ARL-TN-1336 ● Oct 2026

STAIT-Replay: Multimodal 3D Spatiotemporal Stream Replay for Enabling Common Grounding in Human–Agent Teams

by Russell Cohen Hoffing

DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

NOTICES

Disclaimers

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

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

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

ARL-TN-1336 ● OCT 2026





STAIT-Replay: Multimodal 3D Spatiotemporal Stream Replay for Enabling Common Grounding in Human–Agent Teams

Russell Cohen Hoffing
DEVCOM Army Research Laboratory

DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

REPORT DOCUMENTATION PAGE

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
		START DATE	END DATE
October 2026	Technical Note	6/20/2026	9/26/2026

4. TITLE AND SUBTITLE
STAIT-Replay: Multimodal 3D Spatiotemporal Stream Replay for Enabling Common Grounding in Human–Agent Teams

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

6. AUTHOR(S)
Russell Cohen Hoffing

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
DEVCOM Army Research Laboratory ATTN: FCDD-RLA-IC Playa Vista, CA 90094	ARL-TN-1336

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

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

13. SUPPLEMENTARY NOTES
ORCID: Russell Cohen Hoffing, 0000-0002-3478-8196

14. ABSTRACT
Researchers and analysts attempting to understand real-world, naturalistic 3D spatiotemporal datasets derived from humans, autonomous agents, and mixed human–agent teams contend with cognitive and technical bottlenecks due to massive spatially and temporally heterogeneous sensor streams. These datasets comprise high-frequency poses, LiDAR point clouds, 2D camera feeds, conversational dialogue, eye gaze vectors, and discrete interaction events. For instance, an analyst attempting to evaluate what an autonomous ground robot "sees" via LiDAR scans must inspect 3D point clouds in isolation—a process that frequently leaves object identities ambiguous. Similarly, analyzing dialogue transcripts without spatial context obscures the situational awareness motivating communication. Synchronously pairing 3D spatiotemporal data, 2D video feeds, pose, trajectories, and occupancy grids provide the critical visual and spatial grounding necessary to understand this otherwise obtuse data. We present `STAIT-Replay`, an open-source, interactive 3D spatiotemporal visualization and auditing tool developed as part of the broader STAIT (SpatioTemporal Analysis and Inference Toolkit) suite and built on top of the Rerun visualization framework. `STAIT-Replay` enables unified visualization and interaction with heterogeneous 3D spatial data, 2D visual feeds, dialogue, eye gaze rays, and discrete event streams on a unified timeline for interactive playback.

15. SUBJECT TERMS
Military Information Sciences, Humans in Complex Systems, Information Autonomy and Teaming Sciences Division, human–agent, human–machine, human–computer, interaction, spatiotemporal data, point cloud, STAIT, toolkit, toolbox, open source

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT	b. ABSTRACT	C. THIS PAGE		
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UU	20

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
Russell Cohen Hoffing	(520) 691-5011

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

Contents

List of Figures iv

List of Tables iv

1. Introduction 1

2. Features and Core Capabilities 2

3. System Architecture 3

4. Case Study: Resolving Ambiguous Dialogue and Clarification Acts in Human–Robot Teaming 6

5. Potential Operational Use Cases: Military Relevance and Technology Extensions 7

6. Conclusion 9

7. References 10

List of Symbols, Abbreviations, and Acronyms 13

Distribution List 14

List of Figures

Figure 1. Example dataset replay in native Rerun application window. 1) 3D camera pose view with camera frustum, camera path labeled with start and end, and occupancy grid in gray. 2) 2D camera video with 3) associated event table indicating time-stamped pictures sent to commander. 4) 2D overhead occupancy grid with marker showing robot heading. 5) Dialogue data with entity path linked to individual speakers and level indicating annotated dialogue data with the body showing the dialogue. 6) Selection view showing the other column data from the tabular data file (i.e., .xls) associated with that row's selected dialogue time stamp and entity. 7) Timeline showing current time and visual view of data samples over time. 3

Figure 2. Screenshot of SCOUT dataset with red rectangle denoting unclear motivation behind communications. 7

List of Tables

Table 1. Core code files and responsibility. 4

1. Introduction

Common grounding research, robotics, human–machine interaction, AI, and cognitive science research increasingly depend on naturalistic, unscripted datasets captured in complex physical environments. These span large-scale egocentric video (e.g., Ego4D[1]), dense 3D egocentric digital twins with gaze (e.g., HD-EPIC[2]), and situated human–robot collaborative teaming experiments featuring bidirectional spoken dialogue and robot telemetry (e.g., SCOUT[3,4]). Establishing common ground—the mutual knowledge, beliefs, and assumptions essential for coordinated action[5]—is particularly challenging in human–robot teams due to asymmetrical perception and spatial viewpoints and understandings. Existing research tools are typically optimized for a single data domain, creating friction during multimodal dataset auditing and analyzing human and agent behavior. Existing visualization tools most similar to STAIT-Replay are typically optimized for robotics telemetry, such as the open-source Rviz[6] and closed-source Foxglove Studio,[7] but do not easily support human-derived data streams and events for customized playback and auditing. Many other tools are data-specific: mobile eye-tracking software (e.g., Pupil Player[8] and Tobii Pro Lab[9]) specializes in gaze rays without situated 3D spatial environments; acoustic and linguistic annotation tools (e.g., Praat[10] and ELAN[11]) provide multi-tier transcription and 2D video alignment, but lack 3D spatial coordinate representation; static mesh tools like MeshLab[12] handle geometry without temporal playback; and computer vision annotation tools like CVAT[13] focus on bounding-box dataset labeling rather than interactive exploratory replay. Other platforms are tightly coupled to specific game engines, such as ARETT[14] (which requires compiling within the Unity 3D environment); or are no longer being actively developed, such as the Platform for Situated Intelligence (PSI).[15] The challenge of visualization software is to be able to easily accommodate heterogeneous data streams regardless of sampling rate.

The **SpatioTemporal Analysis and Inference Toolkit (STAIT)** provides a software pipeline for standardizing, processing, and analyzing multi-agent interaction streams, as well as annotating, refining, and exporting digital twin representations.[16] Within the STAIT ecosystem, `STAIT-Replay` serves as the primary visual exploration, diagnostic, and auditing tool and is built on top of the Rerun visualization framework.[17] STAIT-Replay extends STAIT toolkit capabilities to address the aforementioned challenges and limitations of existing software. STAIT-Replay is positioned to be used downstream of raw data ingestion or wrangling into the STAIT standard but does not require STAIT standardized data and can be used in parallel to spatial annotation tools (e.g., the STAIT-Mapper Refiner).

2. Features and Core Capabilities

The main feature of STAIT-Replay is the customizable and scalable simultaneous playback of multimodal 3D and 2D data streams, including 3D point clouds, 3D/2D pose and trajectories, 2D images or MP4 videos, occupancy grids, dialogue or text, and discrete events. Figure 1 shows an example screen capture from the SCOUT dataset,[3,4] which is displaying the 3D pose visualization including camera pose with the projected frustum at every moment as well as the entire camera path with the start and end labeled, and the 3D occupancy grid. The 2D camera video is also displayed alongside the event table containing when the robot sent camera frames to the commander. Each visible event time stamp can be clicked to jump to the corresponding time stamp and locks all other streams to this time (or closest timepoint). Similarly, the dialogue table is displayed where the entity is linked to each individual speaker. When selected, the right selection panel shows the rest of the values from the .xlsx file where the events were loaded from. The STAIT-replay tool can be run in a native Rerun desktop application, which allows for larger RAM limits and larger windowed resolution, but lacks the interactive features available in the Jupyter Notebook Viewer. The Jupyter notebook allows for 1) searching event files for keywords that generate time-stamped jump buttons to lock all streams to that timepoint, 2) jump to a pose associated time stamp, and 3) notetaking. Each of features allow for ease of auditing complex data streams to gain understanding and awareness of data.

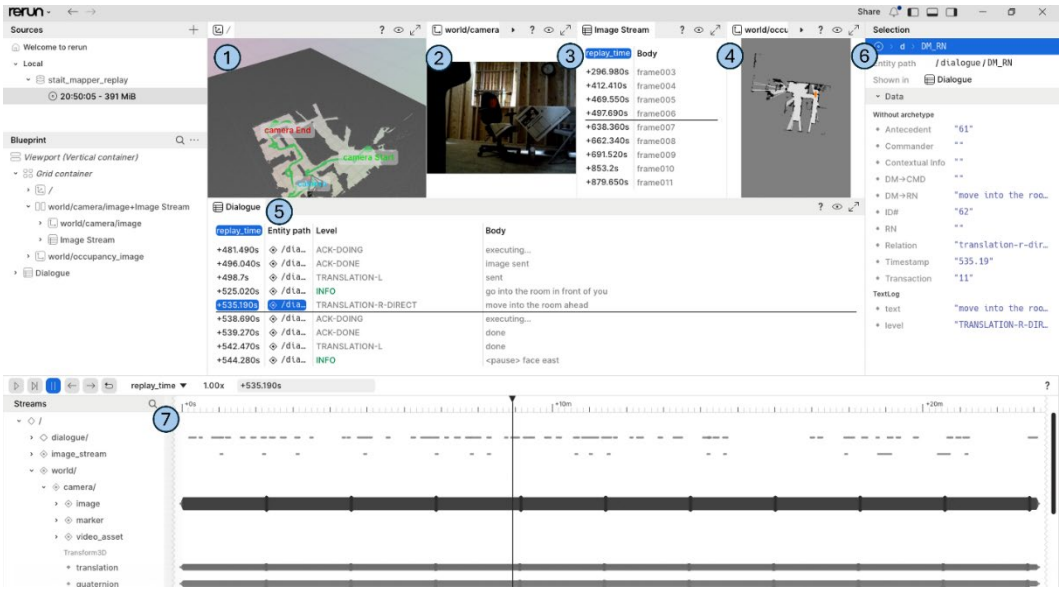


Figure 1. Example dataset replay in native Rerun application window. 1) 3D camera pose view with camera frustum, camera path labeled with start and end, and occupancy grid in gray. 2) 2D camera video with 3) associated event table indicating time-stamped pictures sent to commander. 4) 2D overhead occupancy grid with marker showing robot heading. 5) Dialogue data with entity path linked to individual speakers and level indicating annotated dialogue data with the body showing the dialogue. 6) Selection view showing the other column data from the tabular data file (i.e., .xls) associated with that row's selected dialogue time stamp and entity. 7) Timeline showing current time and visual view of data samples over time.

3. System Architecture*

`STAIT-Replay` follows an adapter-and-orchestrator architecture. Dataset-specific code is confined to a source adapter; however, synchronization, Rerun logging, layout generation, and notebook controls are shared by all datasets. The source adapter translates dataset files into a small set of modular Python records, and the rest of the package translates those records into one Rerun recording. Table 1 provides a guide to each file's responsibility.

* GPT-5.6 Terra was used to improve an initial draft of Section 3 and was subsequently heavily revised by the author. Additionally, this same AI tool was used to generate the initial draft of the references based on the text and each was checked and revised by the author. The code underpinning the STAIT-Replay module was developed with the assistance of Gemini Flash versions 3.6–3.8 and personally reviewed by the author to verify and audit code.

Table 1. Core code files and responsibility.

Path	Responsibility
`replay/base.py`	Defines the common data records—such as `Pose`, `ImageFrame`, `PointFrame`, `Gaze`, and `EventEntry`—and the abstract `ReplaySource` interface implemented by every dataset source adapter. This is the package's data contract.
`replay/registry.py`	Maps a short source name, such as `scout`, `hdepic`, or `r2c2`, to the corresponding `ReplaySource` class. It provides source lookup and construction without placing dataset-specific imports in the user-facing entry points.
`replay/sources/__init__.py`	Imports the built-in adapters and registers them with `registry.py`. Importing this module makes the source names available to the CLI and notebook application.
`replay/sources/`	Source adapter folder containing modular readers for adapting data files to Rerun formats.
`replay/annotations.py`	Implements the dataset-agnostic event-table loader for CSV, Excel, and Parquet files. It resolves time stamps or facets or melts rows into event streams, preserves source columns as metadata, and returns `EventEntry` records.
`replay/log_to_rerun.py`	Contains the shared replay engine. `log_source()` consumes a `ReplaySource`, logs every modality to Rerun, creates trajectories and video references, logs events, and sends a matching viewport blueprint. The same file also implements the desktop CLI and `.rrd` export path.
`replay/notebook_app.py`	Adds the Jupyter/VS Code interface around the shared logging engine. It embeds the viewer and provides event search, click-to-jump from trajectory points, selection-aware notetaking, and JSON note-sidecar export.
`replay/__init__.py`	Defines the public package API, including `ReplaySource`, registry helpers, `log_source()`, and notebook `launch()`.
`RunExamples/*.ipynb`	Provides dataset-specific examples that configure and launch the notebook application. These notebooks call the package; they do not contain the core replay implementation.
`tests/test_*.py`	Verifies event parsing, source registration, Rerun logging and blueprint behavior, notebook controls, and dataset-specific adapter behavior.
`pyproject.toml` / `environment.yml`	Define the installable Python package and its pip/Conda dependencies. `environment.yml` also installs FFmpeg, which is used for video-backed replay.

End-to-End Replay Flow

The runtime flow consists of the following steps:

1) **Select an entry point and source.** A replay can begin from either `python -m replay.log_to_rerun` to launch the native desktop Rerun application or `replay.notebook_app.launch()` to launch a notebook-embedded viewer. The user supplies a source key and dataset path, either through the CLI call or the notebook viewer. For example, via the CLI `--source scout` selects `ScoutRosbagSource` (parsing ROS 1 bags[18]), while via the notebook `launch(source_name="hdepic", ...)` selects `HdepicSource` (ingesting HD-EPIC 3D digital twins and gaze streams[2]).
2) **Register and construct the adapter.** Importing `replay.sources` registers the built-in classes in `registry.py`. `build_source(),` then creates the requested adapter and passes through dataset-specific options such as event-table paths or column mappings.
3) **Inspect the native data.** The adapter discovers the streams it can support; determines the recording bounds; and prepares any required indices, transforms, calibration, or sidecar locations. This complexity remains inside the adapter.
4) **Emit normalized records.** The adapter yields instances of the data classes in `base.py`. Each record has an elapsed time stamp and a hierarchical Rerun `entity_path`; for example, a camera pose and image may be mapped as `world/camera` and `world/camera/image`.
5) **Log each stream on one timeline.** `log_source()` sets `replay_time` to each record's time stamp and logs the corresponding Rerun archetype. Streams do not need identical sampling rates and are not resampled into a single table. Rerun resolves the state of each entity at the selected playback time.
6) **Create the viewport layout.** After identifying which image and event streams were actually logged, `build_blueprint()` creates a 3D spatial view, one 2D view per image stream, and one text-log view per event stream. An event stream may request placement beside a related camera view through `paired_view`.
7) **Present or save the result.** The CLI either opens the native Rerun viewer or writes an `.rrd` recording. The notebook path embeds the same recording and then attaches Python widgets and viewer callbacks.
8) **Add notebook interactions without changing source data.** Search-result buttons and selected trajectory points move the shared `replay_time` cursor. Notes capture the active time and selection in a separate `<dataset>_replay_notes.json` file and are also logged to the current in-memory recording.

4. Case Study: Resolving Ambiguous Dialogue and Clarification Acts in Human–Robot Teaming

Here, we show a use case in the native Rerun viewer from the SCOUT dataset[3,4] to understand the utility of STAIT-Replay for understanding dialogue intent for developing agents.

In collaborative human–robot trials from the SCOUT dataset, a remote Dialogue Manager (DM) and Robot Navigator (RN) act as a robot to execute Commander (CM) orders to an unmanned ground vehicle. Transcripts contain clarification requests that are unclear what motivated this transaction when viewed as isolated text. Crucially, the CM operates under perceptual asymmetry with intermittent visual situational awareness, receiving only sporadic still pictures taken by the RN (visible in the event table), rather than continuous video. When reviewing isolated dialogue transcripts, the motivation behind clarification requests is often unclear. For example, in this back-and-forth exchange, the CM says to go through a doorway based on an intermittent still image; however, there are no doorways currently in the robot's live forward view. The only doorway in the robot's vicinity is blocked. The DM intuits that the CM may be referring to the hallway and asks accordingly, careful to specify which specific hallway (i.e., next to the orange bucket rather than to the right hallway out of the line of sight), and then translates to the RN to travel into the hallway. This interpretation and translation to appropriate instructions demonstrates how an analyst can identify critical behaviors and information for robots to be able to understand, clarify, and behave appropriately in collaborative tasks. Prior to STAIT-Replay, analysts looked at SCOUT data independently with multiple software platforms (Excel, VLC, etc.), leading to reduced situational awareness of the data. See Figure 2.

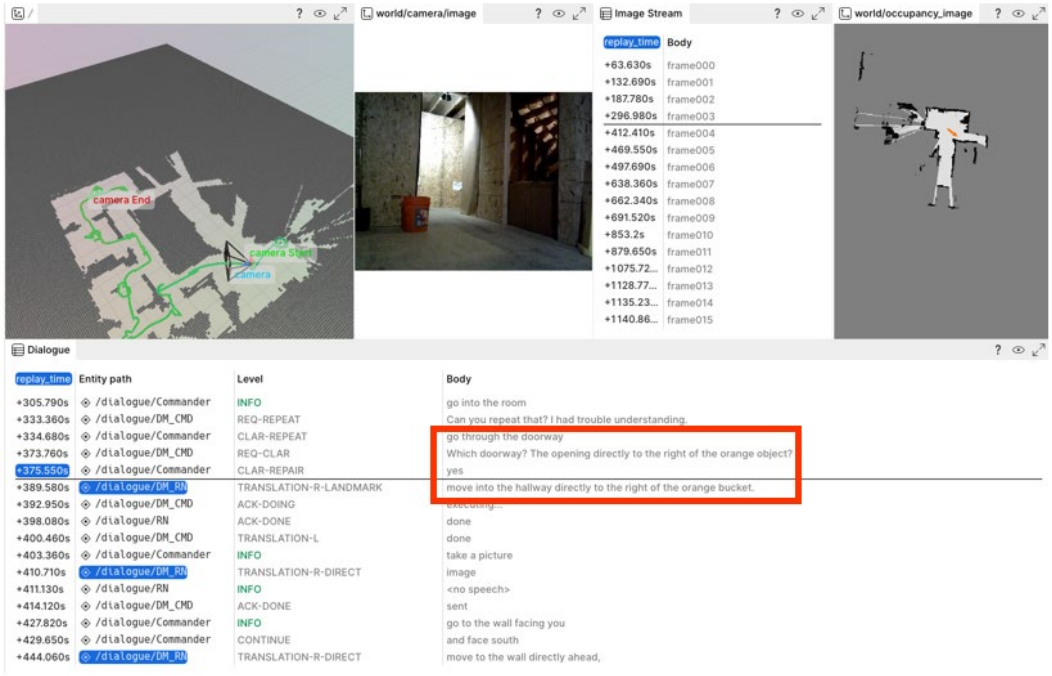


Figure 2. Screenshot of SCOUT dataset with red rectangle denoting unclear motivation behind communications.

5. Potential Operational Use Cases: Military Relevance and Technology Extensions

While STAIT-Replay in its current form serves as an exploratory, post-hoc visual auditing and research tool, its synchronized multimodal spatiotemporal architecture provides a foundational substrate for follow-on operational capabilities. In military and tactical environments—characterized by high-tempo operations, contested communications, and mixed human–machine formations—the underlying replay technology has strong potential to be extended into specialized diagnostic, verification, and mission-support systems:

1) **Embodied Vision-Language Model (VLM)/Large Language Model(LLM) Pipeline Diagnostics and Explainability for Autonomous Systems:** Modern multimodal natural language processing (NLP) pipelines increasingly integrate VLMs and LLMs to interpret natural language commands and generate robot action plans. Unlike structured symbolic semantic pipelines (e.g., Abstract Meaning Representation parsers[19]), LLMs introduce stochasticity and potential hallucinations. Future operational diagnostic extensions of STAIT-Replay could function as a visual debugging harness: when an embodied tactical agent takes an erroneous action or generates a nonsensical dialogue response in the field,

analysts can reconstruct the exact visual, spatial, and linguistic input received by the model at that millisecond to isolate the failure mode.

2) **Spatial Reasoning, Topological Memory and Long-Term Q&A Verification:** In spatial intelligence research (e.g., 3D dynamic scene graphs[20]), autonomous robots must build topological representations across multi-building tactical environments. For example, a robot navigating outside observes an enclosed elevated breezeway connecting two buildings, enters Building A, traverses the breezeway into Building B, and exits. When queried post-mission (*"Where did you just emerge from?"*), the agent must answer Building B despite having originally entered Building A. Extended replay tools would allow military supervisors to audit the entire spatial path at a glance, validating that topological transitions, visual landmarks, and inertial odometry correctly align with the robot's internal memory state.
3) **Multi-Robot Deception Detection and Temporal Scene Change Analysis:** In multi-agent tactical operations involving potential adversarial deception or environmental manipulation, multiple reconnaissance robots may survey the same geographical zone at different times. As a technology extension, comparative spatiotemporal auditing modules could allow analysts to overlay spatial observations from Robot 1 against later observations from Robot 2 (or dismounted reconnaissance teams) to detect whether environmental geometry has changed, or whether electronic warfare sensor tampering has introduced inconsistencies into an agent's telemetry.
4) **Human–Robot Exercise Benchmarking and After-Action Review (AAR):** Future operational extensions of STAIT-Replay could provide a standardized foundation for creating human–agent teaming benchmarks and military AAR systems. By extending notetaking into structured event scoring pipelines, military evaluators could catalog failure events during collaborative live–virtual–constructive trials, recording proposed human expert resolutions and benchmarking autonomous VLM/LLM models against human-authored decision-making under realistic mission stress.
5) **Offline Reconnaissance Assessment and Downstream Mission Planning:** Human mission commanders can leverage initial robot exploration streams visualized in STAIT-Replay as a first-pass reconnaissance survey. While STAIT-Replay remains read-only, human visual assessment of spatial coverage gaps can feed follow-on mission planning systems and spatial annotation tools (i.e., the STAIT-Mapper Refiner), enabling operators to author follow-on mission waypoints and task allocations offline based on visualized reconnaissance gaps.

6. Conclusion

`STAIT-Replay` addresses a critical bottleneck in robotics, embodied AI, and multimodal human–agent research: reconciling disparate 3D spatial geometry, 2D video feeds, dialogue, and discrete event streams. By providing dataset-agnostic coding tools and spatiotemporal dataset interactivity, `STAIT-Replay` enables researchers to decode agent intent, debug complex multimodal foundation models, and perform grounded 3D dataset refinement at scale.

7. References

1. Grauman K et al. Ego4D: around the world in 3,000 hours of egocentric video. In: Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2022 June 18–24; New Orleans, LA. Piscataway (NJ): IEEE; 2022. p. 18973–18990. https://doi.org/10.1109/CVPR52688.2022.01842

2. Perrett T et al. HD-EPIC: A highly-detailed egocentric video dataset. In: Proceedings of the 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2025 June 11–15; Nashville, TN. Piscataway (NJ): IEEE; 2025. p. 23901–23913. https://doi.org/10.1109/CVPR52734.2025.02226

3. Lukin SM et al. SCOUT: A situated and multi-modal human-robot dialogue corpus. In: Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024); 2024 May 20–25; Torino, Italy. ELRA and ICCL; 2024. p. 14445–14458.

4. Bonial C et al. Human–robot dialogue annotation for multi-modal common ground. Language Resources and Evaluation. 2025;59(4):1525–1575. https://doi.org/10.1007/s10579-024-09784-2

5. Clark HH, Brennan SE. Perspectives on socially shared cognition. In: Resnick LB, Levine JM, Teasley SD, editors. Perspectives on Socially Shared Cognition. Washington (DC): American Psychological Association; 1991. p. 127–149. https://doi.org/10.1037/10096-006

6. Kam HR et al. RViz: a toolkit for real domain data visualization. Telecommunication Systems. 2015;60(2):337–345. https://doi.org/10.1007/s11235-015-0034-5

7. Foxglove Technologies Inc. Foxglove Studio: robotics visualization and debugging platform [software]. https://foxglove.dev

8. Kassner M, Patera W, Bulling A. Pupil: an open source platform for pervasive eye tracking and mobile gaze-based interaction. In: UbiComp '14 Adjunct: Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication; 2014 Sep 13–17; Seattle, WA. New York (NY): Association for Computing Machinery; 2014. p. 1151–1160. https://doi.org/10.1145/2638728.2641695

9. Tobii AB. Tobii Pro Lab: eye tracking software for behavioral research [software]. https://www.tobii.com/products/software/behavior-research-software/tobii-pro-lab

10. Boersma P, Weenink D, Shchupak A. Praat: doing phonetics by computer [computer software]. Version 7.0.02. 2026. https://praat.org

11. Sloetjes H, Wittenburg P. Annotation by category: ELAN and ISO DCR. In: Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08); 2008 May 28–30; Marrakech, Morocco. Paris (France): European Language Resources Association; 2008. p. 816–820.

12. Cignoni P et al. MeshLab: an open-source mesh processing tool. In: Proceedings of the Sixth Eurographics Italian Chapter Conference; 2008 Jul 2–4; Fisciano, Italy. Geneve (Switzerland): Eurographics Association; 2008. p. 129–136. https://doi.org/10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136

13. Sloetjes H, Wittenburg P. Annotation by category: ELAN and ISO DCR. In: Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08); 2008 May 28–30; Marrakech, Morocco. Paris (France): European Language Resources Association; 2008. p. 816–820.

14. Kapp S, Barz M, Mukhametov S, Sonntag D, Kuhn J. ARETT: augmented reality eye tracking toolkit for head mounted displays. Sensors. 2021;21(6):2234. https://doi.org/10.3390/s21062234

15. Bohus D et al. Platform for situated intelligence. arXiv. 2021. Preprint arXiv:2103.15975. https://arxiv.org/abs/2103.15975

16. Cohen Hoffing R, Nonte M, Touryan J. Spatiotemporal analysis and inference toolkit: standardizing the organization and analysis of multi-agent, human–machine-derived mixed reality data. DEVCOM Army Research Laboratory (US); 2026 Apr. Report No.: ARL-TN-1306. https://arl.devcom.army.mil/arlreport/arl-tn-1306/

17. Rerun Technologies. Rerun: an open-source SDK for spatial and temporal multimodal data visualization [computer software]. Version 0.34.1. https://rerun.io

18. Quigley M et al. ROS: an open-source Robot Operating System. In: Proceedings of the ICRA Workshop on Open Source Software; 2009 May 12; Kobe, Japan. Piscataway (NJ): IEEE; 2009. p. 5.

19. Banarescu L et al. Abstract meaning representation for sembanking. In: Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse; 2013 Aug 8–9; Sofia, Bulgaria. Stroudsburg (PA): Association for Computational Linguistics; 2013. p. 178–186.

20. Rosinol A, Gupta A, Abate M, Shi J, Carlone L. 3D dynamic scene graphs: actionable spatial perception with places, objects, and humans. arXiv. 2020. Preprint arXiv:2002.06289. https://arxiv.org/abs/2002.06289

List of Symbols, Abbreviations, and Acronyms

2/3D	Two-/Three-Dimensional
AAR	After-Action Review
AI	Artificial Intelligence
API	Application Programming Interface
CM	Commander
CSV	Comma Separated Values
CVPR	Computer Vision and Pattern Recognition
DM	Dialogue Manager
LiDAR	Light Detection and Ranging
LLM	Large Language Model
NLP	Natural Language Processing
PSI	Platform for Situated Intelligence
RN	Robot Navigator
STAIT	Spatiotemporal Analysis and Inference Toolkit
VLM	Vision-Language Model

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