# Generating 3D Scene Graphs from RGB-D Video

Ahmad Hamid
Faculty of Engineering (LTH), Lund University
Lund, Sweden

Ibrahim Aljuboori
Faculty of Engineering (LTH), Lund University
Lund, Sweden

***Abstract*—Scene understanding is a central part of modern computer vision systems, especially in robotics and autonomous systems where it is not enough to simply identify objects. To enable a deeper understanding of the environment, information about the spatial positions of the objects and their relationships to each other is also required. In this work, a pipeline for generating 3D scene graphs from RGB-D video data is presented. The proposed method combines YOLO-based object detection and segmentation, hand tracking with MediaPipe, depth-based 3D localization, temporal tracking with XMem, and inference of spatial relationships between objects and hands. The generated scene graphs are stored in a structured JSON format and visualized in three dimensions for analysis and verification. The results show that the system can successfully detect and localize objects and hands, maintain consistent identities across multiple frames, and infer meaningful spatial relationships. Despite some limitations, such as misclassification of objects and the use of rule-based relational inference, the work demonstrates that it is possible to generate structured 3D scene graphs from RGB-D data and lays the foundation for future research in scene understanding and human-object interaction.**

## I. Introduction

Modern computer vision systems have become very accurate in both object detection and classification. However, such tasks only provide a limited understanding of a given scene. Humans naturally incorporate spatial context and the relationships between surrounding objects, which allows for a more comprehensive interpretation of the environment. Therefore, it can be said that modern computer vision systems require more than object-level information to reason about interactions and scene structure. Hence, the interest in approaches that model not only the objects present in a scene, but also the relationships between them is growing rapidly. One such representation is the 3D scene graph, where objects are represented as nodes and the relations between them as edges. Such a structured and comprehensive description of the environment is beneficial in fields such as robotics and autonomous systems.

A 3D scene graph's goal is to provide a structured representation of objects and their relationships with one another in a three-dimensional environment. However, such a three-dimensional representation is not only about the x and y (horizontal and the vertical) axes, but also about the z (depth) axis. Hence, RGB-D camera sensors come into play, such cameras provide four types of output for each pixel in a given frame, the three RGB color values, and the more advanced Depth value. The depth value for each pixel is can be used to estimate the three-dimensional position of given objects. Hence, enabling the construction of the scene graphs which capture semantic and spatial information.

In this work, we provide a pipeline for generating 3D scene graph representation of RGB-D camera input. The proposed approach utilizes object segmentation, hand tracking, and 3D localization of objects. The generated scene graph is then stored in structured JSON format and visualized for demonstration and analysis.

The main contributions of this work can be summarized as follows:

- A pipeline for generating 3D scene graph representations from RGB-D video data.
- Integration of YOLO-based object segmentation and MediaPipe hand tracking within a unified framework.
- Depth-based localization of objects and hands in a shared 3D coordinate system.
- A structured JSON representation and 3D visualization framework for scene analysis and future robotic applications.
- Public repo: https://github.com/ikawan/3DSceneGraph.git

## II. Related Work

A. Scene Graph Generation Research on scene graph generation has grown rapidly in recent years. One of the early methods was presented by Xu et al. [1], where scene graphs were directly generated from visual data. Their work created the foundation for upcoming methods that utilize object detection and relational understanding together to create structured representations of a scene.

B. RGB-D Scene Understanding and Manipulation The development of RGB-D sensors has enabled more advanced scene understanding by combining visual information with depth data. The KIT Bimanual Manipulation Dataset [2] introduced a comprehensive collection of synchronized RGB-D sequences of everyday manipulation activities along with detailed annotations of objects and actions. Previous work based on the dataset has mainly focused on activity recognition and analysis of human-object interactions. In this work, the same type of data is used to generate structured 3D scene graphs.

C. Hand Tracking and Human-Object Interaction Understanding human-object interactions is a central part of manipulation-based scene understanding. Modern hand tracking methods, such as MediaPipe Hands [3], enable robust estimation of hand landmarks in real time and have therefore become common in both robotics and computer vision. In the

proposed method, hand tracking is used as a complement to object detection to provide a more complete representation of the scene.

## III. System Design and Methodology

The proposed system generates 3D scene graphs from RGB-D video through five main stages: object detection and segmentation, hand detection, 3D localization, temporal tracking, and spatial relation inference. Figure 1 shows an overview of the complete pipeline.

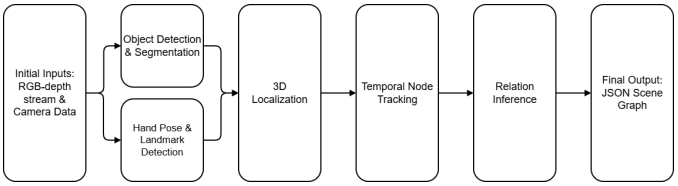


Fig. 1. System architecture for the proposed 3D scene graph generation pipeline.

TABLE I
Summary of the main processing modules.

| Module | Input | Output |
|---|---|---|
| YOLO Segmentation | RGB Frame | Object Masks |
| MediaPipe Hands | RGB Frame | Hand Landmarks |
| 3D Projection | RGB-D Data | 3D Coordinates |
| XMem Tracking | Object Masks | Persistent IDs |
| Relation Engine | 3D Nodes | Spatial Relations |
| JSON Export | Scene Graph | JSON Output |

### A. System Overview

The system processes synchronized RGB frames, depth maps, and camera intrinsics. RGB data is used to detect objects and hands, while depth information is used to estimate their positions in 3D space.

To support different datasets and hardware configurations, the pipeline can operate even when camera calibration data is unavailable. In such cases, a fallback pinhole camera model is estimated using the image resolution and an assumed field of view. This allows the remaining stages of the pipeline to continue operating without requiring device-specific calibration information.

Object detection and hand detection are performed in parallel on each frame. The resulting detections are projected into a shared 3D coordinate system using the aligned depth map. These 3D entities are then tracked over time to maintain consistent identities across frames. Finally, spatial relationships are inferred between entities and stored in a structured scene graph representation.

### B. Object Detection and Segmentation

Objects are detected using the YOLO26 instance segmentation framework [4]. For each frame, the model produces object class labels, confidence scores, bounding boxes and segmentation masks. These detections form the basis of the scene graph nodes generated later in the pipeline.

Several YOLO segmentation variants were evaluated using a manually annotated dataset containing approximately 5,400 images. The largest model, YOLO26X-seg, was used as a reference and achieved an average F1 score of 0.881 with an average inference time of 25.0 ms per frame.

YOLO26M-seg was selected for the final system because it provided a better balance between accuracy and speed. The model achieved an average F1 score of 0.839 and reduced inference time to 18.2 ms per frame, which translates to approximately 55 FPS. This performance allowed the system to maintain real-time operation while leaving computational resources available for tracking and relation inference.

The segmentation masks produced by YOLO are particularly important for the later 3D localization stage. Instead of relying only on bounding boxes, depth values can be sampled directly from the segmented object region, resulting in more accurate 3D position estimates and bounding volume calculations.

The output of this module is a set of object detections containing class labels, confidence scores, segmentation masks and bounding box coordinates.

### C. Hand Detection

Hand detection is performed using the MediaPipe Hands framework. For each frame, the system extracts 21 key landmarks per hand, along with a handedness label (left or right). These landmarks represent the main joints of the fingers and palm, which are used to estimate hand position and shape.

The hand detection module runs in parallel with object detection to ensure both are processed at the same temporal resolution. The output includes 2D landmark positions in image coordinates, which are later projected into 3D space using depth information.

Egocentric video introduces challenges such as mirrored viewpoints and frequent occlusions. To handle this, a simple correction step is applied to ensure that left and right hand labels remain consistent with the camera perspective. This step improves stability when tracking interactions between hands and objects over time.

The output of this module is a set of hand landmarks per frame, which are treated as additional nodes in the scene graph.

### D. 3D Localization

To construct a 3D scene representation, detections from both the object and hand modules are projected from the image plane into a shared 3D coordinate system. This is done using the aligned depth map and camera intrinsic parameters.

The first step is a simple depth filtering step. Objects that fall outside a defined interaction range relative to the user are ignored during downstream processing. This helps reduce false positives from background regions such as walls or distant objects.

For each detected point $(u, v)$, the corresponding depth value $Z$ is extracted from the depth image. The 3D coordinates $(X, Y, Z)$ are then computed using the standard pinhole camera model:

$$X = \frac{(u - c_x) \cdot Z}{f_x} \tag{1}$$

$$Y = \frac{(v - c_y) \cdot Z}{f_y} \tag{2}$$

where $f_x$ and $f_y$ are the focal lengths, and $c_x$, $c_y$ are the principal point offsets.

For object detections, depth values are sampled from within the segmentation mask rather than from a single pixel. This reduces noise and improves the stability of the estimated 3D position. A robust representative depth value is computed using the median of valid depth pixels inside the mask.

Each object is then represented as a 3D centroid along with an estimated 3D bounding box. The bounding box is constructed using percentile-based filtering of depth and spatial coordinates to reduce the effect of outliers.

The output of this stage is a set of 3D object and hand nodes, each with a stable spatial position in the camera coordinate system.

### *E. Temporal Tracking*

Objects and hands detected in individual frames are not inherently consistent over time. To build a usable scene graph, each entity must maintain a stable identity across frames. This is handled using a temporal tracking module based on XMem [5].

XMem is used to associate detections across time by maintaining a memory of past visual features. When a new frame is processed, each detected object is compared against stored features from previous frames. If a match is found, the existing object ID is reused. If no match is found, a new ID is assigned.

This approach allows the system to handle common challenges such as occlusions and temporary object disappearance. For example, if a hand moves in front of an object and blocks it from view, the object can still be correctly re-identified once it becomes visible again.

In addition to feature-based matching, a fallback association method is used when confidence is low. This method compares spatial position, bounding box overlap and depth similarity between current detections and existing tracks. A match is accepted if these measures exceed predefined thresholds.

The output of this stage is a set of temporally consistent object and hand tracks, each with a stable identifier across the video sequence.

### *F. Spatial Relation Inference*

Once objects and hands are represented as stable 3D nodes, spatial relationships are computed between them. The goal of this stage is to convert raw spatial positions into meaningful relationships that describe the scene structure.

Relations are computed by comparing every pair of nodes in the scene. For each pair, geometric distance and spatial alignment are evaluated to determine whether a relation exists.

A basic set of rules is used to define relationships:

- **Near**: Two objects are considered near if the Euclidean distance between their 3D centers is below a fixed threshold.
- **Touching**: A touching relation is assigned when the distance between object boundaries is close to zero, verified using both 2D mask overlap and 3D proximity.
- **Above / Below**: These relations are determined by comparing vertical positions in the camera coordinate system.
- **On Top Of**: This is inferred when an object is above another object and also satisfies the touching condition.

These rules are applied consistently across all frames, allowing the system to build a structured representation of spatial interactions over time, an example is shown in figure 2.

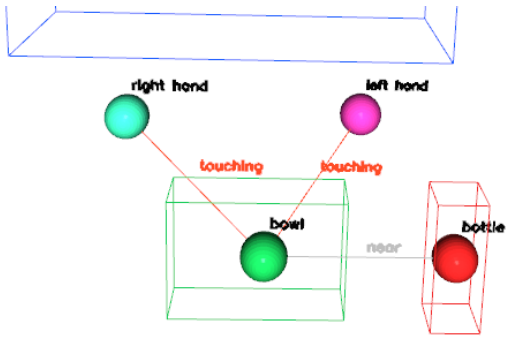


Fig. 2. Example of nodes and relations inferred between them.

### *G. Scene Graph Representation*

The final output of the system is a structured scene graph stored in JSON format. This format is used because it is simple to parse, easy to extend and compatible with downstream applications such as robotics and simulation systems.

Each frame is represented as a graph consisting of nodes and edges. Nodes correspond to detected objects and hands, while edges represent spatial relationships between them.

Each node contains information about its class, confidence score and spatial position in 2D and 3D space. For objects, additional information such as bounding boxes and segmentation mask statistics is also stored. Hand nodes contain landmark positions and a unique identifier.

Edges describe relationships between nodes, such as *near*, *on_top_of*, or *touching*. These relationships are generated in the spatial relation inference stage and stored as directed connections between nodes.

An example of the final scene graph structure is shown figure 3.

This representation keeps the scene graph compact while preserving both spatial structure and interaction information. It also allows the system to scale to longer video sequences by storing each frame independently or linking consecutive graphs over time.

```
"mask_area": 25717,
"median_depth_m": 1.6920000314712524,
"center_3d_m": [
  -0.0777951255440712,
  -0.282344251871109,
  1.6920000314712524
],
"bbox_3d_m": {
  "min_xyz": [
    -0.446522057056427,
    -0.6215885877609253,
    1.533000111579895
  ],
```


Fig. 3. Example snippet of JSON scene graph representation.

## IV. Results

The proposed pipeline was evaluated on RGB-D sequences from the KIT Bimanual Manipulation Dataset. Figure 4 shows an example of an RGB frame together with the 3D scene graph generated from it by the system.

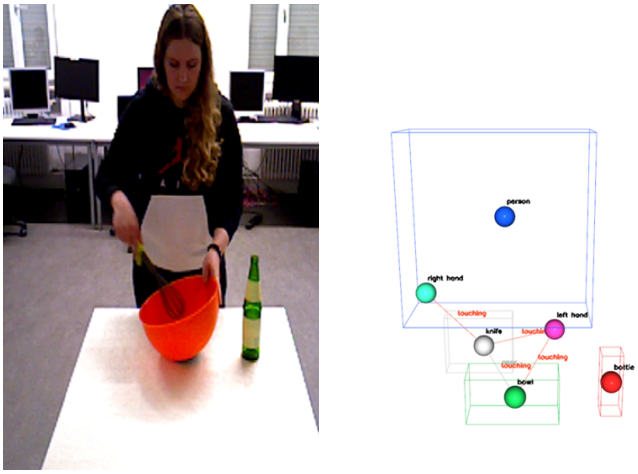


Fig. 4. Example result generated by the proposed pipeline. Left: RGB input frame. Right: Corresponding 3D scene graph showing detected objects, hands, and inferred spatial relationships.

The results show that the system was able to successfully detect and localize multiple objects in shared three-dimensional coordinate system. In the shown example, objects such as a person, both hands, a bowl, a knife, and a bottle were detected. The detected objects were represented as nodes in the graph and positioned in each of their estimated three-dimensional coordinates. However, it should be mentioned that the system misclassified some objects, an example of that can be observed in figure 4, where a whisk was misclassified as a knife.

Furthermore, the system was able to infer relations between the generated nodes. In figure 4, multiple *touching*-relations were illustrated between the hands, the knife, and the bowl. This enabled a more structured description of the scene were both the objects and their interactions where represented.

For object detection, the YOLO26M-seg model was chosen as it showcased a good balance between accuracy and computational cost. The model reached an average F1-score of 0.839 while maintaining an inference speed of approximately 55 FPS, which enables real-time processing within the complete pipeline.

The qualitative evaluation shows that the generated scene graphs preserves both of the semantic and the spatial information from the original RGB-D observations. The visualization tool also facilitated verification of the positions of the objects and the inferred relationships between them.

## V. Limitations and Future Work

Despite the fact that the proposed pipeline successfully genertes structured 3D scene graphs from RGB-D video data, there still exists some limitations.

The first limitaion is related to object detection. The system uses a YOLO-seg model that is pretrained on the COCO-dataset which does not include all the classes that are in the KIT Bimanual Manipulation Dataset; as a result, certain objects are misclassified. An example of this can be seen in figure figure 4, where a whisk was classified as a knife. Furthermore, and due to the fact that there exists no ground truth for the segmentation's masks, training a custom object detection/segmentation model was deemed outside the scope of this project. Therefore, a future work would be to address this limitation by creating task specific annotations and train custom detection/segmentation model.

Another limitaion is related to the relation inference. Relations such as *near*, *touching* and *on top of* are currently generated with the help of manually defined geometrical and distance rules. Even though such relations work well in simple scenarios, such a method lacks robustness in more complex situations with multiple objects' interactions. Future work could therefore investigate learning-based methods for relational inference that can capture more advanced semantic relationships between objects in the scene.

A third limitation concerns the scope of the relationships that can be represented in the generated scene graph. The current implementation focuses on a limited number of geometric relationships, such as *near*, *touching*, *above* and *on top of*. Although these relationships provide valuable information about the spatial arrangement of objects, they do not fully capture the semantic aspects of human-object interactions. For example, the system can identify that a hand is touching an object, but cannot determine whether the object is being held, used or manipulated in any specific way. Future work could therefore expand the relation vocabulary with more advanced interaction-based relationships, such as *holding*, *pouring* or *cutting*, to enable richer and more meaningful scene understanding.

Despite the mentioned limitaions, results show that it is feasible to generate structured 3D scene graphs from RGB-

D data and that the proposed pipeline provides a stable foundation for future research within scene understanding and human-object-interaction.

## VI. Conclusion

In this paper, a pipeline for 3D scene graph generation from RGB-D video data was presented. The proposed method combines YOLO-based object detection and segmenation, hand tracking with MediaPipe, depth based 3D localization, temporal tracking and inference of spatial relationships to create structured representations of a scene.

The generated scene graphs were saved in a JSON-based format and they were visualized in a three-dimensional manner for analysis and verification. The results showed that the system was able to successfully detect and localize objects and hands, maintain consistent identities across multiple frames and infer meaningful spatial relationships between different entities in the scene. The generated scene graphs preserved both of the semantic and the spatial information while maintaining performance suitable for real-time applications simultaneously.

Although several limitations were identified, including misclassifications of some objects and the use of rule-based relational inference, the results show that it is possible to generate structured 3D scene graphs from RGB-D observations. Future work could focus on improved object detection, more advanced interaction-based relations, and learning-based methods for relational inference to further improve scene understanding.

## References


[1] Xu, D., Zhu, Y., Choy, C. B., and Fei-Fei, L., "Scene Graph Generation by Iterative Message Passing," *CVPR*, 2017.

[2] Krebs, F. et al., "The KIT Bimanual Manipulation Dataset," *IEEE-RAS International Conference on Humanoid Robots*, 2021.

[3] Lugaresi, C. et al., "MediaPipe: A Framework for Building Perception Pipelines," *CVPR*, 2019.

[4] Ultralytics, "YOLO26: Real-Time Object Detection and Segmentation Models," Ultralytics Documentation. [Online]. Available: https://docs.ultralytics.com/models/yolo26/. [Accessed: Jan. 2025].

[5] Cheng, H. K. et al., "XMem: Long-Term Video Object Segmentation with an Atkinson-Shiffrin Memory Model," *ECCV*, 2022.