Towards Autonomous Caretakers via Procedure-based Automation

Michael W. Lanighan Ana Huamán Quispe Seth Gee David Kortenkamp

Abstract— **To support future uncrewed space habitats, caretaker robots will need to perform maintenance by following procedures similar to those used by human crews. This work details the integration of the PRIDE electronic procedure platform with skill representations called Affordance Templates (ATs) by treating ATs as procedural subroutines. We introduce a single, generic System Representation (SysRep) to encapsulate all AT commands and telemetry, along with new automation (PAX) plugins to bridge the Java-based PRIDE system with the ROS 2-based AT framework. This integrated system is demonstrated in a simulated Intra-Vehicular Activity (IVA) environment based on the NASA iMETRO facility, where a UR10 robot arm successfully follows a human-readable PRIDE procedure to autonomously complete a multi-step Carbon Dioxide Removal Assembly (CDRA) filter replacement task, validating this approach for integrating robotics into standard mission operations. A second demonstration is conducted using an Extra-Vehicular Activity (EVA) robot performing an inspection task with the same SysRep, validating the general applicability of the approach.**

I. Introduction

Robotic maintenance and repair of space habitats will play an increasingly important role in mission operations. NASA envisions space habitats such as Gateway and Lunar bases to be uncrewed or dormant for lengthy periods of time. Caretaker robots will need to handle both routine maintenance and preparatory activities prior to crew arrival [1]. Current crew flight operations, such as performing maintenance or repair activities, are performed by following carefully created procedures to ensure safe operations [2]. Ideally, caretaker robots would follow the same or similar procedures for their maintenance and repair tasks. This would make it easier for ground control personnel to task robots and track their progress. Such procedures should be human-readable and allow for interleaving ground control and robot actions. In [3], the desire for robot caretakers to integrate with procedure systems is explicitly stated:

> The concept of operation for a robotic system would start with procedures and information about the task being supplied to the robotic assets required. The robotic assets would then execute the procedures, asking for support, clarification, or confirmation when required. (p. 69)

In this paper, we discuss the integration of an automation-capable electronic procedure platform with adjustably autonomous robot action templates. The electronic procedure platform is called PRIDE and allows for mixed human and automated procedure execution [4], which is further described in Section II-A. The robot action templates are called Affordance Templates (ATs) and provide reusable, platform-agnostic task encodings [5], [6] and are further described in Section II-B. We demonstrate this integration using a caretaker space habitat robot based on NASA Johnson Space Center's iMETRO robotic evaluation platform [7] and an EVA robot based on the Canadarm3 being developed for Gateway.

Authors are with TRACLabs Inc. 1331 Gemini Street; Houston, TX 77058. Corresponding Author: mlanighan@tracblabs.com. This work was supported by NASA contract 80NSSC24CA028.

II. Related Work

A. Electronic Procedures

NASA has transitioned space operations to electronic procedure systems including the International Procedure Viewer (IPV) for ISS operations [8]. Commercial space companies are also pioneering electronic procedure systems for their operations [9]. For Artemis, NASA has a vision of integrating procedures with telemetry and commanding systems[1]. Towards that purpose, TRACLabs developed an electronic procedure system called PRIDE that includes procedure automation capabilities [4], [10]. The PRIDE system consists of an authoring tool in which users can build procedures [11]. The authoring tool loads a list of system commands and sensor data (called a System Representation or SysRep) that can be referenced in the procedure. The SysRep can be automatically generated from ATs, as described later in this paper. The PRIDE system also includes a web-based user display of the procedure with embedded sensor values and command buttons that allow a user to manually verify system status and command system actions. Finally, PRIDE has an automation plugin called the Procedure Agent for eXecution (PAX) that can take any procedure, automate verifying sensors, and send commands automatically, pausing when human input is needed. All procedure actions, whether by a user or by PAX, and all data entered into the procedure are logged to a database. This logged data includes the user's id (or PAX) and a timestamp. This allows for both real-time monitoring and for after-action reporting and auditing of any procedure execution.

B. Affordance Templates

Due to bandwidth and latency limitations, command and control of remote assets become increasingly difficult as distance increases. One approach to mitigate this is by leveraging semi-autonomous operation paradigms, where rather than directly controlling the remote asset, operators

[1]https://www.nasa.gov/wp-content/uploads/2023/02/orion-reference-guide-111022.pdf (page 20)

command tasks that are then autonomously executed by the asset. One approach to enable such semi-autonomy is ATs. Originally developed to enable semi-autonomous operation of humanoids in the DRC [5], they have since been adopted and further developed by the community to enhance semi-autonomous robot capabilities [6], [12], [13]. ATs define keyframe trajectories relative to virtual objects that, when combined with motion planning tools or controllers, realize a defined task relative to the modeled object. At runtime, virtual overlays of the target objects and keyframes can be adjusted either by operators or through autonomous registration to match the observed runtime environment.

C. *Controlling Space Robots via Procedures*

Several research groups have investigated combinations of procedures and autonomous tools to control robot assets [14], [15], [16]. Previous work by TRACLabs has investigated combining the automated procedure system of PRIDE and the autonomous robot capabilities provided by ATs. Initial integration relied on manually creating specific PRIDE SysReps and corresponding Java interface code for every specific AT to enable their use in PRIDE [17]. Follow-on work extended PRIDE's automation capabilities to consume AT JSON files and automatically create the necessary PRIDE SysReps [18] during procedure authoring. While such automation lowers developer burden, it can result in brittle interfaces that must be updated/regenerated if ATs are updated and leads to an explosion of SysReps that ultimately need to be tracked and maintained. To address that issue, this work aims to simplify the interface between PRIDE and ATs by providing a single, parametrizable interface to ATs by treating ATs as procedural subroutines.

III. Affordance Templates as Procedural Subroutines

To facilitate the creation of procedures to command and monitor a robot task composed of actions encoded as AT, we implemented two new resources that take advantage of the generalizable nature of the AT library.

A. *Generic AT SysRep*

Independent of the task or the robot involved in a specific mission, task execution using ATs involves the same sequence of steps: (1) Load the AT containing the desired trajectory to be executed; (2) Register the template's virtual objects with objects of interest in the environment; (3) Verify that the action is realizable by requesting the task to be planned; and (4) Execute the desired AT trajectory. Noting this, we created a **generic AT SysRep**, shown in Fig. 4, that encapsulates the standard command and telemetry necessary for an operator to command robots to complete tasks defined using the AT description language.

1) Commands: The top portion of Fig. 4 shows the `commands` section of the AT SysRep. The commands exposed through the SysRep allow PRIDE Procedures to control AT task execution. Adding new instances of templates is achieved through `add_affordance_template`.


```yaml
id: affordance_templates
name: CRAFTSMAN AffordanceTemplates YAML

commands:

- [add_affordance_template, Add Affordance Template,
  [[affordance_template, string],[hide_waypoints, boolean]]]

- [delete_affordance_template, Delete Affordance Template,
  [[affordance_template, string], [id, integer]]]

- [plan_trajectory, Plan Trajectory,
  [[affordance_template, string], [trajectory, string]]]

- [execute_plan, Execute Plan,
  [[affordance_template, string], [trajectory, string]]]

- [set_waypoint_pose, Set Waypoint Pose,
  [[affordance_template, string], [id, integer],
  [trajectory, string], [waypoint_id, integer],
  [ee_name, string], [x, real], [y, real], [z, real],
  [roll, real], [pitch, real], [yaw, real],
  [frame_id, string]]]

- [set_display_object_pose, Set Display Object Pose,
  [[affordance_template, string], [id, integer],
  [display_object, string], [frame, string],
  [x, real], [y, real], [z, real],
  [roll, real], [pitch, real], [yaw, real]]]

telemetry:

- [robot_active, Robot Active, boolean]

- [planner_node_active, Planner Node Active, boolean]

- [affordance_template_server_active,
  Affordance Template Server Active, boolean]

- [execute_status, Execution Status, string]

- [plan_status, Plan Status, string]

- [plan_valid, Plan Valid, boolean]
```


Fig. 1. Generic AT SysRep.

Multiple ATs and/or potentially multiple instances of the same AT can be interfaced with concurrently. Registering AT objects to sensed or known object locations can be achieved via `set_display_object_pose`, which allows operators to update AT positions to match the runtime context. This capability is especially useful for reusability purposes, as it allows an operator to use the same AT (e.g., an AT that encodes a pick and place skill) for the same task under different scenarios (e.g., picking up an air filter from a rack and placing it at its installation point vs. picking it up from the latter and disposing of it). Planning motions encoded in AT trajectories is enabled through `plan_trajectory`, executing planned motions through `execute_plan` and unloading (deleting) an AT once the task has been achieved `delete_affordance_template`.

In addition to these basic commands, our AT SysRep further provides commands that allow the operator to modify a loaded AT by updating relevant task characteristics by modifying the task keypoints, `set_waypoint_pose`. This can allow additional supporting applications, such as grasp generation tools [19], to update AT waypoints to match the specific runtime context.

2) Telemetry: The lower portion of Fig. 4 shows the telemetry definitions for the SysRep. The information re-

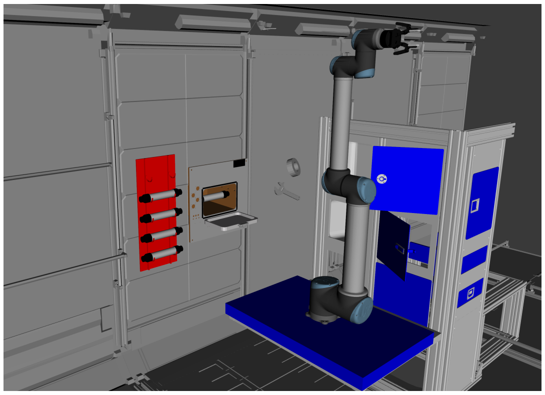

Fig. 2. A view of the IVA trainer setup used for the IVA CDRA filter replacement demo.

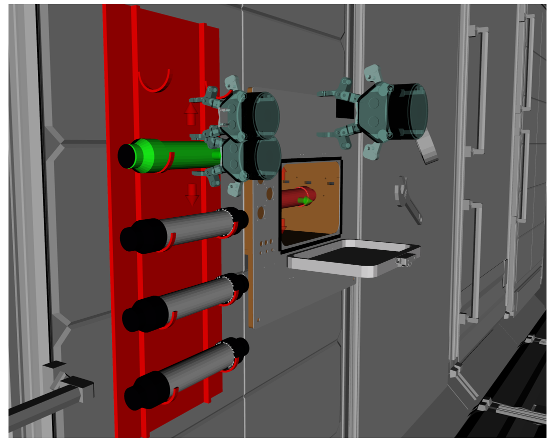

Fig. 3. The CDRA filter AT aligned with a CDRA filter on the rack.

turned informs the operator whether an AT has been successfully planned, `plan_status` and `plan_valid`, whether the execution finished correctly, `execute_status`, as well as data that describes the robot status in general, such as if the robot is actively in motion, `robot_active`, and whether the planning and control system is operative, `affordance_template_server_active` and `planner_node_active`.

B. PAX plugin

To support the generic AT SysRep, two PAX plugins were implemented, one for commanding and one for telemetry processing. PAX plugins on the PRIDE side must be implemented in Java, whereas the AT process on the robot side operates within the ROS 2 framework, a robotics middleware based in C++/Python with services and topics as the main means of communication. To bridge these systems, we leveraged the *rosbridge* suite [20], which provides a JSON interface to ROS 2 and allows any client to exchange information with ROS 2 processes. Specifically, we implemented the AT PAX classes using the *jrosbridge* library[2], which provides an interface to rosbridge in native Java.

IV. DEMONSTRATION

A. IVA Caretaker Robot Simulation

The IVA caretaker demonstration environment is designed to simulate a CDRA filter replacement using a UR10 arm equipped with a Robotiq two-fingered gripper. The simulation environment, shown in Fig. 2, leverages the physics simulator Gazebo and consists of a Gateway module containing a UR10 arm mounted on a table, several CDRA filters on a storage rack, and a CDRA filter system cabinet on one of the station walls. The simulation leverages 3D models from the open-source NASA JSC iMETRO facility[3][7].

The simulated system is visualized in RViz2. While the simulation can be directly visualized in Gazebo, in the demonstration, Gazebo is run headless to emulate what a remote operator would experience. Running headless also helps reduce resource requirements when performing the demonstration. In order to visualize the robot and CDRA filter models correctly in RViz2, the simulated telemetry (robot states) generated by Gazebo is bridged to the ROS 2 system via topics. Once the topics are bridged, RViz2 is able to subscribe to these model robot state topics, which can be viewed by the user. In a physical system, such telemetry would be provided by the hardware instead of the simulator.

Interacting with the IVA robot is done through PRIDE, which interfaces with ATs that define behaviors relative to the CDRA filter, the CDRA filter system cabinet, and the CDRA filter rack. The AT display objects shown in RViz use the same model that is loaded in Gazebo and match the size and shape of the simulated models. This display object is overlaid onto the simulated model location, registering the encoded skill relative to the simulated object. This registration allows the robot to understand where the objects to grasp are located within the environment, enabling the system to generate collision-free motions to obtain and transport the filters. Fig. 3 shows the AT in RViz2. The CDRA filter object is shown in green, overlaid on the simulated CDRA filter. The motion to pick the filter from the rack is represented by a sequence of end-effector goals, shown in teal. This AT is leveraged by a PRIDE procedure during the IVA CDRA filter demonstration to acquire and change out the filter.

B. Procedure Authoring

The PRIDE authoring tool loads a SysRep containing ATs and makes them available to the procedure author. Figure 4 shows the SysRep loaded into Pride Author. When the author wants to deploy an AT in a procedure, they simply drag-and-drop the AT from the SysRep onto the procedure canvas, as shown in Figure 5. There are basic actions to add an AT, plan a trajectory based on an AT, and then execute that plan. The parameters of each command can be hard-coded or can use input provided by an operator.

C. Demonstration Scenario

Two demonstration scenarios were developed – one using an IVA manipulator and another using an EVA manipulator.

[2]https://github.com/rctoris/jrosbridge
[3]https://github.com/NASA-JSC-Robotics/iMETRO

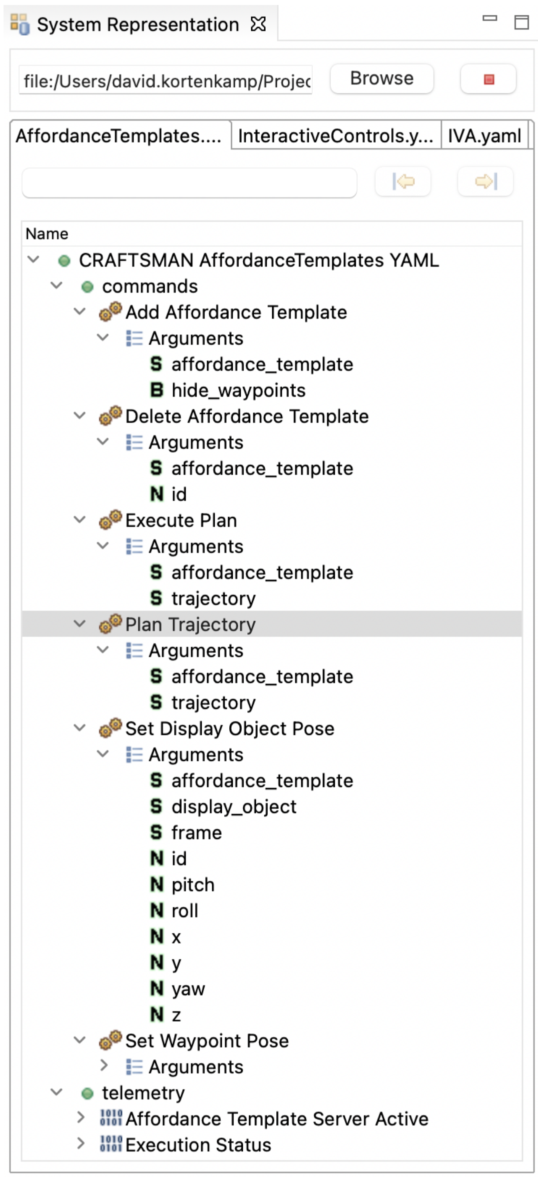

Fig. 4. A SysRep containing ATs loaded into the PRIDE authoring tool.

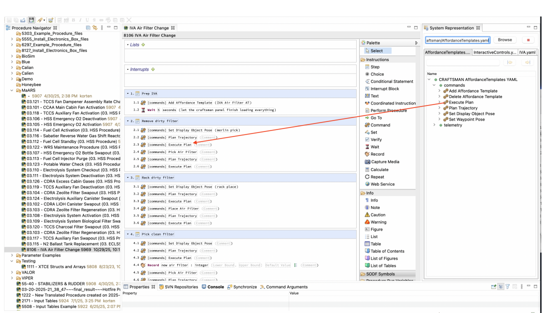

Fig. 5. Dropping an AT instruction into a procedure.

Fig. 6. IVA CDRA filter replacement procedure viewed in PRIDE.

Both use the same commanding infrastructure described in the previous section. The ability to command two different robots using the same AT mechanism validates the general applicability of our approach.

1) IVA Demonstration: The IVA demonstration scenario replaces a dirty or spent CDRA filter in the Carbon Dioxide Removal Assembly (CDRA) with a clean one. The procedure, shown in Fig. 6, consists of steps that include removing the CDRA filter from the system cabinet, placing the dirty CDRA filter on a CDRA filter rack, picking a clean CDRA filter from the same CDRA filter rack, and placing the clean CDRA filter into the CDRA filter system cabinet.

The procedure follows a basic set of steps:

1) Prepare the AT and position the robot
2) Remove the dirty filter from the CDRA filter system
3) Place the dirty filter on the CDRA filter rack
4) Pick a clean filter from the CDRA filter rack
5) Insert the clean filter into the CDRA filter system
6) Complete the task and await a new procedure

Step 1 consists of adding the CDRA filter AT to the RViz2 environment and positioning the robot arm from straight up into a position that is ready to grasp a CDRA filter. To remove a dirty CDRA filter from the filter system cabinet in step 2, the AT is aligned with the CDRA filter currently in the filter system cabinet, and a trajectory plan is generated to grasp the filter (Fig. 7). The procedure then unlocks the dirty filter from the cabinet before the robot can remove the dirty filter from the cabinet. Once removed, step 3 positions the AT onto an empty slot on the CDRA filter rack and plans a trajectory to place the dirty CDRA filter onto an empty slot.

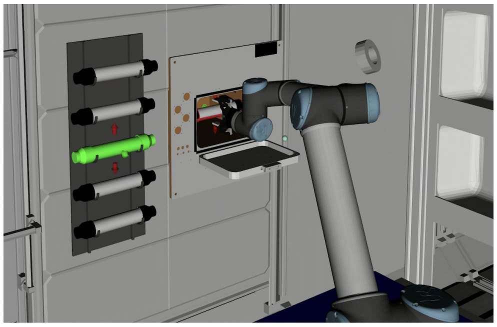

Fig. 7. Removing a spent CDRA filter from the system cabinet.

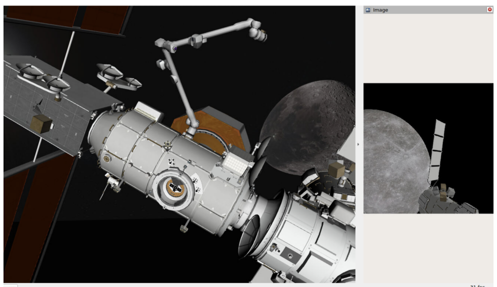

Fig. 8. View of the EVA robotic manipulator.

Then, step 4 selects and positions the AT on a clean CDRA filter on the rack (Fig. 3). To complete the filter exchange, step 5 moves the AT to the desired location within the filter system cabinet to insert the clean CDRA filter. Once inserted, the filter is locked in place, and step 6 moves the robot arm back into a ready position and removes the AT from RViz2.

2) EVA Demonstration: The EVA robot is a version of the Canadarm3 being designed for Gateway[4]. Figure 8 shows the EVA manipulator and a camera view from the end effector. A simple EVA procedure (shown in Figure 9) was created using the exact same AT SysRep as the IVA procedure – the one from Figure 4. The procedure directs the Canadarm3 to inspect various external locations around the Gateway modules for leaks or impacts. This demonstration shows the generality of our approach with a single SysRep able to directly control multiple robots using different ATs. As more ATs are created for a wide variety of robots, the same SysRep can be used to write procedures that control all of them. Also, if the ATs are updated or changed, the procedures that reference them do not need to change.

V. User Interaction

PRIDE supports a mixed-initiative approach to robot autonomy. The procedure shown in Figure 6 can be run in either fully autonomous mode, or a user can manually click each "Send" button to initiate robot actions. In the former case, commands are sent by PAX to the robot, monitored for successful completion, and then the next command is executed. While not shown in this particular procedure, PRIDE supports conditional branching based on data returned from a command, system telemetry, or user input.

[4]https://www.asc-csa.gc.ca/eng/canadarm3/about.asp

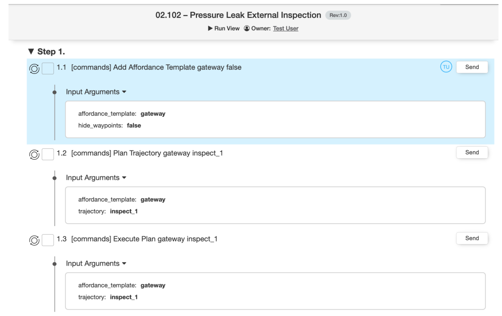

Fig. 9. A simple procedure for commanding the EVA manipulator.

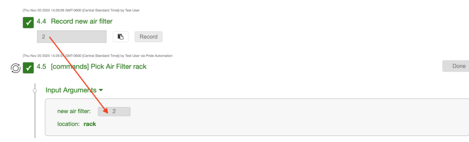

Fig. 10. Data entered by the user.

PRIDE also supports the interleaving of automation and manual execution. For example, a user can be asked to input data that is then used by the robot. An example is shown in Figure 10 where a filter id has been entered by a user and then passed to a robot command as a parameter. If the procedure is being run autonomously, PAX will stop at the Record Instruction and wait for the user to enter the data and complete the instruction. Then PAX will automatically resume with the next instruction. The circular arrows icon beside the instruction denotes an action that can be automated, while instructions (such as 4.4 in Figure 10) that must be completed by a user do not have that icon. As another example, Figure 11 shows a manual instruction (5.1) that must be completed by the user before automation can continue with the procedure. The PAX executive pauses at that instruction and waits for the user to complete it before proceeding. Users are also able to stop automation at any time, and they can add break points to pause automation at pre-defined locations in the procedure.

VI. Conclusion

Integrating robot caretakers with electronic procedures will open up new opportunities to incorporate robotics into traditional space habitat operations. By working from the same procedural artifacts as crew members and ground controllers, robots can be more easily included in routine mission plans. For example, we are currently working on using procedures

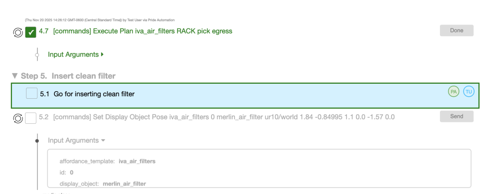

Fig. 11. Requesting permission from the user before continuing.

to integrate robotic agents with a prototype crew assistance agent called Daphne [21] that is being tested in NASA analog mission environments. Daphne simply needs to start the relevant procedure to perform robotic maintenance or repair operations without needing any understanding of the robot's control interface. Because procedures are the common "language" of human spaceflight operations, robots need to speak this language as well for their use in mission operations to be sustainable.

REFERENCES

[1] S. Fuller, E. Lehnhardt, C. Zaid, and K. Halloran, "Gateway program status and overview," *Journal of Space Safety Engineering*, vol. 9, no. 4, pp. 625–628, 2022.

[2] T. Hoppenbrouwers, L. Ferra, M. Markus, and M. Wolff, "Operations data files, driving force behind international space station operations," *Acta Astronautica*, vol. 138, pp. 255–261, 2017.

[3] J. Badger, D. Higbee, T. Kennedy, S. Vitalpur, M. Sargusingh, S. Shull, B. Othon, F. J. Davies, E. Hurlbert, N. Townsend, *et al.*, "Spacecraft dormancy autonomy analysis for a crewed martian mission, Tech. Rep. NASA/TM-2018-219965, 2018.

[4] D. Schreckenghost, S. Bell, D. Kortenkamp, and J. Kramer, "Procedure automation: Sharing work with users." in *AAAI Spring Symposia*, 2018.

[5] S. Hart, P. Dinh, and K. Hambuchen, "The affordance template ros package for robot task programming," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[6] S. Hart, A. H. Quispe, M. Lanighan, and S. Gee., "Generalized Affordance Templates for Mobile Manipulation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2022.

[7] N. Dunkelberger, E. Sheetz, C. Rainen, J. Graf, N. Hart, E. Zemler, and S. Azimi, "Design of the iMETRO facility: A platform for intravehicular space robotics research," in *2025 22nd International Conference on Ubiquitous Robots (UR)*. IEEE, 2025, pp. 390–397.

[8] M. Martignano, M. Wolff, U. Brauer, and P. Kiernan, "Software assisted authoring and viewing of ISS crew procedures," *Acta Astronautica*, vol. 61, no. 11-12, pp. 1053–1060, 2007.

[9] J. Gabbert, M. Devereaux, J. Owen, D. Kortenkamp, S. Bell, and G. Kbidy, "Integrated mission operation concepts for the dream chaser cargo system," in *Proceedings of the 17th International Conference on Space Operations (SpaceOps)*, 2023.

[10] S. Bell and D. Kortenkamp, "Embedding procedure assistance into mission control tools," in *Proceedings of the IJCAI Workshop on AI in Space*, 2011.

[11] M. Izygon, D. Kortenkamp, and A. Molin, "A procedure integrated development environment for future spacecraft and habitats," in *Proceedings of the Space Technology and Applications International Forum (STAIF 2008), available as American Institute of Physics Conference Proceedings Volume 969*, 2008.

[12] S. J. Jorgensen, M. W. Lanighan, S. S. Bertrand, A. Watson, J. S. Altemus, R. S. Askew, L. Bridgwater, B. Domingue, C. Kendrick, J. Lee, *et al.*, "Deploying the NASA Valkyrie humanoid for IED response: An initial approach and evaluation summary," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019.

[13] A. Pettinger, P. F. Cassidy Elliott, and M. Pryor, "Reducing the teleoperator's cognitive burden for complex contact tasks using affordance primitives," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 11 513–11 518.

[14] P. Schmaus, D. Leidner, T. Krüger, R. Bayer, B. Pleintinger, A. Schiele, and N. Y. Lii, "Knowledge driven orbit-to-ground teleoperation of a robot coworker," *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 143–150, 2019.

[15] M. Bualat, T. Fong, M. Allan, X. Bouyssounouse, T. Cohen, L. Fluckiger, R. Gogna, L. Kobayashi, G. Lee, S. Lee, *et al.*, "Surface telerobotics: development and testing of a crew controlled planetary rover system," in *AIAA SPACE 2013 conference and exposition*, 2013, p. 5475.

[16] T. Smith, M. Bualat, A. Akanni, O. Alexandrov, L. Barron, J. Benton, G. Chuang, B. Coltin, T. Fong, J. Garcia, *et al.*, "ISAAC: An integrated system for autonomous and adaptive caretaking," in *10th annual International Space Station Research and Development Conference (ISSRDC)*, 2021.

[17] S. Hart, J. Kramer, S. Gee, and R. R. Burridge, "The pharaoh procedure execution architecture for autonomous robots or collaborative human-robot teams," in *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2016, pp. 888–895.

[18] S. Hart, J. Kramer, S. Bell, D. Schreckenghost, S. Gee, A. Huamán Quispe, and R. R. Burridge, "PHARAOH: Remotely Operating Robotic Systems using Human-Readable Procedures," in *Proceedings of the 16th International Conference on Space Operations (SpaceOps-2021)*, 2021.

[19] A. Murali, B. Sundaralingam, Y.-W. Chao, W. Yuan, J. Yamada, M. Carlson, F. Ramos, S. Birchfield, D. Fox, and C. Eppner, "Graspgen: A diffusion-based framework for 6-dof grasping with on-generator training," *arXiv preprint arXiv:2507.13097*, 2025.

[20] R. Toris, J. Kammerl, D. V. Lu, J. Lee, O. C. Jenkins, S. Osentoski, M. Wills, and S. Chernova, "Robot web tools: Efficient messaging for cloud robotics," in *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2015, pp. 4530–4537.

[21] P. K. Josan, P. Dutta, R. Abbott, A. V. Martin, B. J. Dunbar, R. K. Wong, D. Selva, and A. Diaz-Artiles, "Virtual assistant for spacecraft anomaly resolution: Effects on human performance metrics," *Journal of Aerospace Information Systems*, vol. 22, no. 4, pp. 264–274, 2025.