Deploying Sensemaking Robots in Healthcare Environments

Krishna Pavani Munta[1], Ilyass Tran[1,2] and Enrico Motta[1]

[1] *Knowledge Media Institute, The Open University, Walton Hall, Milton Keynes MK7 6AA, United Kingdom*
[2] *CESI Lineact, CESI School of Engineering, Labège 31670, France*

Abstract

Busy hospital environments need systems in place to monitor and prevent safety hazards, and more generally to detect and deal with anomalous situations. Sensemaking robots can play an important role in these scenarios, especially in contexts (e.g., at night) when staff resources are limited. Unfortunately, standard deep learning models on mobile robots are not flexible enough to deal with the dynamics of real-world environments. To address this issue, we present a novel robot cognition framework that integrates computer vision models with structured knowledge, in the form of ontological knowledge and common-sense filters, to help robots to make sense of the environment around them. This approach filters out hallucinations and improves the performance of the robot on sensemaking tasks. In addition, it allows users to modify the explicit knowledge representation parameters dynamically – e.g., specifying a 'keep clear' radius for blocked exit detection, by editing the relevant common-sense parameters. The initial demonstration produces promising results, showing that our architecture allows robots to identify several safety hazards, such as spillages and blocked exits.

Keywords

robot cognition, anomaly detection, knowledge engineering, common sense reasoning, e-health.

1. Introduction

The environment in a busy hospital is very volatile and continuously changing. As a consequence of severe clinical staff shortages since Brexit and the COVID-19 pandemic [1], UK hospitals have been experiencing very busy environments with increased staff workload. Due to these circumstances, it is very difficult to monitor hospital safety effectively throughout a 24-hour cycle. To address these problems, facilities typically deploy computer vision models on security cameras. However, a wall-mounted security camera is static; it cannot target all the hazards and detect them [2]. While traditional deep learning vision-based models can be used to identify hazards, they cannot understand the context around them (e.g., classify a bag found in a corridor as abandoned based on context.). As a result, these AI models are prone to generating false alarms, which might increase the workload for the hospital staff [3].

This problem affects most deep-learning based approaches to anomaly detection. For instance, the EfficientAD solution tackles anomaly detection by detecting pixel-level anomalies in an environment [5]. As a result, it cannot reason about the context around potential anomalies. For this reason, our approach builds on the neurosymbolic architecture for health

The 25th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2026) - Satellite Events (Workshops, Tutorials, Demos, Posters)
krishna.munta@open.ac.uk (K. P. Munta); ilyass.tran@viacesi.fr (I. Tran); enrico.motta@open.ac.uk (E. Motta)
© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR-WS.org/Vol-4340/paper8.pdf

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-26

and safety inspection proposed by Chiatti et al. [6]. We aim to improve this work by taking advantage of the latest developments in computer vision and integrating them with knowledge representation and common-sense reasoning. In this paper, we present a demonstration of our initial prototype, which allows robots to identify correctly safety hazards, such as spillages and blocked exits.

2. Architecture

The system's architecture is designed to be modular and adaptive, comprising two primary layers: a neural and a reasoning layer. The neural layer consists of two models, You Only Look Once (YOLO) [9] and Grounding DINO [8], running together and functioning as the eyes of the robot. Additionally, the framework selectively uses the Segment Anything Model 3 (SAM 3) [10] on an as-needed basis for tasks requiring shape extraction. However, visual detection alone is insufficient for autonomous safety monitoring in hospitals. Therefore, we added a reasoning layer that functions as a knowledge engine, applying structured rules to support common sense reasoning.

2.1. System Setup

Currently, the system uses an offline data processing pipeline to manage computation. The physical robot, equipped with an onboard RGB-D camera, navigates through the clinical environment. Instead of processing data live, it records video and 3D depth into sensor logs (ROS bag files). The recorded bag files are transferred to a dedicated, high-performance laptop equipped with an NVIDIA RTX 5090 GPU.

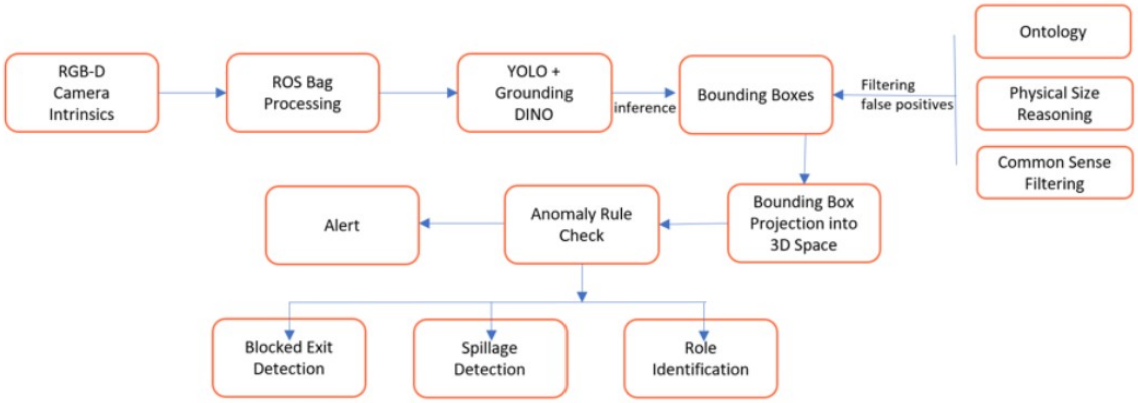


Figure 1. Architecture of the System.

As shown in **Figure 1,** the frames are first passed through the neural layer comprising object detection models to generate bounding box detections. These detections are then filtered using physical size reasoning and common-sense filtering, before being projected into 3D space. The resulting 3D detections are passed on to the anomaly rule check, which applies the reasoning-layer logic to identify hazards such as blocked exits, and to generate structured safety alerts.

2.2. Neural Layer

The neural layer combines a deep learning model and a vision transformer: (a) *You Only Look Once (YOLO) v26*, finetuned on hospital-relevant objects, and (b) *Grounding Detection Transformer with Improved Denoising Anchor Boxes (DINO)*, an open-vocabulary detector used only to verify ambiguous or novel classes.

The inference engine comprises a YOLO trained on hospital objects. The model detects 60 hospital-related objects. The object classes covered by YOLO can be found in Appendix – List of Object Classes. This layer performs checks continuously on every frame and detects objects. For accuracy, we applied class-specific confidence thresholds. When the confidence score for a YOLO detection falls below the threshold, DINO is triggered as a fallback detector and detects the missing objects. *Grounding DINO,* a vision language transformer, detects objects with a single prompt, allowing it to detect classes outside YOLO's training set. It also covers weaker classes that YOLO struggles to detect. However, this neural layer may create bounding boxes which might be inaccurate. For this reason, these outputs are checked by symbolic reasoning.

2.3. Reasoning Layer

This layer acts as a reasoning gate for the detections produced by the neural layer. This layer sets dimensional and common-sense constraints.

2.3.1. Physical Size and Depth Reasoning

For some classes, when the neural layer produces a bounding box (BB) that requires further verification, rather than accepting the detection, the system projects the BB coordinates onto the 3D points created by the RGB-D camera. By performing this projection, the two-dimensional BB can now represent the actual physical size of the object. The resulting physical size measurements are then compared to pre-defined "real world" physical size estimates. If an object is found to have an incorrect physical size estimate (i.e., a BB produced for a "bin" cannot possibly be a "door"), then the system automatically removes that particular detection from the system to avoid triggering an incorrect anomaly. In the future, these checks will be integrated into a knowledge graph to support reasoning in a wide variety of situations [4].

2.3.2. Common Sense Reasoning

This filtering acts as a hard logical gate, enforcing spatial relationship rules based on clinical reality. Any detection that violates physics or commonsense logic, e.g., identifying an object on the floor as a security camera would violate the commonsense rule that, in most practical scenarios, security cameras are mounted on a wall or a ceiling. This is achieved by performing spatial reasoning on the relation between the object and the floor, by calculating the distance from the floor of the detected bounding box. This is a rather generic solution that allows us to handle most constraints through spatial reasoning [6]. By applying these predefined spatial rules, the false detection is immediately regarded as hallucination and removed. The current system implements this functionality in a limited way, using a Python Script file. In future work, we plan to replace this flat rule-based approach with scene graphs, which will allow these rules to be modeled more formally as "validation constraints" within a structured graph representation [7].

2.3.3. Ontology

Reasoning is supported by a formal ontology consisting of 60 classes. This provides the necessary semantic vocabulary for reasoning about anomaly detection in a clinical environment. The ontology distinguishes an object's identification from its situational identification and arranges all concepts according to six root classes, *Physical Object, Person, Location, Event, Role, and Context*. A key relation in the ontology is the *playsRole* property, which gives situational status (such as Obstacle) to an object. For instance, a Wheelchair is ontologically a Chair but can be classified as an Obstacle if abandoned in front of an exit door. The Context class allows us to define the circumstances under which an object is assigned a particular role. For instance, a cable becomes a Trip Hazard when its location obstructs pedestrian circulation. Thus, the hazard arises from the context, not from the cable itself. A preliminary version of the ontology can be found on GitHub. Currently, Ontology for various objects has been defined but not yet integrated with the framework. Ongoing efforts are focused on using ontology as a reasoning layer on top object detection.

2.4. Anomaly Detection

The system detects anomalies using the spatial and logical rules described above, together with additional health-and-safety-specific rules. It aims to provide early warnings by identifying situations that may require human attention, spanning from environmental hazards to access-related issues.

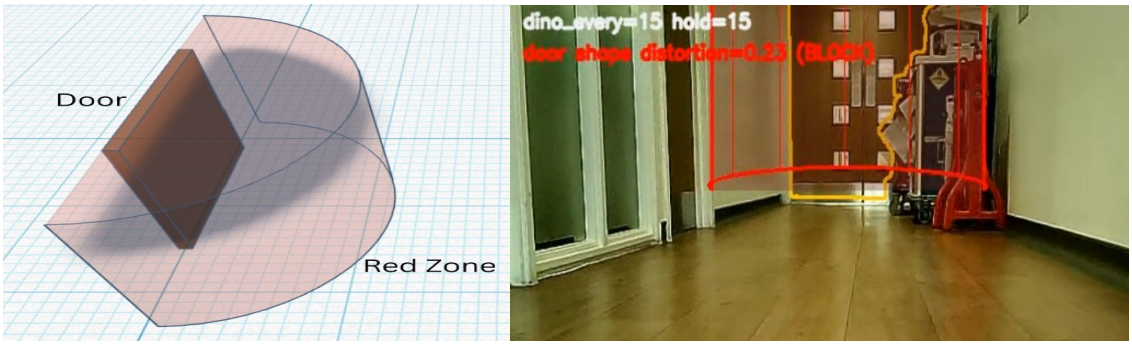


Figure 2. (a) Keep Clear Zone for an Exit Door and (b) Real-World Blocked Exit Detection

A core module addresses blocked exit detection using RGB and Depth data. Once the system detects a door, its region is refined through segmentation using Meta's *Segment Anything Model (SAM 3)*. Later, each refined object is projected onto the depth map to get its 3D position. The module uses an ontology filter to discard false positives for doors. As shown in Figure 2, The system constructs a 3D semi-cylindrical "*keep clear zone*" around the door, where the cylinder's height is equal to the door height. If another detected object's 3D points fall into this keep clear zone for a duration *t*, where $t \geq 5$ seconds, the system alerts that the exit is being blocked. Alongside this, the neural layer is trained on the spillages; once the spillages are detected on the floor, an immediate alarm is triggered.

Beyond environmental hazards, the system is also designed to flag civilians entering restricted areas. The current system is concerned with identifying different roles of a person such as a *doctor, patient, healthcare worker,* or a *civilian,* based on their appearances and visual

cues such as *stethoscopes, lab coats, hospital gowns, civilian dressing,* and *scrubs.* In the future, this module will be extended to zone-aware access control with the inclusion of a semantic map of the facility.

As our research proceeds further, we plan to extend the system to cover a broader range of anomaly detection scenarios, including unsupervised children in hospital settings, fire and smoke hazards, improperly secured doors and windows, and trip hazards such as loose cables or misplaced equipment, and others. Each of these scenarios will be integrated into the reasoning layer following the approach, combining visual detection with context-aware rules; ensuring the system remains adaptable as new hospital-specific risks are identified.

3. Demonstration and Results

The performance of the model on synthetic data is 92.7% (mean average precision). This number is indicative only for object detection without the connection with ontology and common-sense reasoning. However, we are of course interested in performance in highly dynamic real-world clinical environments, which pose a much bigger challenge. We have carried out preliminary tests on video footage recorded in an actual hospital environment in Milton Keynes, which has produced promising results (precision = 61.6%). This number is indicative with filtering rules and physical dimension gating. The drop in performance is due to low-resolution videos, occlusions, blur, and lighting variations. Here, it is important to emphasize that these are early results; further rigorous quantitative testing and ablation studies will be needed to assess the overall performance of the architecture on a variety of anomaly detection scenarios.

The demonstration features a live dashboard allowing the attendees to visually look at the system flagging different kinds of anomalies, such as blocked exits and spillages. The system can also detect objects across 60 object classes on which the models are trained, with results shown directly in the visualization. The demo is also interactive; attendees can adjust parameters such as the radius of the semi-cylindrical keep-clear zone to see how detection changes. Recorded demonstrations of the working system, which include object detection and blocked exit detection, are available on Zenodo at this location.

4. Ongoing and Future Work

The current system provides a foundation for the main goal we are working towards, the system can detect hospital-related objects, make sense of the detected object's dimensions, check whether the object is correctly placed, and then make decisions about anomaly detection based on the relevant rules. However, at the time of writing, several relevant anomaly detection rules have not been implemented yet, across all 60 objects. Hence, as already pointed out, we plan to extend the system to cover a broader range of anomaly detection scenarios. Another task will focus on achieving a closer integration of the current Python-scripted checks with the ontology, using Knowledge Graphs and Scene Graphs, to develop robust neurosymbolic architecture.

Later, the person role identification module will be integrated into a semantic map built with the help of a robot in the clinical facility. We are also refining the neural layer to increase accuracy. This includes enriching the training datasets with edge-case scenarios and connecting them to the ontology to reduce false positives. The ongoing research repository is on GitHub and can be found here.

Acknowledgements

This research was partly funded by an ITEA 4/Innovate UK grant – project Resilient Enterprise (ITEA 4 grant #23046, Innovate UK grant #600646). We would like to acknowledge our collaborators at Swift Robotics for supporting our experiments and data collection activities.

Declaration on Generative AI

During the preparation of this work, Google Gemini was used to assist with Grammar and spelling checks, Improve writing style, and Citation management. After using these tools, the content has been reviewed and edited as needed.

References

[1] H. Alderwick, Conservative Party's legacy on the NHS, BMJ (2024) q1491. URL: https://doi.org/10.1136/bmj.q1491.

[2] M. Boshoff, B. Kuhlenkötter, P. Koslowski, Dynamic camera planning for robot manufacturing processes using a UAV, Robotics 14 (2025) 23.

[3] S. M. H. Abidi, S. M. Raza, S. Y. Shin, SafeVision: Vision–language reasoning for context-aware safety monitoring, Neurocomputing 669 (2026) 132479. doi:10.1016/j.neucom.2025.132479.

[4] W. Hao, J. Menglin, T. Guohui, M. Qing, L. Guoliang, R-KG: A novel method for implementing a robot intelligent service, AI 1 (2020) 117–140. doi:10.3390/ai1010006.

[5] K. Batzner, L. Heckler, R. Koenig, EfficientAD: Accurate Visual Anomaly Detection at Millisecond-Level Latencies, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2024, pp. 128-138. doi:10.48550/arXiv.2303.14535.

[6] A. Chiatti, E. Motta, E. Daga, Robots with Commonsense: Improving Object Recognition through Size and Spatial Awareness, in: A. Martin, K. Hinkelmann, H.-G. Fill, A. Gerber, D. Lenat, R. Stolle, F. van Harmelen (Eds.), AAAI 2022 Spring Symposium on Machine Learning and Knowledge Engineering for Hybrid Intelligence (AAAI-MAKE 2022), Palo Alto, California, USA, 21-23 Mar 2022, CEUR Workshop Proceedings, CEUR-WS.org, 2022. URL: https://oro.open.ac.uk/81576/

[7] X. Chang, P. Ren, P. Xu, Z. Li, X. Chen, A. Hauptmann, A comprehensive survey of scene graphs: Generation and application, IEEE Transactions on Pattern Analysis and Machine Intelligence 45 (2023) 1–26. doi:10.1109/TPAMI.2021.3137605.

[8] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, J. Zhu, L. Zhang, Grounding DINO: Marrying DINO with grounded pre-training for open detection, in: Lecture Notes in Computer Science, Springer, 2024, pp. 38–55. URL: https://doi.org/10.1007/978-3-031-72970-6_3.

[9] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2016, pp. 779–788. URL: https://doi.org/10.1109/cvpr.2016.91.

[10] N. Carion, L. Gustafson, Y.-T. Hu, S. Debnath, R. Hu, D. Suris, et al., SAM 3: Segment Anything with Concepts, arXiv preprint arXiv:2511.16719 (2025).

A. Appendix – List of Object Classes

This section describes the object classes detected within hospital environments by the system. The primary detection is handled by the YOLO model. Grounding DINO serves as a fallback for objects YOLO may miss and functions as a detector for new classes not included in the original YOLO training set. Certain classes also feature logic-based sub-classes identified through specific visual patterns and colours. The total number of classes are 60.

Trained YOLO Model

1. person
2. fire hydrant
3. bench
4. backpack
5. handbag
6. suitcase
7. baseball glove
8. bottle
9. cup
10. fork
11. knife
12. spoon
13. chair
14. Couch
15. bed
16. dining table
17. toilet
18. sink
19. scissors
20. cabinet
21. glove
22. healthcare_worker
23. hospital_bed
24. infusion_pump
25. iv_bag
26. iv_stand
27. nasal_cannula
28. patient
29. patient_monitor
30. surgical_light
31. test_tube
32. vending_machines
33. wheelchair

34. bench_hosp
35. door
36. reception_counter
37. radiator
38. bathroom_labels
39. fire_extinguisher
40. hospital_stretcher
41. security_camera
42. hair_net
43. mask
44. surgical_scissor
45. bag
46. exit_sign
47. spillage
48. bin
49. Electrical_cabinet
50. Hand_sanitizer
51. Hazmat_sign
52. wet_floor_sign

DINO Fallback Classes
1. surgical_scissor
2. surgical_light
3. glove
4. mask
5. hair_net
6. radiator
7. exit_sign
8. door
9. hand_sanitizer
10. bin
11. hazmat_sign
12. iv_stand
13. Infusion_pump

DINO Only Additions
1. Medical_tray
2. Power_socket
3. Oxygen_pump
4. Utility_trolley

Sub-classes for Main Classes
1. Bin
a. General_bin
b. Tiger_stripe_bin

c. yellow_bin
2. Person
a. Person (YOLO)
b. Doctor
c. Patient (YOLO)
d. Healthcare_Worker (YOLO)