Dynamic Scene Graphs for Action Recognition

Efe Ongan eongan@ethz.ch
Can Sacan csacan@ethz.ch
Nicolas Dickenmann ndickenmann@ethz.ch
Curtis Groth cgroth@ethz.ch

Abstract

Recent advances in three-dimensional scene understanding have been successful in creating representations of scenes that are efficient to use. However, these approaches rely on vision language models (VLMs) that are too inefficient to run in real-time. To address this, we propose a framework that allows real-time scene graph generation and action recognition. We collect RGB-D data from iPhones and HD-EPIC, augmented to create a dataset and establish a framework to generate spatial and dynamic scene graphs in real time. We illustrate the utility of our scene graphs in the action recognition problem using graph neural networks and match the performance of state-of-the-art vision language models. Our code can be found at https://github.com/3dvision-cecn/realtime_scene_understanding.

1. Introduction

Accurate, lightweight scene graphs have become a key representation for action-recognition and other downstream tasks. In this work, we present a framework designed to efficiently generate these scene graphs. Specifically, we represent hands and objects as nodes and connect them with distance edges in real time on consumer-grade hardware. Using Apple ARKit, we collected a 2 h egocentric RGB-D kitchen dataset. For real-time object detection and segmentation, we integrate YOLO [10] and EdgeTAM [38], respectively. In addition, we employ AM-RADIO [23] (combining DINOv2, CLIP, and SAM) and CLIP only [22] models to generate rich embeddings. We verify the effectiveness of our scene graphs using GNNs for action prediction.

Our contributions include:

1. A near real-time pipeline that fuses YOLO-EdgeTAM segmentation, ViT-Pose hand tracking and AM-RADIO embeddings into a 3D dynamic scene graph executable at ≤ 1 Hz on consumer GPUs.
2. Designing and benchmarking a heterogeneous GNN that jointly reasons over spatial and temporal edges and matches or surpasses state-of-the-art video transformers on action recognition with fewer trainable parameters.
3. A 2 hour egocentric RGB-D kitchen dataset with synchronized LiDAR depth and camera poses recorded.

2. Related work

2.1. 3D scene graphs

Our work builds on recent advances in 3D scene reconstruction. Past works innovate on creating 3D scene understanding[6], recognition of objects[1], and the creation of a scene graph based on these objects[26, 35].
Most relevant to our work are ConceptGraphs[9] and HOV-SG[29]. ConceptGraphs focuses on building an open-vocabulary 3D scene graph and recognizing actions using a vision language model (VLM), while HOV-SG introduces an open-vocabulary hierarchical structure to semantic storing of graph relations, as well as a metric for measuring semantic accuracy of object relations. Although these works obtain impressive results in the construction of scene graphs, they require a level of processing time that is prohibitive to real-time deployment.

2.2. Graph Neural Networks

[13, 25, 31] demonstrate the state-of-the-art of temporal graph neural networks. These networks learn on dynamic graphs with respect to time, learning on the structure between nodes as dynamic graphs update, and passing updates through time. [7, 11, 15, 28, 36] additionally demonstrate the usage of graph neural networks in AR settings, focusing on applying the learning of temporal relations in graph structures to human action recognition in datasets.

2.3. Depth Estimation and Depth Completion

Depth estimation is an important factor in properly recognizing objects and geometric relations. [12, 16, 32] cover the state-of-the-art in foundation model depth estimation and allow for accurate depth information to be gained in absence of a ground truth. This is important in egocentric video where reliable depth data is hard to acquire, or when lidar measurements such as those from ARKit are noisy or unreliable.

3. Dataset

In literature, there are multiple datasets like EK-100[4] and EGO-4D[8], which include egocentric monocular observations for action recognition and understanding tasks. All of them suffer from one of two shortcomings, either lacking camera metadata (EK-100) or consistent data collection for proper spatial understanding. EGO-4D is partially recorded with the Aria glasses (more in section 3.2). Both datasets also lack the depth data required to capture spatial understanding of the scene. One might argue that depth is not required by spatial understanding and monocular camera views are enough to infer the scene, like AVION[37]. This is one of the fundamental questions this project aims to answer. For these reasons we are going to mainly focus on two types of datasets.

3.1. iPhone Dataset

First is an iPhone dataset captured in kitchen scenes with an egocentric-looking iPhone with LiDAR depth data and internal Visual-Inertial odometry provided by ARKit. The recordings are done with the application Record3D[24]. The egocentric data-recording setup can be seen in 1. This configuration allowed us to record a high quality dataset with depth and camera trajectory information. The included camera and sensors such as microphones are properly time synchronised and do not require any specialist hardware. This low-cost setup might facilitate large-scale data collection in the future which is required by downstream applications like robotics.

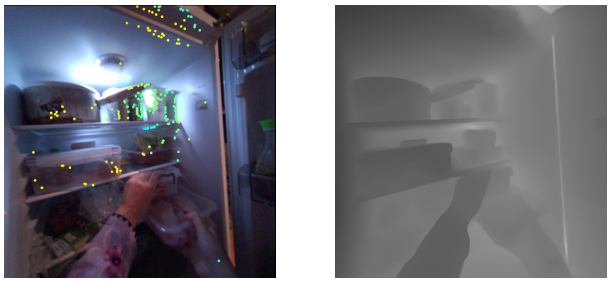
Figure 1. Initial iPhone data recording setup. Later improved with the purchase of a proper head attachment kit.

3.2. HD-EPIC Dataset

HD-EPIC[18] includes kitchen scenes recorded via Aria (Gen-1) glasses and data processing provided by the Aria-MPS[5] services. The MPS provides undistortion of the lenses and SLAM trajectory of the device, but does not include depth data. In order to augment the capture with depth data, three main options were investigated.

1. The use of monocular depth estimation methods like ML-DepthPro[2] and DepthAnythingV2[33]. Both methods were unable to provide consistent scale and disambiguation of hand and background objects. This might be attributed to their training datasets, which include static and non-egocentric scenes.
2. Stereo depth matching with SLAM cameras included in the Aria-glasses. But due to the non-overlapping FOV, there are not enough visual features to be matched. But in the future with Aria-glasses(Gen-2) with overlapping front-facing SLAM cameras, this will become possible.
3. Methods like [34], which process sequences of images for temporal cues. It is also disregarded because most of the actions consist of stationary camera and moving hands.
4. Depth completion methods like OMNI-DC[39] which promises aligned depth data. This method has been applied in the end in conjunction with SLAM features as sparse inputs.

OMNI-DC[39], trained on multiple datasets including sparse LiDAR inputs or SLAM features, is a promising direction to achieve consistent and world-aligned monocular depth estimation. For HD-EPIC, it has been applied with projected SLAM features from side cameras to the front RGB camera. This introduces visibility issues due to different perspectives, but the data provided by Aria-glasses only includes semidense-SLAM observations from side cameras, and a compromise was taken. The results can be seen in Figure 2. One chronic failure case of this method happens



(a) Sparse Depth From SLAM Cameras

(b) Depth Estimate From OMNI-DC

Figure 2. Sparse Depth and Completed Depth

when SLAM keypoints become occluded by dynamic objects like a hand. The depth-completion in this area subsequently becomes ambiguous. Nonetheless, the use of sparse keypoints for egocentric depth estimation provides an interesting avenue for future research and thus has been selected as our main method for data processing with HD-EPIC. This choice added an additional 700ms of processing time per frame (see 1) but is mainly intended for training purposes and not for real-time usage at inference.

3.3. Dataset Labeling

We use the AVION [37] model to label the iPhone and the HD-EPIC dataset as ground truth. This approach allows for fast and cheap labeling. We get one label consisting of a

verb and a noun for each 16 frames. The pool of labels is limited to the labels that avion was trained on which includes 97 verbs and 300 nouns. This choice was mainly dictated by the amount of time required to annotate large number of sequences.

4. Method - Graph Generation Pipeline

The graph generation pipeline has been structured to minimise the processing time required and still be able to fit into a consumer-grade GPU. An example frame processed can be found in fig 3

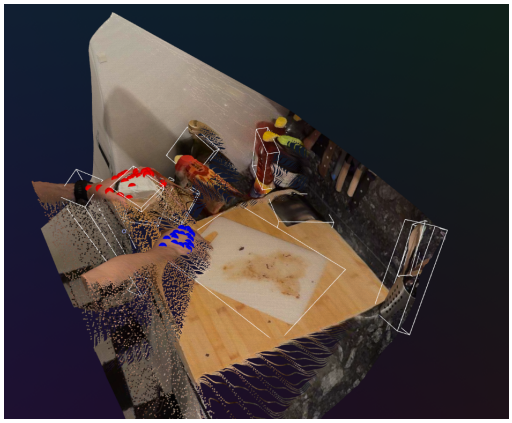


Figure 3. Visualization of all modules working together on the 3D iPhone dataset.

4.1. Hand detection and tracking

The hand detection pipeline is inspired by [17]. It consist of two stages, first using a Cascade-RCNN[3] model to detect the person in the image. Then, using ViT-Pose[30] with the person bounding boxes for hand keypoint detection. Finally, the keypoints are projected to 3D by using a pixel-aligned point cloud for obtaining hand positions. In the future, more fine-grained keypoint structures, including the wrist and arms, can also be utilised. Another aspect that can be utilised in the future is more fine-grained hand structures consisting of multiple points instead of only using the mean position of all keypoints. The detection keypoints can be seen in 4.

4.2. Object Segmentation - Detection

During experimentation, multiple segmentation/detection methods have been investigated in terms of their detection accuracy, speed under the condition of not being limited to a small set of predetermined objects. The following two approaches were the most promising ones evaluated:

1. Sampling a grid of points and using EdgeTAM[38] to generate segmentations. This required at least 32 * 32 points which translated into 800ms per frame. Due to high processing time, this option was disregarded. But it produced the most diverse and accurate segmentations.
2. Using YOLO-V11 [10] for obtaining bounding boxes of candidate objects and initializing EdgeTAM [38] from the centers of these bounding boxes. This reduces the amount of points so that all of them can be processed in one batch and leads to a combined 95 ms processing time.

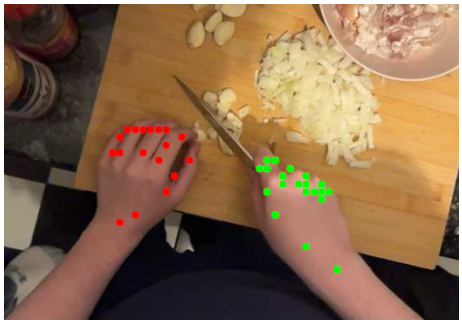


Figure 4. Hand detection from ViT-Pose.

The second method allows for object detection and segmentation in an acceptable time period. The main limitation of this stage is the inability to detect objects if they are heavily occluded by the hand. Due to the close proximity of the hand and interacting objects it may happen that either YOLO is unable provide candidates, or two objects are reduced into one segmentation mask. A purpose-built system for understanding hand-object interactions might be necessary to mitigate this limitation in the future. An example output of this stage can be seen in fig 5.

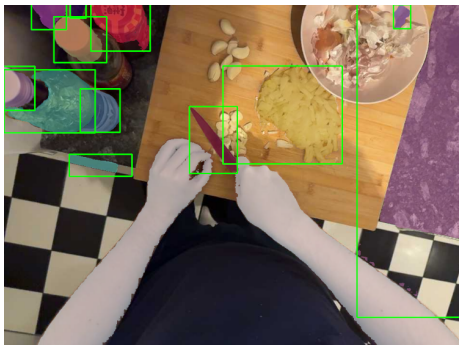


Figure 5. Segmentation from YOLO + EdgeTAM, hands filtered.

4.3. Hand-Object Node Embedding Generation

Hand-object embedding generation inspired by HOV-SG[29] and uses the fusion of three different embeddings generated from a crop of the object, including the background, a crop centering the image, and a masked embedding. These three embeddings are combined by taking the average sum of the features. An example can be found in 6. For ablation studies using CLIP text features, object labels

have been generated from smolVLM [14], provided with objects circled in red, and asked to answer the object label inside. The use of a VLM imposes an additional 2.0 seconds processing time per frame.

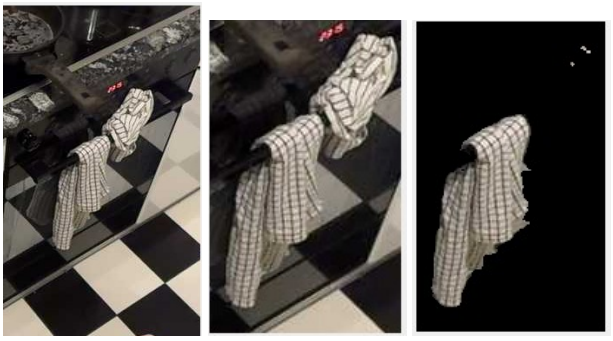


Figure 6. Three object views - global, local and background-removed are embedded with AM-RADIO / CLIP then averaged and concatenated with the object's 3-D center; hands use one view plus their averaged 3-D position.

4.4. Graph generation

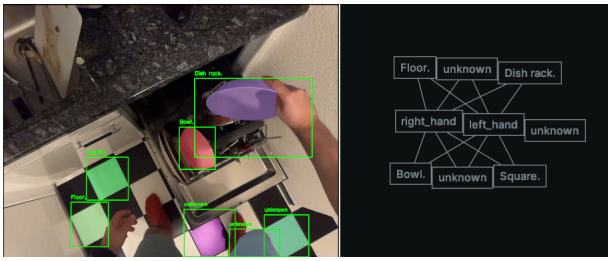


Figure 7. Example of a per frame graph instance with concatenated CLIP text + image features.

We generate our scene graph by connecting all objects to both hands. The nodes contain semantic feature vectors (e.g. from AM-RADIO), 3D positions, and hand identity labels (feature dim. + 3 + 1 dim.). The edges encode the relative 3D vector from each object to the hand and its absolute distance (3 + 1 dim.). We generate the semantic feature vectors following 4.3.

4.5. Final Pipeline Time Analysis

The time analysis of the pipeline in Table 1 demonstrates that more optimizations are necessary in hand detection and embedding generation stages.

Stage	Avg. Processing Time
Depth Completion (HD-EPIC)	700 ms
Hand Detection	291 ms
Object Segmentation – Detection	95 ms
Embedding Generation (AM-RADIO)	414 ms
Graph Generation	2 ms
Average per frame iPhone	**866 ms**

Table 1. Average processing time per stage averaged over 100 frames. All processing times for iPhone recording, except depth completion. All values are reported from a system with Intel 14900k and Nvidia RTX4090.

5. Method - Graph Architecture and Training

5.1. Graph Neural Network

We model each scene as a heterogeneous graph whose nodes are hands or objects and whose edges capture two relation types: spatial (intra-frame distance) and temporal (inter-frame hand tracking). Both edge types are processed with the same Graph Attention Convolution (GATConv) operator wrapped in `HeteroConv`; the only difference is that we instantiate two separate GATConv layers (one per relation type) so the model can learn distinct attention weights for spatial and temporal contexts [20, 27]. Before message passing, a node-MLP projects visual features + 3-D positions into a shared d-dimensional space, while an edge-MLP embeds the distance + relative offset into the same space, enabling edge-aware attention [19].

$$\mathbf{h}_i^{(\text{rel})} = \sum_{j \in \mathcal{N}^{\text{rel}}(i)} \alpha_{ij}^{\text{rel}} \, \mathbf{W}\mathbf{h}_j, \quad \mathbf{h}_i^{(\text{temp})} = \sum_{j \in \mathcal{N}^{\text{temp}}(i)} \alpha_{ij}^{\text{temp}} \, \mathbf{W}\mathbf{h}_j \tag{1}$$

$$\mathbf{h}_i^{\text{new}} = \mathbf{h}_i^{(\text{rel})} + \mathbf{h}_i^{(\text{temp})}, \tag{2}$$

where α_{ij} represents learned attention weights conditioned on the corresponding edge embedding.

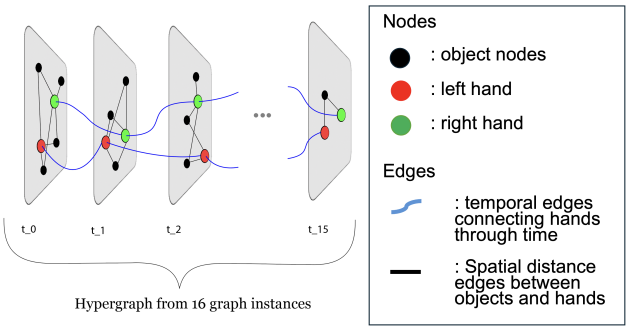


Figure 8. A visualization of our hypergraph. Nodes and edges are labeled in the legend.

5.2. Network Architecture

Figure 8 illustrates the resulting hypergraph. The pipeline is as follows:

1. Node and edge features are embedded by their respective MLPs.
2. Two GATConv layers, one for *relation* edges and one for *temporal* edges, which are executed inside a `HeteroConv` block, and their outputs summed per node.
3. We apply **global mean pooling** over all object nodes to obtain a scene-level graph embedding.
4. A 2-layer MLP (128 $\rightarrow$ out_dim) maps this graph level embedding to verb–noun action logits.

This design keeps the model lightweight yet expressive enough to fuse spatial layout and temporal dynamics.

5.3. Augmentation Strategies During Training

During training of the GNN, we add dropout (0.3) after the first layer of the MLP that transforms edge features to a common representation space with nodes (512 dim.). The same dropout is applied after the second layer of the MLP for node embeddings.

6. Results

Dataset	Duration
iPhone	2h
HD-EPIC	10h

Table 2. The Amount of Data Processed from Both Datasets

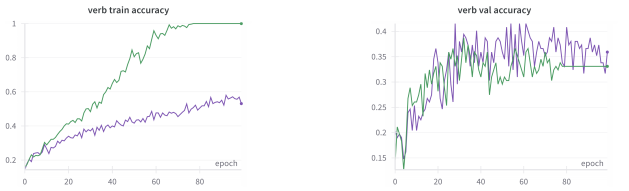
6.1. Comparison of Semantic Embeddings

We evaluate object and hand embeddings by comparing CLIP and AM-RADIO features from cropped hand and object images (Fig. 6). These experiments were conducted on our custom-collected iPhone dataset of 700 space-time scene graphs (see 8). We generate CLIP embeddings for both image and text (each 512-D), and compare CLIP-text, CLIP-text+image (concatenated to 1024-D), and AM-RADIO image embeddings (3072-D). Action classification results and verb noun accuracies are reported in Table 3. We observe that clip-text embeddings alone are not well suited for object representation in the wild. Concatenation with clip-image features further increases overall accuracies by 10%. AM-RADIO features further increase representational capacity and classification accuracy by another 10%.

6.2. Comparison to Video Transformer Methods

We compare our method to the state of the art method on action recognition, AVION[37]. We train the AVION method from scratch on the same data split as the experiments run in Section 6.1. We use a pretrained model on the OpenAI ViT-B/16[21] and train on the training split of our iPhone dataset for 100 epochs. We implemented our own version of AVION and we differentiate between a default version which over-fits quickly and a regularized version that uses dropout, weight decay and data augmentation for increased generalization (see Figure 9). The results of these experiments are shown in Table 3.

Future work includes comparing our architecture to state-of-the-art multimodal models such as Gemini 2.5 Pro or OpenAI o3.



(a) Overfits easily without regularization

(b) AVION with regularization performs better

Figure 9. AVION w/out regularization techniques

6.3. Spatial Information

We analyze the effect of removing node positions from node features and only pass AM-RADIO features. The accuracy differences were negligable which means positional encodings in nodes have little to zero effect. Next, by removing edge positional information and passing empty values for edge attributes we observe 5-10 % drop across all metrics: Val Acc. @1 to **0.306** Val Verb Acc.@1 to **0.408** and Val Noun Acc.@1 to **0.385**. Compare to AM-RADIO + GNN row on 3.

6.4. Generalizability

An UMAP clustering of the graph embeddings in Figure 10 shows separation between data-points processed from our two different datasets. Our iPhone dataset consists of a single kitchen, while also our processed HD-EPIC dataset only consists of two kitchens.

This distribution shift can be observed during validation, if the training split consists only of data from HD-Epic and validation only from our iPhone dataset. This puts into doubt how well the use of AM-RADIO embeddings generalizes to different environments. Currently, our method does not exhibit strong generalization, but further experimentation is necessary in a more diverse dataset to give a conclusive answer. We completed a further ablation study without the spatial features of the depth information to pinpoint the problem within the visual features. The results can be found in the table below 4. For both cases the highest @1 accuracy is taken during 50 epochs of training.

	Feat Dim.	Val Loss	Val Acc.@1	Val Acc.@5	Val Verb acc.@1	Val Noun acc.@1
CLIP text only + GNN	512	5.849	0.176	0.345	0.301	0.246
CLIP text & image + GNN	1024	5.764	0.277	0.445	0.401	0.336
AM-RADIO + GNN	3072	4.819	**0.350**	**0.562**	**0.445**	**0.445**
AVION		4.831	0.275	0.409	0.352	0.359
AVION regularized		**4.487**	0.296	0.465	0.409	0.345
	——	Train Loss	Train Acc.@1	Train Acc.@5	Train Verb acc.@1	Train Noun acc.@1
CLIP text only + GNN		3.528	0.182	0.415	0.314	0.233
CLIP text & image + GNN		2.544	0.346	0.605	0.458	0.371
AM-RADIO + GNN		1.695	**0.489**	**0.830**	**0.562**	**0.524**
AVION		2.579	0.322	0.635	0.502	0.355
AVION regularized		2.807	0.293	0.569	0.436	0.327

Table 3. Comparison of different versions of our GNN based method with AVION. All approaches are trained on the same random 80/20 validation split on our iPhone dataset for 100 epochs. The best epoch (highest val acc. @1) is reported.

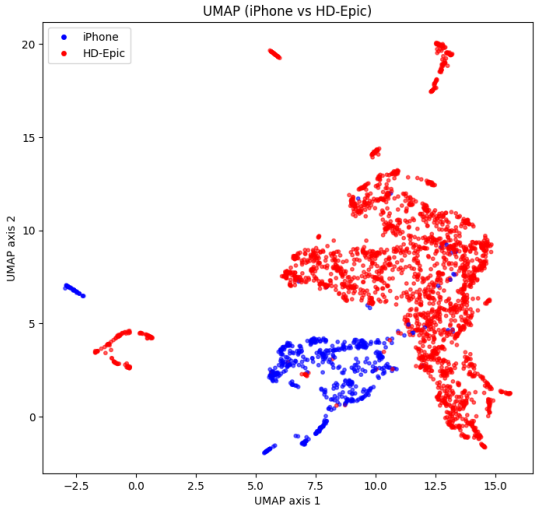


Figure 10. UMAP visualization of HD-Epic and iPhone graph embeddings. Red from HD-EPIC and blue from iPhone

	Loss	Acc. @1	Acc. @5	Verb acc. @1	Noun acc. @1
Train w Spatial	2.422	0.324	0.683	0.427	0.369
Test w Spatial	7.910	0.128	.271	0.241	0.160
Train w/o Spatial	3.505	0.198	0.476	0.279	0.228
Test w/o Spatial	6.225	0.067	0.221	0.186	0.100

Table 4. HD-EPIC training and iPhone validation generalizability experiments

7. Discussion

Table 3 shows that our model surpasses the transformer-based action classifier AVION, which operates directly on raw images. Extracting structured features with our image-understanding pipeline gives the downstream graph network richer information and clear gains in the low-data regime. It is still unclear how well this advantage will continue when the data is scaled, where the heuristics included during the image understanding pipeline might hinder further improvement.

Comparing feature types 3 reveals that assigning text labels to objects with a VLM does not provide accuracy gains. The highest scores come from image-only AM-RADIO embeddings, which also outperform CLIP image embeddings. These results show that distilling multiple features provides the best backbone for action recognition.

Dropping absolute node positions in 3D leaves performance unchanged, but removing the relative edge offsets (relative distance + directional vector) costs five to ten points, so geometry encoded as pairwise distances is the more important scene-geometry information 6.3.

The ablation study in Table 4 shows that spatial information is relevant and leads to better results. But the same evaluation and separated UMAP embeddings 10 displays a fundamental domain gap between the augmented HD-EPIC dataset and iPhone. Further evaluation studies and a more diverse iPhone dataset (multiple kitchens) are required to understand how spatial information and graph features can be scaled.

8. Conclusion and Future Work

Our approach for AR using scene graphs and GNNs represents a near real-time method that can run on consumer hardware and match the performance of state-of-the-art pre-trained video transformers on a self-collected dataset.
Further improvements in image feature extractors (tokenizers, e.g. CLIP, AM-RADIO) could lead to better performance. Furthermore, experiments comparing our method to pre-trained zero-shot multimodal models could further verify our approach.
Scaling experiments should be performed on a larger dataset to understand how well the method will generalize.

9. Work distribution

Efe was instrumental in developing the method for retrieving depth layers as well as hand tracking; additionally, he ran our distributed experiments. Can was responsible for feature selection, the GNN architecture and ablation studies. Nicolas worked on the real-time object detection and segmentation as well as running the ablation experiments with avion. Curtis contributed by implementing HAMER and doing literature reviews.

References

[1] Tjark Behrens, René Zurbrügg, Marc Pollefeys, Zuria Bauer, and Hermann Blum. Lost & Found: Tracking changes from egocentric observations in 3d dynamic scene graphs. *IEEE Robotics and Automation Letters*, pages 1–8, 2025. 1

[2] Aleksei Bochkovskii, Amaël Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan R. Richter, and Vladlen Koltun. Depth pro: Sharp monocular metric depth in less than a second, 2025. 2

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection, 2017. 3

[4] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The epic-kitchens dataset. In *European Conference on Computer Vision (ECCV)*, 2018. 2

[5] Jakob Engel, Kiran Somasundaram, Michael Goesele, Albert Sun, Alexander Gamino, Andrew Turner, Arjang Talattof, Arnie Yuan, Bilal Souti, Brighid Meredith, Cheng Peng, Chris Sweeney, Cole Wilson, Dan Barnes, Daniel DeTone, David Caruso, Derek Valleroy, Dinesh Ginjupalli, Duncan Frost, Edward Miller, Elias Mueggler, Evgeniy Oleinik, Fan Zhang, Guruprasad Somasundaram, Gustavo Solaira, Harry Lanaras, Henry Howard-Jenkins, Huixuan Tang, Hyo Jin Kim, Jaime Rivera, Ji Luo, Jing Dong, Julian Straub, Kevin Bailey, Kevin Eckenhoff, Lingni Ma, Luis Pesqueira, Mark Schwesinger, Maurizio Monge, Nan Yang, Nick Charron, Nikhil Raina, Omkar Parkhi, Peter Borschowa, Pierre Moulon, Prince Gupta, Raul Mur-Artal, Robbie Pennington, Sachin Kulkarni, Sagar Miglani, Santosh Gondi, Saransh Solanki, Sean Diener, Shangyi Cheng, Simon Green, Steve Saarinen, Suvam Patra, Tassos Mourikis, Thomas Whelan, Tripti Singh, Vasileios Balntas, Vijay Baiyya, Wilson Dreewes, Xiaqing Pan, Yang Lou, Yipu Zhao, Yusuf Mansour, Yuyang Zou, Zhaoyang Lv, Zijian Wang, Mingfei Yan, Carl Ren, Renzo De Nardi, and Richard Newcombe. Project aria: A new tool for egocentric multi-modal ai research, 2023. 2

[6] Tim Engelbracht, René Zurbrügg, Marc Pollefeys, Hermann Blum, and Zuria Bauer. Spotlight: Robotic scene understanding through interaction and affordance detection, 2024. 1

[7] Tiantian Geng, Feng Zheng, Xiaorong Hou, Ke Lu, Guo-Jun Qi, and Ling Shao. Spatial-temporal pyramid graph reasoning for action recognition. *IEEE Transactions on Image Processing*, 31:5484–5497, 2022. 1

[8] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, Eugene Byrne, Zach Chavis, Joya Chen, Feng Cheng, Fu-Jen Chu, Sean Crane, Avijit Dasgupta, Jing Dong, Maria Escobar, Cristhian Forigua, Abrham Gebreselasie, Sanjay Haresh, Jing Huang, Md Mohaiminul Islam, Suyog Jain, Rawal Khirodkar, Devansh Kukreja, Kevin J Liang, Jia-Wei Liu, Sagnik Majumder, Yongsen Mao, Miguel Martin, Effrosyni Mavroudi, Tushar Nagarajan, Francesco Ragusa, Santhosh Kumar Ramakrishnan, Luigi Seminara, Arjun Somayazulu, Yale Song, Shan Su, Zihui Xue, Edward Zhang, Jinxu Zhang, Angela Castillo, Changan Chen, Xinzhu Fu, Ryosuke Furuta, Cristina Gonzalez, Prince Gupta, Jiabo Hu, Yifei Huang, Yiming Huang, Weslie Khoo, Anush Kumar, Robert Kuo, Sach Lakhavani, Miao Liu, Mi Luo, Zhengyi Luo, Brighid Meredith, Austin Miller, Oluwatumininu Oguntola, Xiaqing Pan, Penny Peng, Shraman Pramanick, Merey Ramazanova, Fiona Ryan, Wei Shan, Kiran Somasundaram, Chenan Song, Audrey Southerland, Masatoshi Tateno, Huiyu Wang, Yuchen Wang, Takuma Yagi, Mingfei Yan, Xitong Yang, Zecheng Yu, Shengxin Cindy Zha, Chen Zhao, Ziwei Zhao, Zhifan Zhu, Jeff Zhuo, Pablo Arbelaez, Gedas Bertasius, David Crandall, Dima Damen, Jakob Engel, Giovanni Maria Farinella, Antonino Furnari, Bernard Ghanem, Judy Hoffman, C. V. Jawahar, Richard Newcombe, Hyun Soo Park, James M. Rehg, Yoichi Sato, Manolis Savva, Jianbo Shi, Mike Zheng Shou, and Michael Wray. Ego-exo4d: Understanding skilled human activity from first- and third-person perspectives, 2024. 2

[9] Qiao Gu, Ali Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, et al. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5021–5028. IEEE, 2024. 1

[10] Glenn Jocher and Jing Qiu. Ultralytics yolo11, 2024. 1, 3

[11] Yiming Li, Xiaoshan Yang, and Changsheng Xu. Dynamic scene graph generation via anticipatory pre-training. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13864–13873, 2022. 1

[12] Haotong Lin, Sida Peng, Jingxiao Chen, Songyou Peng, Jiaming Sun, Minghuan Liu, Hujun Bao, Jiashi Feng, Xiaowei Zhou, and Bingyi Kang. Prompting depth anything for 4k resolution accurate metric depth estimation. 2024. 1

[13] Antonio Longa, Veronica Lachi, Gabriele Santin, Monica Bianchini, Bruno Lepri, Pietro Lio, Franco Scarselli, and Andrea Passerini. Graph neural networks for temporal graphs: State of the art, open challenges, and opportunities, 2023. 1

[14] Andrés Marafioti, Orr Zohar, Miquel Farré, Merve Noyan, Elie Bakouch, Pedro Cuenca, Cyril Zakka, Loubna Ben Allal, Anton Lozhkov, Nouamane Tazi, Vaibhav Srivastav, Joshua Lochner, Hugo Larcher, Mathieu Morlon, Lewis Tunstall, Leandro von Werra, and Thomas Wolf. Smolvlm: Redefining small and efficient multimodal models, 2025. 4

[15] Andrei Nicolicioiu, Iulia Duta, and Marius Leordeanu. Recurrent space-time graph neural networks, 2019. 1

[16] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. 1

[17] Georgios Pavlakos, Dandan Shan, Ilija Radosavovic, Angjoo Kanazawa, David Fouhey, and Jitendra Malik. Reconstructing hands in 3d with transformers, 2023. 3

[18] Toby Perrett, Ahmad Darkhalil, Saptarshi Sinha, Omar Emara, Sam Pollard, Kranti Parida, Kaiting Liu, Prajwal Gatti, Siddhant Bansal, Kevin Flanagan, Jacob Chalk, Zhifan Zhu, Rhodri Guerrier, Fahd Abdelazim, Bin Zhu, Davide Moltisanti, Michael Wray, Hazel Doughty, and Dima Damen. Hd-epic: A highly-detailed egocentric video dataset, 2025. 2

[19] PyTorch Geometric Team. Edge features in graph attention – `edge_dim` option, 2025. Accessed: 2025-06-23. 4

[20] PyTorch Geometric Team. torch_geometric.nn.conv.gatconv, 2025. Accessed: 2025-06-23. 4

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 5

[22] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *arXiv preprint*, arXiv:2103.00020, 2021. v1, accessed 23 Jun 2025. 1

[23] Mike Ranzinger, Greg Heinrich, Jan Kautz, and Pavlo Molchanov. Am-radio: agglomerative vision foundation model – reduce all domains into one. *arXiv preprint arXiv:2312.06709*, 2023. Accepted at CVPR 2024. 1

[24] Record3d. Record3d, 2025. https://record3d.app/ [Accessed: 2025-06-22]. 2

[25] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs, 2020. 1

[26] Jonas Schult, Francis Engelmann, Alexander Hermans, Or Litany, Siyu Tang, and Bastian Leibe. Mask3d: Mask transformer for 3d semantic instance segmentation, 2023. 1

[27] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *Proc. International Conference on Learning Representations (ICLR)*, 2018. 4

[28] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs, 2018. 1

[29] Abdelrhman Werby, Chenguang Huang, Martin Büchner, Abhinav Valada, and Wolfram Burgard. Hierarchical open-vocabulary 3d scene graphs for language-grounded robot navigation. *Robotics: Science and Systems*, 2024. 1, 3

[30] Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. Vitpose: Simple vision transformer baselines for human pose estimation, 2022. 3

[31] Yuanyuan Xu, Wenjie Zhang, Ying Zhang, Maria Orlowska, and Xuemin Lin. Timesgn: Scalable and effective temporal graph neural network. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 3297–3310, 2024. 1

[32] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *arXiv:2406.09414*, 2024. 1

[33] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2, 2024. 2

[34] Chengbo Yuan, Geng Chen, Li Yi, and Yang Gao. Self-supervised monocular 4d scene reconstruction for egocentric videos. *arXiv preprint arXiv:2411.09145*, 2024. 2

[35] Chenyangguang Zhang, Alexandros Delitzas, Fangjinhua Wang, Ruida Zhang, Xiangyang Ji, Marc Pollefeys, and Francis Engelmann. Open-vocabulary functional 3d scene graphs for real-world indoor spaces, 2025. 1

[36] Jingran Zhang, Fumin Shen, Xing Xu, and Heng Tao Shen. Temporal reasoning graph for activity recognition. *IEEE Transactions on Image Processing*, 29:5491–5506, 2020. 1

[37] Yue Zhao and Philipp Krähenbühl. Training a large video model on a single machine in a day. *arXiv preprint arXiv:2309.16669*, 2023. 2, 5

[38] Chong Zhou, Chenchen Zhu, Yunyang Xiong, Saksham Suri, Fanyi Xiao, Lemeng Wu, Raghuraman Krishnamoorthi, Bo Dai, Chen Change Loy, Vikas Chandra, and Bilge Soran. Edgetam: On-device track anything model, 2025. 1, 3

[39] Yiming Zuo, Willow Yang, Zeyu Ma, and Jia Deng. Omnidc: Highly robust depth completion with multiresolution depth integration, 2024. 2