. SLAM allows the sensor or sensor derived data to *understand* where in space the sensor is located relative to other poses and objects. Ouster's implementation of SLAM uses a backend of KISS-ICP which is a lightweight limited parameter odometry estimation method [14]. This allows the algorithm to function with a wide range of environments and with limited to no LiDAR parameterization before a data collect. The selection of Ouster's SLAM algorithm was made in large part because of the robustness of the algorithm's implementation and its ability to function across diverse complex environments. Without SLAM the 3D maps produced for Inspecta would look like Figure 39.

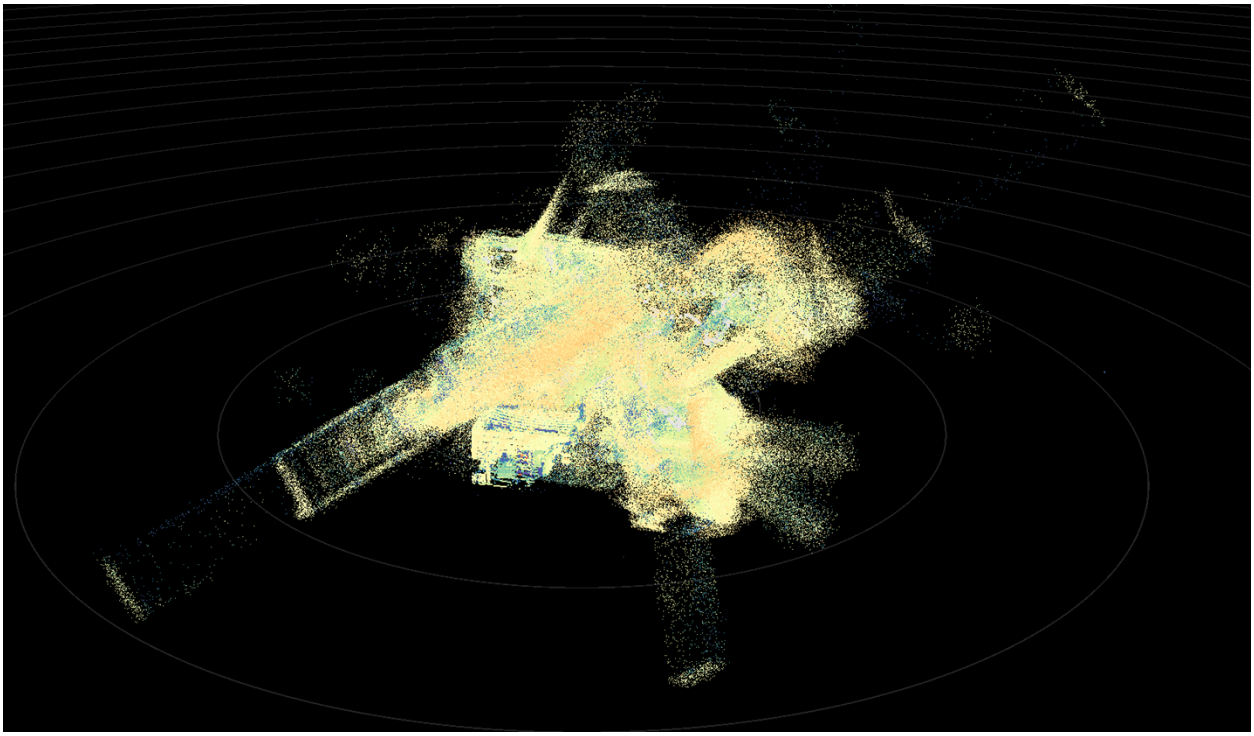


Figure 39: Depiction of LiDAR data without the SLAM algorithm.

With SLAM the software can understand the meaning and position of the original point clouds and combine them into a cohesive final point cloud (Figure 40).

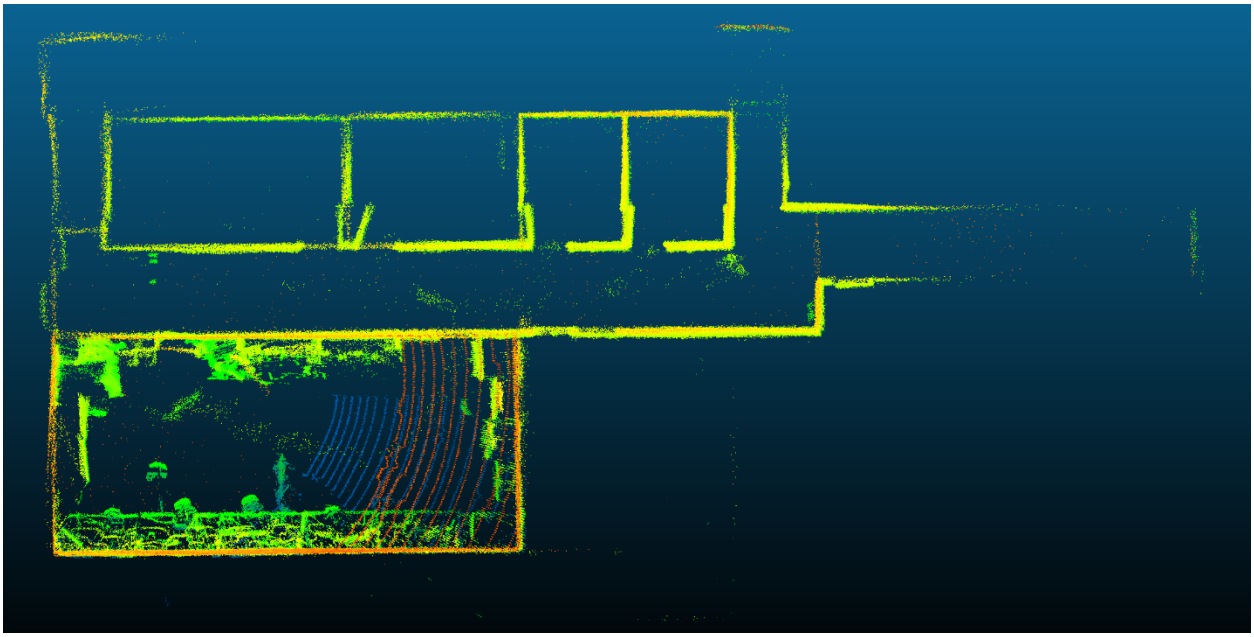


Figure 40: Depiction of LiDAR data with the SLAM algorithm.

As of the writing of this report point cloud processing is still mainly a manual process requiring a significant amount of analyst time and computation. Automation of point cloud creation and cleaning is an ongoing field of research and while potential solutions do exist, they are often brittle, having only been applied in a laboratory setting and are still actively under development. These solutions even if they could have been successfully integrated into Inspecta's mapping module would not have met the diverse and robust needs of IAEA inspectors.

4.2.2. 2D maps

A CV approach was selected to generate Inspecta's 2D maps. At its core, an unlabeled point cloud, like the ones generated by the OS1, consist of three primary features x, y, and z[34]. These features denote a point's position within the point cloud. Transitioning from three dimensions to two results in the subtraction of the z value and a primary focus on the x, and y values. With this in mind the team settled on using CV and image processing techniques to extract 2D floor plans from the processed point clouds. These CV techniques have a vetted technical basis and have seen wide adoption in multiple domains; performance issues and benefits are well known and documented. Figure 41 visualizes the current workflow of the 2D maps.

Figure 41: Major components of 2D map generation.

The team identified research challenges associated with this workflow that will require future study to fully realize the 2D mapping functions. These challenges are listed in section 4.3.

The example walk used in this report was short (3 minutes and 53 seconds) and generated a point cloud of over a million points. This includes the elimination of outliers and noise reduction during the 3D mapping process. To process the point cloud directly would take an estimated 21 terabytes of RAM[35]. Given the computation requirements, the 2D mapping function first simplifies the point cloud (Figure 42).

[34] LiDAR sensors detect other features such as signal return strength. These vary by sensor and are generally treated as metadata.

[35] While the integration of Spot's onboard GPU would lower this computational load it would be too computationally expensive to be considered for permanent integration.

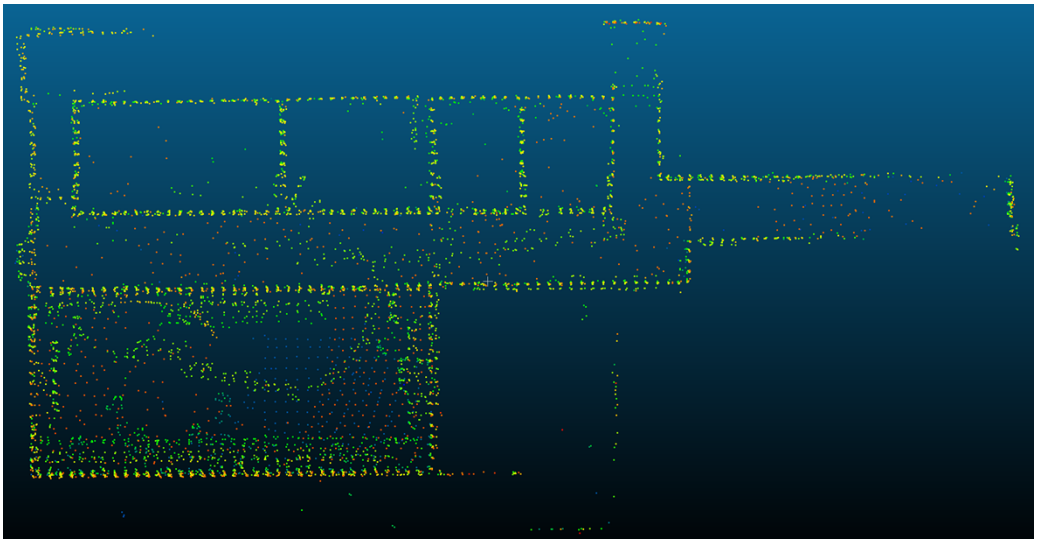


Figure 42: Voxel reduced point cloud.

This is done through the manipulation of voxel positions and filtering techniques. From here an image is created based on the dimensions of the original point cloud. This empty image is then filled with the data from the simplified point cloud in Figure 42. This results in an image like Figure 43.[36]

Figure 43: 2D input image used for edge detection.

With the input image created, the Canny algorithm is used to establish edges (walls) within the image. The Canny algorithm has three primary steps [15].

1. Noise reduction – the further removal of potential noise present in the image in Figure 43.
2. Identification of the intensity gradient – identifying the pixel direction that will eventually compose a line.
3. Removal of non-edge pixels – removal of those pixels that are not associated with an edge.

Based on the sigma (sensitivity metric) selected, the Canny algorithm produces images like those in Figure 44.

[36] This is a lower resolution image than the team's target resolution. The current resolution challenges are due to resource limitations.

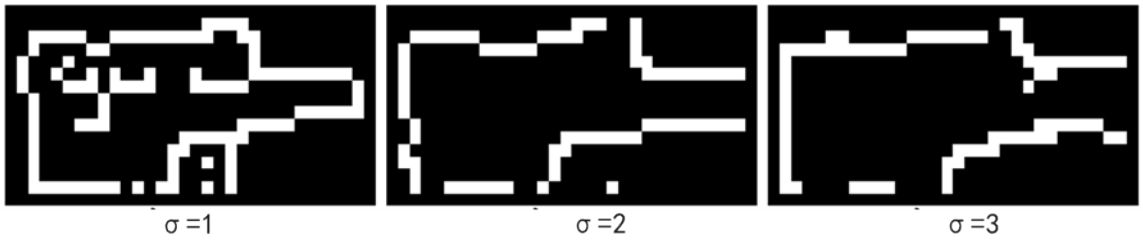


Figure 44: 2D Canny maps with different sigma values.

4.3. Identified Challenges and Next Steps

4.3.1. Traditional methods

The team experimented with a traditional approach to constructing environments from point clouds. This involves the creation of a mesh based on voxel position and establishing relevant vertexes. This process is manual, time and computationally expensive and prone to trial and error. If successful, these meshes can be used to extract floor plans. When tested with the OS1, the meshing approach produced the mesh found in Figure 45.

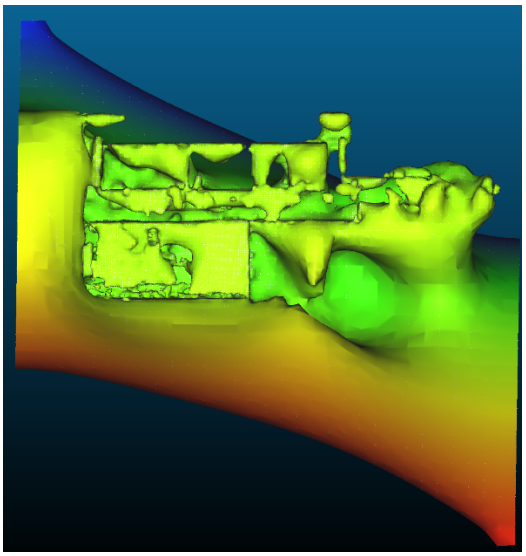


Figure 45: Mesh generated from the OS1 LiDAR walk.

Even if this approach had been successful, it would have been a significant challenge to automate this approach for repeatable applicability with diverse environments. Notably the workflow from point cloud to a 2D floor plan image is not common. Typically point clouds remain in their native state or are used for 3D environment creation. The software AutoCAD does have a capability to extract 2D images from 3D point clouds; however, this is a manual proprietary process and relies heavily on user input and manipulation.

4.3.2. Point cloud fidelity

Point cloud fidelity will be critical in the successful creation of 2D maps. The OS1 does have the hardware capacity to produce these point clouds; however, the exact limitations, if any, need to be determined through multiple LiDAR walks.

4.3.3. Pre-processing

A key component to an increase in 2D map fidelity will be the refinement of the point cloud pre-processing. This will require a refinement of the 3D processing functions, potentially adding a clustering method to pre-divide and then recombine walls and floors of the point cloud with a method similar to random sample consensus. Additionally, a gradient implementation of the conversion from point cloud to an image needs to be implemented. The current version of the mapping module notes a pixel as either empty or full. For the Canny algorithm to be fully impactful, a gradient for pixel direction needs to be established. This gradient could be created by integrating the z values into the pre-processing analysis. While z values are used now for initial point cloud cleaning and filtering, they should be further used to refine the creation of the simplified point cloud used in the Canny function. This would also allow for future identification of different floors and the creation of multiple refined 2D maps.

5.

6. FUTURE WORK

The Inspecta tool is intended to be modular and mission-centric, with skills developed over time to perform various inspection tasks. During this project, we have developed the architecture with this concept in mind and have implemented the following skills to assist the seal examination task: speech synthesis and recognition, wake word, seal/container OCR, document OCR, information recall, robotic autonomy, map acquisition, and computer vision. These skills showcase capabilities but have not been optimized for performance. Each of these skills can be further developed to ensure more robust performance, and during early FY25 the development team will determine which existing skills to focus on for improvement, especially in terms of preparing the Inspecta system for additional skills. In particular, the team may select seal/container OCR improvements due to varying lighting conditions and clarity of seal/containers numbers (due to degradation over time from environmental exposure and routine handling), fine-tuning computer vision (object detection of 55-gallon drums, metal cup seals, and common objects that might appear in nuclear facilities), pose estimation and 3D vision, information recall (lack of nuclear language and context meant that information recall performance has been limited), robotic autonomy including SLAM and search upgrades, robotic arm manipulation to interact with seals, and continued integration between the Inspecta app on the phone and Spot.

New skills may include object detection and OCR of the IAEA's new Field Verifiable Passive Loop Seal (FVPS)[37] and container/container numbers, robotic arm manipulation such that Spot can grasp and image seal numbers for OCR, dropping waypoints on the 2D map within Inspecta such that inspectors can identify significant locations or inspection activities, implementing the ability for Spot to follow an inspector using a fiducial on the inspector's outer garment, ensuring that related skills developed by other national laboratories have a path for integration with the Inspecta system (container counting has been implemented elsewhere and could be a valuable capability for Inspecta), and laying the framework for CZT gamma measurements of UF6 cylinders with Spot. This final task may require development of path planning within the Spot autonomy stack - essentially developing the capability for Spot to locate UF6 cylinders, plan a path for measurements, keep track of precise location of each measurement and traverse the path taking CZT measurements.

The team had considered implementing spent fuel pool verification, but the IAEA is proceeding with floating Cerenkov viewing devices and thus we may focus on dry spent fuel storage tasks and UF6 cylinder measurements instead.

[37] At the 65th INMM Annual Meeting during July 2024, an IAEA presentation discussed three use cases of interest for quadruped robots – seal examination (which we are developing through Inspecta/Spot currently), reading container numbers and counting containers in storage (partially addressed by LBNL through their NA-22 container counting project), and acquiring gamma measurements using CZT detectors on UF6 cylinders in enrichment facilities.

7. REFERENCES

[1] IAEA Safeguards, "Emerging Technologies Workshop: Trends and Implications for Safeguards Workshop Report," IAEA, Vienna, 2017.

[2] IAEA Safeguards, "Emerging Technologies Workshop: Insights and Actionable Ideas for Key Safeguards Challenges Workshop Report," IAEA, Vienna, 2020.

[3] IAEA Safeguards, "Research and Development Plan: Enhancing Capabilities for Nuclear Verification STR-385," IAEA, Vienna, 2018.

[4] IAEA, "IAEA Safeguards Development and Implementation Support Programme for Nuclear Verification 2020-2021," IAEA, Vienna, 2020.

[5] A. Mutluer, "Robotics Challenge Winning Design Helps Speed up Spent Fuel Verification," 18 March 2019. [Online]. Available: https://www.iaea.org/newscenter/news/robotics-challenge-winning-design-helps-speed-up-spent-fuel-verification. [Accessed 5 August 2022].

[6] A. Enders, "Safeguarding the Future: IAEA Looks for Improved Solutions for Passive Loop Seals for Nuclear Verification," 1 July 2020. [Online]. Available: https://www.iaea.org/newscenter/news/safeguarding-the-future-iaea-looks-for-improved-solutions-for-passive-loop-seals-for-nuclear-verification. [Accessed 1 September 2022].

[7] K. Aymanns and A. Reznicnek, "Use of passive sealing systems for casks in Spent Fuel Storage Facilities in Germany," Forschungszentrum Julich GmbH, Julich, 2016.

[8] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. Mahoney and K. Keutzer, "A survey of quantization methods for efficient neural network inference," arXiv preprint arXiv:2103.13630, 2021.

[9] Y. Ren, C. Hu, X. Tan, Q. S. Z. Tao, Z. Zhou and L. Tie-Yan, "Fastspeech 2: Fast and high-quality end-to-end text to speech," arXiv preprint arXiv:2006.04558, 2020.

[10] W. Ping, K. Peng, A. Gibiansky, S. O. Arik, A. Kannan, S. Narang, J. Raiman and J. Miller, "Deep voice 3: Scaling text-to-speech with convolutional sequence learning," arXiv preprint arXiv:1710.07654, 2017.

[11] S. Schneider, A. Baevski, R. Collobert and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," arXiv preprint arXiv:1904.05862, 2019.

[12] International Atomic Energy Agency, "IAEA Safeguards Glossary," International Atomic Energy Agency, Vienna, 2022.

[13] V. Sanh, *DistilBERT, A Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter,* arXiv preprint arXiv:1910.01108, 2019.

[14] P. Bajaj, D. Campos, N. Craswell, L. Deng, J. Gao, X. Liu, R. Majumder, A. McNamara, B. Mitra, T. Nguyen and others, *Ms marco: A human generated machine reading comprehension dataset,* arXiv preprint arXiv:1611.09268, 2016.

[15] I. Beltagy, K. Lo and A. Cohan, *SciBERT: A pretrained language model for scientific text,* arXiv preprint arXiv:1903.10676, 2019.

[16] P. Rajpurkar, R. Jia and P. Liang, *Know what you don't know: Unanswerable questions for SQuAD,* arXiv preprint arXiv:1806.03822, 2018.

[17] Boston Dynamics, "Panel Discussion: Spot in Nuclear Environments," [Online]. Available: https://www.bostondynamics.com/spot/resources/panel-spot-nuclear-environments?utm_source=ctabanner&utm_medium=inspection&utm_campaign=fy22q2-nuclear-panel. [Accessed 23 August 2021].

[18] B. Bonn. [Online]. Available: https://blog.bostondynamics.com/spot-collects-data-senses-radiation-in-nuclear-environments. [Accessed 23 August 2021].

[19] I. Vizzo, G. Tiziano, B. Mersch, L. Wisemann, J. Behley and C. Stachniss, "Kiss-icp: In defense of point-to-point icp–simple, accurate, and robust registration if done the right way.," *IEEE Robotics and Automation Letter,* vol. 8, no. 2, pp. 1029-1036, 2023.

[20] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence,* vol. 6, pp. 679-698, 1986.

[21] "Basics of IAEA Safeguards," 17 March 2021. [Online]. Available: https://www.iaea.org/topics/basics-of-iaea-safeguards.

[22] V. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," Soviet Physics Doklady, 1966.

DISTRIBUTION

Email—Internal

Name	Org.	Sandia Email Address
Technical Library	1911	sanddocs@sandia.gov

Sandia National Laboratories

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.