

– Supplementary Material –

“Where am I?” Scene Retrieval with Language

Jiaqi Chen¹, Daniel Barath¹, Iro Armeni²,
Marc Pollefeys^{1,3}, and Hermann Blum^{1,4}

¹ ETH Zürich

² Stanford University

³ Microsoft Mixed Reality & AI Lab, Zürich

⁴ University of Bonn

In the supplementary material, we show more details regarding our data, method, and results. More specifically, we first show more examples and details of scene graphs and text-graphs. We then show examples from the process of transforming a text description into a text-graph. Additionally, we show an investigation regarding the robustness of our embeddings when different input pairs are given to the joint embedding model. Lastly, we show qualitative results of which scenes were matched with which text descriptions.

A Scene Graph and Text-Graph Examples

A.1 ScanScribe Dataset Example

In Figure 1 we show a detailed labeled example of a 3D scene graph from the 3DSSG [3] dataset. We list the nodes and edges on the right hand side. The purple text describes the node attributes. The scene graph on the bottom left is labeled with the corresponding object IDs. We prune the edges of our scene graph based on a distance threshold, set to 1.5m. We also *only* consider edges that represent spatial relationships such as ‘left,’ ‘right,’ ‘behind,’ ‘close by,’ etc. This means we also prune any edges that are labeled with the following: ‘bigger than,’ ‘smaller than,’ ‘higher than,’ ‘lower than,’ ‘same symmetry as,’ ‘same as,’ ‘same color,’ ‘same material,’ ‘same texture,’ ‘same shape,’ ‘same state,’ ‘same object type.’ This scene graph also corresponds with the text-graph from Figure 2.

In Figure 2 we show a detailed labeled example of a text-graph that corresponds with the scene graph in Figure 1. We correspond the yellow object ID labels with the labels shown in the scene image on top. The bottom description also highlights the text-graph objects in yellow. We point out that the text-graph is a subset of the scene graph.

A.2 Human Dataset Example

In Figure 3 we show a detailed labeled example of a 3D scene graph from the 3DSSG [3] dataset. We list the nodes and edges on the right hand side. Not all nodes on the right hand side are labeled in the scene graph due to visibility. The purple text describes the node attributes. The scene graph on the bottom left

is labeled with the corresponding object IDs. We prune the edges of our scene graph based on a distance threshold, set to 1.5m. We also only consider edges that represent spatial relationships such as ‘left,’ ‘right,’ ‘behind,’ ‘close by,’ etc. This means we also prune any edges that are labeled with the following: ‘bigger than,’ ‘smaller than,’ ‘higher than,’ ‘lower than,’ ‘same symmetry as,’ ‘same as,’ ‘same color,’ ‘same material,’ ‘same texture,’ ‘same shape,’ ‘same state,’ ‘same object type.’ This scene graph also corresponds with the text-graph from Figure 4.

In Figure 4 we show a detailed labeled example of a text-graph, which corresponds with the scene graph in Figure 3. We correspond the yellow object ID labels with the labels shown in the scene image on top. The bottom description also highlights the text-graph objects in yellow. We point out that the text-graph is a subset of the scene graph. Furthermore, not all objects are included in the text-graph, and there are also additional objects identified by the human generated description that are not in the scene graph. Nevertheless, certain objects correspond between the text-graph and scene graph, as do their attributes, for example the “gray sofa.”

B Processing Text to Text-Graph Examples

In this section, we describe details about the process of transforming a text-query into a “text-graph.” We begin with a code snippet in Figure 5 that shows the GPT instructions that we give to GPT-4 [1]. In addition to the prompt, we also give special instructions to the system such that the resulting outputs are more consistent. We use this prompt and template for every text-query. Next, in Figures 6 and 7 we show examples of text-queries and their corresponding JSON “text-graph” representations, which are created by GPT.

B.1 GPT prompt

We give the template, as seen in Figure 5, to GPT to ensure more consistent outputs. Additionally, we found that giving system instructions also gave more consistent and higher quality outputs. We specifically limit the “label” to being one-word because occasionally the language model would output attributes as part of the label. During data generation, we found that GPT-4 [1] gave better outputs than GPT-3 [2], therefore we decide to use GPT-4. The outputs were better in that they more consistently captured the object labels and attributes correctly. Additionally, the outputs from GPT-4 followed the given template, whereas GPT-3 would sometimes include fewer or more fields.

B.2 Input Text and Output JSON Graph Pairs

From these input/output pairs in Figures 6 and 7, we can see that GPT captures the main objects described in the scene, their attributes, and their mentioned spatial relationships to other objects in the scene. Additionally, in the second example in Figure 6, there are two different blankets being referred to, and

the language model correctly captures both as separate blankets in the output. However, there are also instances where the translation does not necessarily work. From these figures, we see the text inputs being “translated” into a “text-graph.” It is worth noting that objects, attributes, and relationships are only represented in the text-graph if they are mentioned in the description.

C Graph Embedding Robustness to Input Pairs

In Figures 8 and 9, we show how varying the paired inputs into the joint embedding model does not significantly change the identifiability of the resulting scene or text-graph embeddings. For Figure 8, we embed 5 scene graphs from different scenes with the same text-graph input, and the resulting scene graph embeddings are consistent per scene. In other words, the cosine similarity is the high only when the scene graph embedding is compared with itself, and low everywhere else. Similarly for Figure 9, the text-graph embeddings are most similar to itself, and dissimilar from all other text-graph embeddings. Furthermore, we conducted an investigation where we maintain the same scene graph input, and change the text-graph paired input, and the resulting scene graph embeddings have covariances of 0, meaning the scene graph embeddings do not change that much with different paired inputs. We performed the same investigation for text-graphs and changed the paired scene graph input, the resulting text-graph embeddings also have covariances of 0.

All in all, these investigations demonstrate the robustness of the joint embedding model in generating representative embeddings that do not depend on the input pair. Even though our model requires a paired input (scene graph and text-graph) in order to generate an embedding for both, we show that we are also able to separately embed scene graph and text-graphs. This enables to use our joint embedding model in a retrieval-based manner, where we generate embeddings for all scene graphs, and then simply match an incoming text-query against the database of scene graph embeddings.

D Qualitative Results

In Figures 10, 11, 12, and 13, we show examples of qualitative results. For each dataset, we show 7 examples of successful matches and 4 examples of failure cases. The correct matches are highlighted with green borders. The text-query used is on the left-hand side. We order the three images on the right in order of matching score, with the left-most being the lowest or the best match. For the failure cases, if there are no images with green borders, this means the top three matches were all incorrect.

D.1 ScanScribe Dataset Results

From Figure 10, we see the top matching scenes for each description tend to have some similar aspects. For example in the first row, all top three matched

scenes are offices. In the sixth row, the top matching scenes are all bedroom scenes, and the top two both have a “wardrobe” which is described in the text description. Additionally, the best matching scene all have significantly lower matching scores (highlighted in yellow) than the next best match. Here the “matching score” is defined as $1 - \text{cosine_similarity}(a, b)$ where a and b are the scene graph and text-graph embedding vectors respectively, therefore a lower score means a better match.

In terms of the failure cases in Figure 11 for the ScanScribe dataset, we notice that the matching scores are much closer together than in the successful cases. Additionally, we notice that the incorrect matches mostly have objects in common with the description, signifying the network attended to the wrong objects. For example, in the second row of Figure 11, there are “blinds” in all three scenes.

D.2 Human Dataset Results

Figures 12 and 13 show qualitative results on the Human dataset. In Figure 12, from the first row, we see that the top two matches are both bathroom scenes. Additionally, the best two matches in the second row are both office scenes. The top two matches in rows four and six are both bedroom scenes. And the top two in row five are also both living room scenes. Even though the “matching scores” are not as far apart/differentiating when compared to the ScanScribe successful cases, our method is still able to distinguish between similar scenes. As for the failure cases in Figure 13, the matching scores are also very close, and objects in the description are also in the scene, but for example in the case of rows two, three, and four, the high number of chairs may be biasing the description to match with scenes with chairs.

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
2. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Nee-lakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., et al.: Language models are few-shot learners (2020)
3. Wald, J., Dhano, H., Navab, N., Tombari, F.: Learning 3d semantic scene graphs from 3d indoor reconstructions (2020)



Fig. 1: Detailed Example of Scene Graph We demonstrate a detailed label version of a 3D scene graph. The yellow labels on the nodes correspond to the object IDs listed on the right hand side. The purple text describes the attributes of the objects. The top left shows an image of the scene. The right hand side also shows a list of spatial relationships between the objects, however the list is not extensive and does not include relationships such as “same material as,” “same texture,” “same object type,” etc.

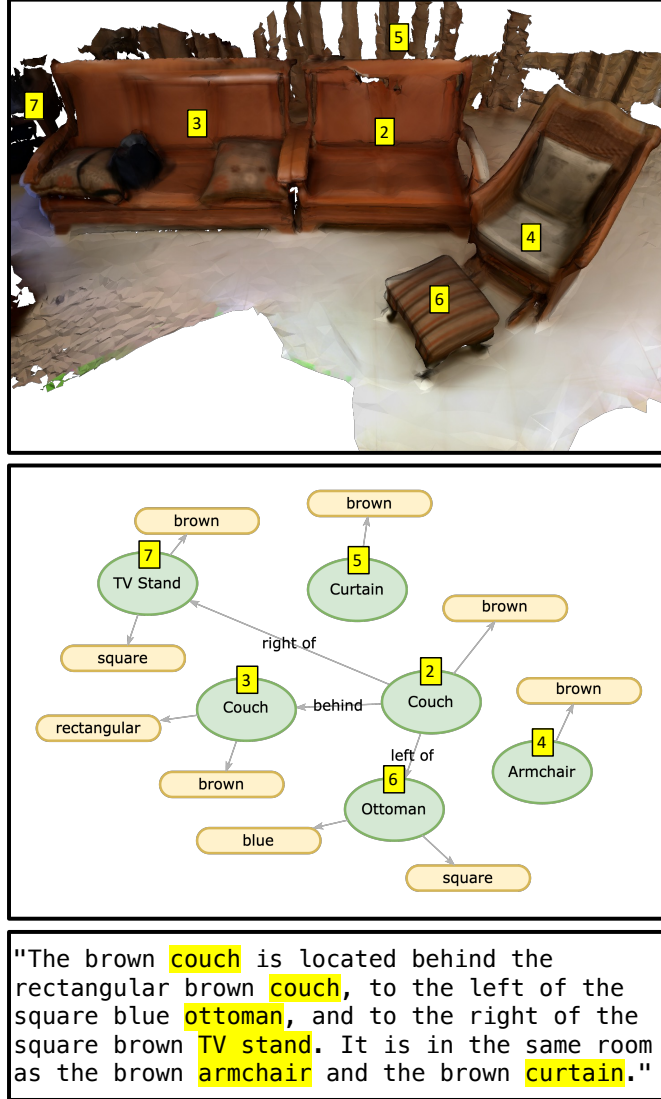
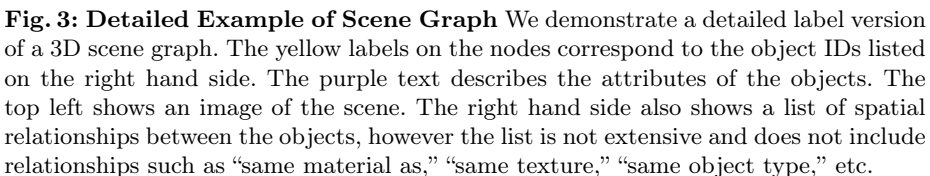


Fig. 2: Detailed Example of Text-Graph: ScanScribe Dataset We show a detailed label version of a text-graph in the middle image. The top image is an image of the scene. The bottom section is the text-query used to generate the text-graph. The text-query is from the ScanScribe dataset. The yellow labels on the nodes correspond with the yellow labeled object IDs in the scene image. The edges between objects in the text-graph represent spatial relations. If there is no edge between two objects, then there is no spatial relationship mentioned in the text description.



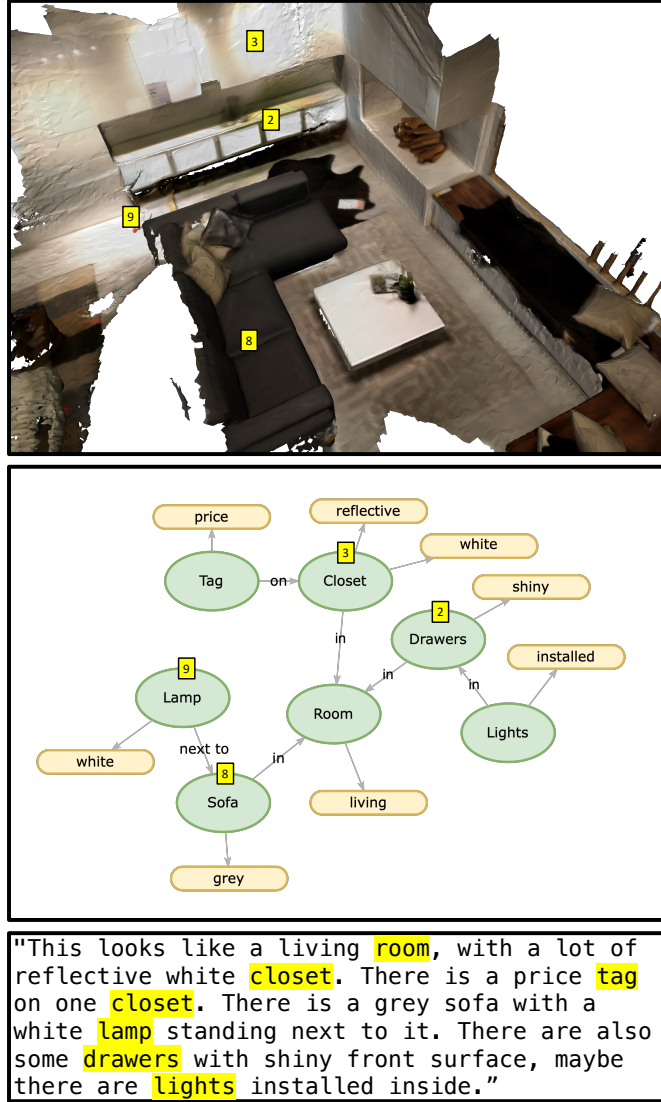


Fig. 4: Detailed Example of Text-Graph: Human Dataset We show a detailed label version of a text-graph in the middle image from the Human dataset. The top image is an image of the scene. The bottom section is the text-query. The yellow labels on the nodes correspond with the yellow labeled object IDs in the scene image. The edges between objects in the text-graph represent spatial relations. If there is no edge between two objects, then there is no spatial relationship mentioned in the text description. Additionally, there are objects in the text-graph that are not in the scene graph due to the natural human language description, such as “tag.” We also label the objects correspondingly if they are synonyms. For example, there is a “cabinet” in the 3D scene graph in Figure 3, but the text refers to it as a “closet.”

```

system_instructions = "Pretend you are an expert at translating a scene
description into a JSON representation of the scene where nodes are objects and
edges are spatial relationships between objects."

user_instructions = "You will only return valid JSON with the same format below,
given scene information by the user. The nodes are objects. The \"label\" child must
be nouns WITHOUT adjectives. The edges between the nodes are the spatial
relationships between objects. The \"attributes\" are adjectives or adverbs only and
do NOT describe spatial relations or positions. Follow the format below:

{
  \"nodes\":
  [
    { \"id\": \"1\",
      \"label\": \"<1 word noun>\",
      \"attributes\": [<insert adjectives>]
    },
  ],
  \"edges\":
  [
    { \"source\": \"\",
      \"target\": \"\",
      \"relationship\": \"\"
    }
  ]
}

The scene: <TEXT_QUERY>
The JSON response:"

# Replace substring "<TEXT_QUERY>" in user_instructions with our real text_query
user_instructions = user_instructions.replace("<TEXT_QUERY>", text)

completion = openai.ChatCompletion.create(
    model="gpt-4-1106-preview",
    response_format={ "type": "json_object" },
    messages=[
        {
            "role": "system",
            "content": system_instructions
        },
        {
            "role": "user",
            "content": user_instructions
        }
    ],
    temperature=0.10
)

```

Fig. 5: GPT Prompt. Here we show the prompt used to transform our text description into a json format graph. We instruct the language model to pretend to be a “scene description” translator that translates text descriptions into scene graphs. The model is also instructed to only return valid JSON formats, following a template provided. We replace the part of the prompt that says <TEXT_QUERY> with the text description that we want to transform into a text-graph. For examples regarding the input “text-query” and output “text-graph,” please refer to Figures 6 and 7.

"The trash can is located to the left of the blue chair, in front of the rectangular white table."	<pre>{ "nodes": [{ "id": "1", "label": "can", "attributes": ["trash"] }, { "id": "2", "label": "chair", "attributes": ["blue"] }, { "id": "3", "label": "table", "attributes": ["rectangular", "white"] }], "edges": [{ "source": "1", "target": "2", "relationship": "left of" }, { "source": "1", "target": "3", "relationship": "in front of" }] }</pre>
"The blanket is on the square brown bed, to the left of the rectangular white pillow and near another blanket."	<pre>{ "nodes": [{ "id": "1", "label": "blanket", "attributes": [] }, { "id": "2", "label": "bed", "attributes": ["square", "brown"] }, { "id": "3", "label": "pillow", "attributes": ["rectangular", "white"] }, { "id": "4", "label": "blanket", "attributes": [] }], "edges": [{ "source": "1", "target": "2", "relationship": "on" }, { "source": "1", "target": "3", "relationship": "left of" }, { "source": "1", "target": "4", "relationship": "near" }] }</pre>
"The brown couch is located behind the rectangular brown couch, to the left of the square blue ottoman, and to the right of the square brown TV stand. It is in the same room as the brown armchair and the brown curtain."	<pre>{ "nodes": [{ "id": "1", "label": "couch", "attributes": ["brown"] }, { "id": "2", "label": "couch", "attributes": ["brown", "rectangular"] }, { "id": "3", "label": "ottoman", "attributes": ["blue", "square"] }, { "id": "4", "label": "TV stand", "attributes": ["brown", "square"] }, { "id": "5", "label": "armchair", "attributes": ["brown"] }, { "id": "6", "label": "curtain", "attributes": ["brown"] }], "edges": [{ "source": "1", "target": "2", "relationship": "behind" }, { "source": "1", "target": "3", "relationship": "left of" }, { "source": "1", "target": "4", "relationship": "right of" }] }</pre>

Fig. 6: Input/Output Pairs from GPT for ScanScribe. The left hand side shows the input text-query that we give to GPT, along with the prompt in Figure 5. These are examples from the ScanScribe dataset. The right hand side shows the returned output from GPT: a JSON text representing a “text-graph.”

"This looks like a living room, with a lot of reflective white closet. There is a price tag on one closet. There is a grey sofa with a white lamp standing next to it. There are also some drawers with shiny front surface, maybe there are lights installed inside."	<pre>{ "nodes": [{ "id": "1", "label": "room", "attributes": ["living"] }, { "id": "2", "label": "closet", "attributes": ["reflective", "white"] }, { "id": "3", "label": "tag", "attributes": ["price"] }, { "id": "4", "label": "sofa", "attributes": ["grey"] }, { "id": "5", "label": "lamp", "attributes": ["white"] }, { "id": "6", "label": "drawers", "attributes": ["shiny"] }, { "id": "7", "label": "lights", "attributes": ["installed"] }], "edges": [{ "source": "2", "target": "1", "relationship": "in" }, { "source": "3", "target": "2", "relationship": "on" }, { "source": "4", "target": "1", "relationship": "in" }, { "source": "5", "target": "4", "relationship": "next to" }, { "source": "6", "target": "1", "relationship": "in" }, { "source": "7", "target": "6", "relationship": "inside" }] }</pre>
"A bed room with a soccer table and a TV left to the bed. There is also a shelf left to the TV."	<pre>{ "nodes": [{ "id": "1", "label": "room", "attributes": ["bed"] }, { "id": "2", "label": "table", "attributes": ["soccer"] }, { "id": "3", "label": "TV", "attributes": [] }, { "id": "4", "label": "shelf", "attributes": [] }], "edges": [{ "source": "3", "target": "1", "relationship": "left to" }, { "source": "2", "target": "1", "relationship": "in" }, { "source": "4", "target": "3", "relationship": "left to" }] }</pre>
"There is a large TV set with a large white TV cabinet. There is a black curtain hanging. There is also a large gray couch with some cushions and blanket on it. In front of the couch stands a large white coffee table. Besides, there are also several shelves and a lot of fancy decorations in this room. It is a large gorgeous living room and a little bit messy."	<pre>{ "nodes": [{ "id": "1", "label": "TV", "attributes": ["large"] }, { "id": "2", "label": "cabinet", "attributes": ["white", "large"] }, { "id": "3", "label": "curtain", "attributes": ["black"] }, { "id": "4", "label": "couch", "attributes": ["large", "gray"] }, { "id": "5", "label": "cushion", "attributes": [] }, { "id": "6", "label": "blanket", "attributes": [] }, { "id": "7", "label": "table", "attributes": ["white", "large", "coffee"] }, { "id": "8", "label": "shelf", "attributes": ["several"] }, { "id": "9", "label": "decoration", "attributes": ["fancy", "lot of"] }, { "id": "10", "label": "room", "attributes": ["large", "gorgeous", "living", "little", "messy"] }], "edges": [{ "source": "1", "target": "2", "relationship": "on" }, { "source": "4", "target": "5", "relationship": "with" }, { "source": "4", "target": "6", "relationship": "with" }, { "source": "4", "target": "7", "relationship": "in front of" }, { "source": "10", "target": "1", "relationship": "contains" }, { "source": "10", "target": "3", "relationship": "contains" }, { "source": "10", "target": "4", "relationship": "contains" }, { "source": "10", "target": "7", "relationship": "contains" }, { "source": "10", "target": "8", "relationship": "contains" }, { "source": "10", "target": "9", "relationship": "contains" }] }</pre>

Fig. 7: Input/Output Pairs from GPT for Human. The left hand side shows the input text-query that we give to GPT, along with the prompt in Figure 5. These are examples from the Human dataset. The right hand side shows the returned output from GPT: a JSON text representing a “text-graph.”

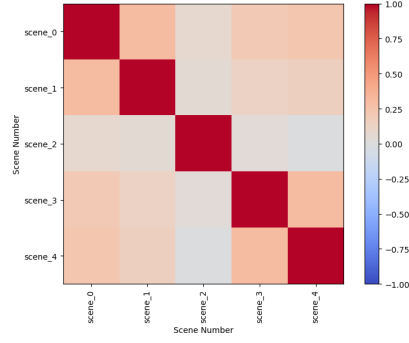


Fig. 8: Cosine Similarity Matrix for Different Scene Graph Embeddings Output with the same Text-Graph Input.

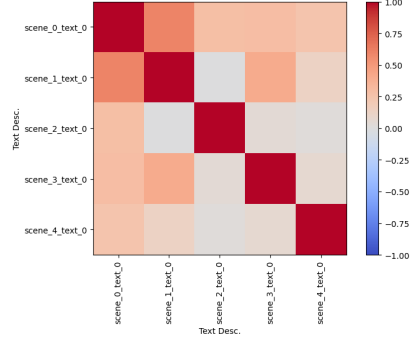


Fig. 9: Cosine Similarity Matrix for Different Text-Graph Embeddings Output with the same Scene Graph Input.



Fig. 10: Qualitative Results ScanScribe: Successful Match. The leftmost column shows the text queries used for this matching. The second, third, and fourth columns represent the top first, second, and third matching scenes found for the text-query. The green outline highlights the correct and highest matching scene for the text-query. The yellow highlights are the “matching score” for that scene, with a lower score meaning a better match. The images on the right only represent the scenes and offer a qualitative result for which scene was matched with which text-query. These text queries are from the ScanScribe dataset. For additional details about the “scene graphs” and “text-graphs” relating to these matches, please refer to the other images in this Supplementary.



Fig. 11: Qualitative Results ScanScribe: Failure Cases. Here we show some failure cases of our method on the ScanScribe dataset. The leftmost column shows the text queries used. The second, third, and fourth columns represent the top first, second, and third matching scenes found for the text-query. The green outline, if it exists, highlights the correct scene for the text-query. The yellow highlights are the “matching score” for that scene, with a lower score meaning a better match. If there are no images with a green outline in the row, then this means the correct scene was not within the top 3 matches. The images on the right only represent the scenes and offer a qualitative result for which scene was matched with which text-query. For additional details about the “scene graphs” and “text-graphs” relating to these matches, please refer to the other images in this Supplementary.

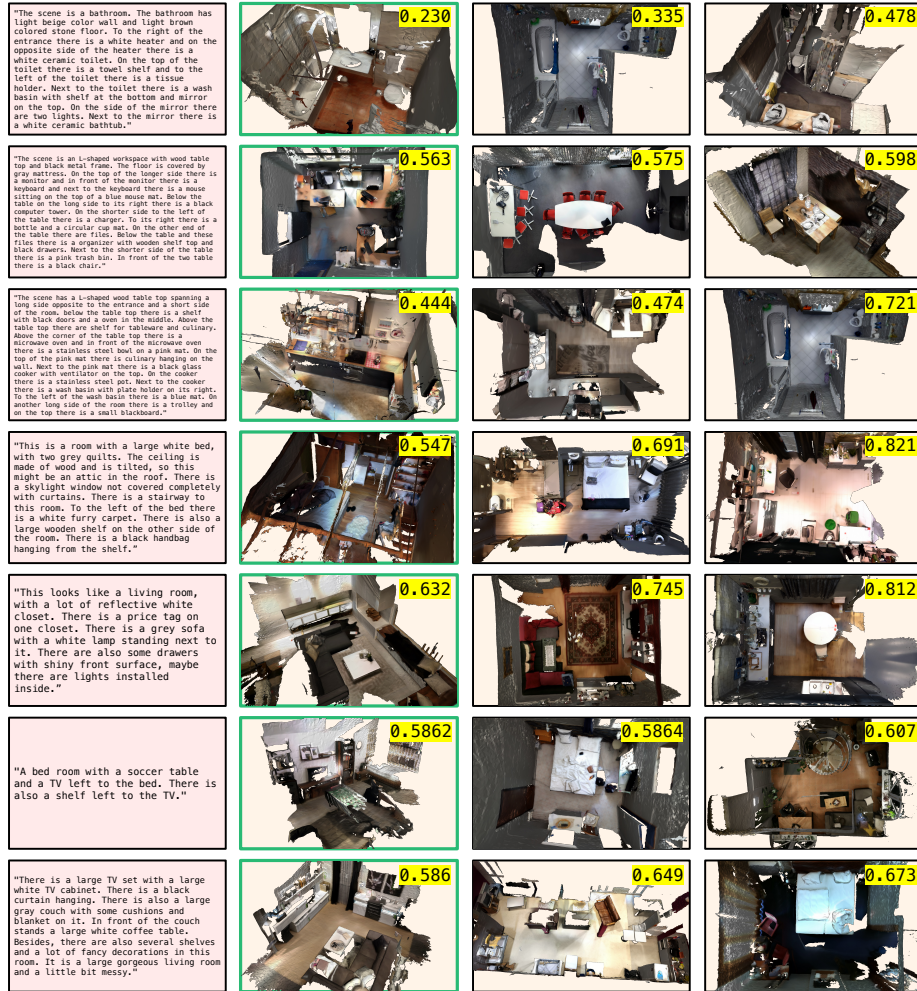


Fig. 12: Qualitative Results Human: Successful Match. The leftmost column shows the text queries used for this matching. The second, third, and fourth columns represent the top first, second, and third matching scenes found for the text-query. The green outline highlights the correct and highest matching scene for the text-query. The yellow highlights are the “matching score” for that scene, with a lower score meaning a better match. The images on the right only represent the scenes and offer a qualitative result for which scene was matched with which text-query. These text queries are from the Human dataset. For additional details about the “scene graphs” and “text-graphs” relating to these matches, please refer to the other images in this Supplementary.



Fig. 13: Qualitative Results Human: Failure Cases. Here we show some failure cases of our method on the Human dataset. The leftmost column shows the text queries used. The second, third, and fourth columns represent the top first, second, and third matching scenes found for the text-query. The green outline, if it exists, highlights the correct scene for the text-query. The yellow highlights are the “matching score” for that scene, with a lower score meaning a better match. If there are no images with a green outline in the row, then this means the correct scene was not within the top 3 matches. The images on the right only represent the scenes and offer a qualitative result for which scene was matched with which text-query. For additional details about the “scene graphs” and “text-graphs” relating to these matches, please refer to the other images in this Supplementary.