Topo-Field: Topometric mapping with Brain-inspired Hierarchical Layout-Object-Position Fields

A Appendix

A.1 Scene Partation Example

The scene can be partitioned into different regions using walls as dividers and lines can be aligned to these walls. This is similar in most scenarios, making the annotation of scene regions a straightforward task as shown in Fig. A1.

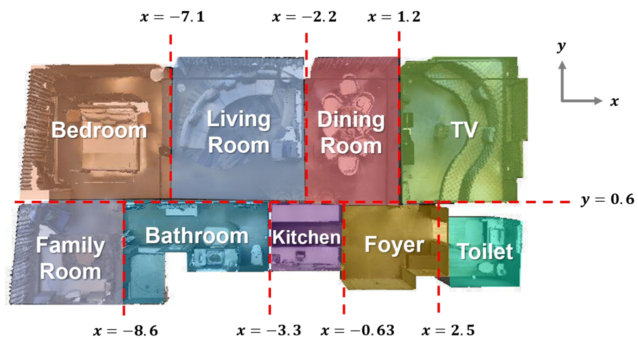


Figure A1: Using walls as dividers to associate lines with them, the scene can be divided into various regions and 3D points can be labeled with related regions easily.

A.2 Vision-language Embeddings Similarity of Region and Objects

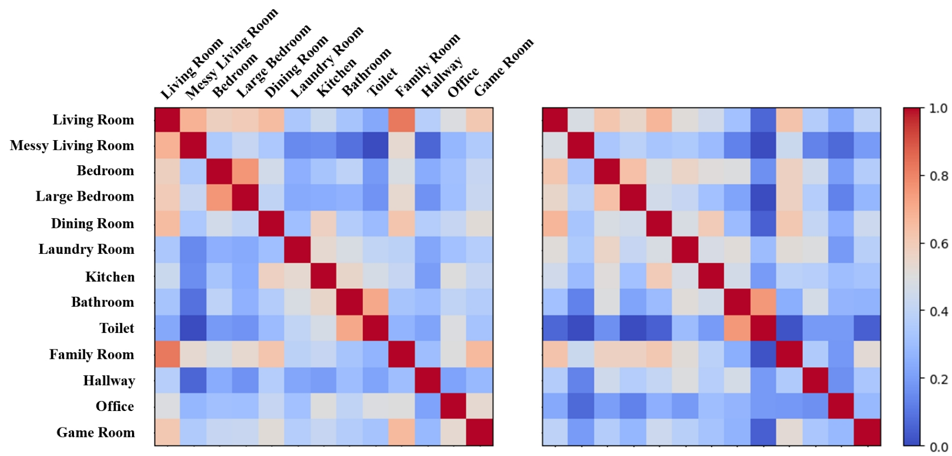
To demonstrate that the relationship of the vision-language and semantic embeddings for different regions is related to our intuition, we compare the similarity in region-region and object-region form and show the results in Fig. A2. It can be seen that based on general knowledge, cognitively related regions (e.g., the dining room and kitchen) and object-region pairs (e.g., sink and kitchen) are also more correlated in the vision-language and semantic feature spaces.

A.3 Ablation Study

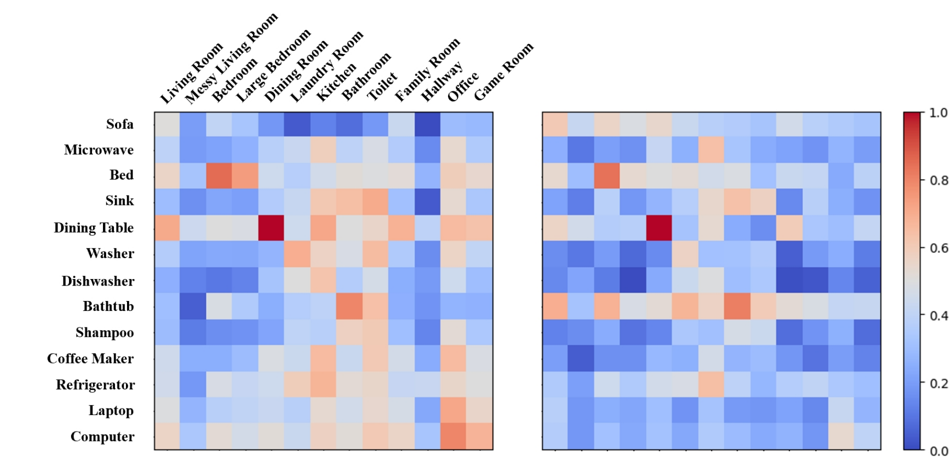
To explicitly encode the region information, we apply the LVM to process the background pixels out of the object bounding box and LLM to encode the region label text. What's more, for object pixels, object label text is combined with the region text in the form of 'object in the region' before being encoded by LLM. To ablate the contribution of vision-language embeddings from CLIP and semantic embeddings from Sentence-BERT in encoding region features, we compare different weight settings between the v-s embeddings when inferring the regions with 3D position inputs. Results are shown in Fig. A3. It can be seen that both vision-language embeddings and semantic embeddings are indispensable, and weight settings with the greatest results are used for Topo-Field.

A.4 Hierarchical Approach Comparison

Hierarchical scene representation is widely studied with numerous tasks, mainly employing scalable receptive fields and representations to fine-tune results of scalable objects and local relations. As Fig.A.8 shows, VoxFusion introduced octree map with various voxel sizes, LERF employed feature pyramids. As far as we know, few of them explicitly consider the layout level information and the association with objects and positions. This idea comes from recent neuroscience findings, and similar theory has not yet been introduced in scene representations.



(a)



(b)

Figure A2: The similarity of a set of region embeddings (as shown in a) and object-region embeddings (as shown in b). The left graph shows the vision-language embedding similarity and the right one shows the semantic embedding similarity.

A.5 TOPOMETRIC SEARCH FOR PLANNING

We employ a simple A* approach for planning. Given a topometric graph G, the start point p, and the target destination object text t. First, the belonged region r of p is inferred according to the main paper. The existing objects nodes embeddings are compared with the encoded visual-language and semantic embeddings of t to find the target object node o. At the same time, if the region of destination object r_d is declared, the search process would be more simple by directly search among region nodes. Here lists the pseudocode of the employed A*.

A.6 TOPOMETRIC MAP NODES EXAMPLES

We list the attributes of nodes and edges in the topometric map as example here in Listing $1-4$, including the object nodes, region nodes, and edges.

```
{
    "id": 0,
    "node_type": region,
    "bbox_extent": [
        4.163309999999999,
        4.207343,
```

Algorithm 1 AStar(G, r, o)

```
openSet ← {r}                                         ▷ Set of nodes to be evaluated
cameFrom ← {}                                         ▷ Mapping of nodes to their parent nodes
gScore[r] ← 0                                         ▷ Cost from start along best known path
fScore[r] ← h(r, o)                                   ▷ Estimated total cost from start to goal
while openSet is not empty do
    current ← node in openSet with lowest fScore value
    if current = o then
        return ReconstructPath(cameFrom, o)
    end if
    remove current from openSet
    for each neighbor n of current do
        tentativeGScore ← gScore[current] + d(current, n)
        if tentativeGScore < gScore[n] then
            cameFrom[n] ← current
            gScore[n] ← tentativeGScore
            fScore[n] ← gScore[n] + h(n, o)
            if n not in openSet then
                add n to openSet
            end if
        end if
    end for
end while
return "No path found"
function ReconstructPath(cameFrom, current)
    path ← [current]
    while current is in cameFrom do
        current ← cameFrom[current]
        insert current at the beginning of path
    end while
    return path
end function
```

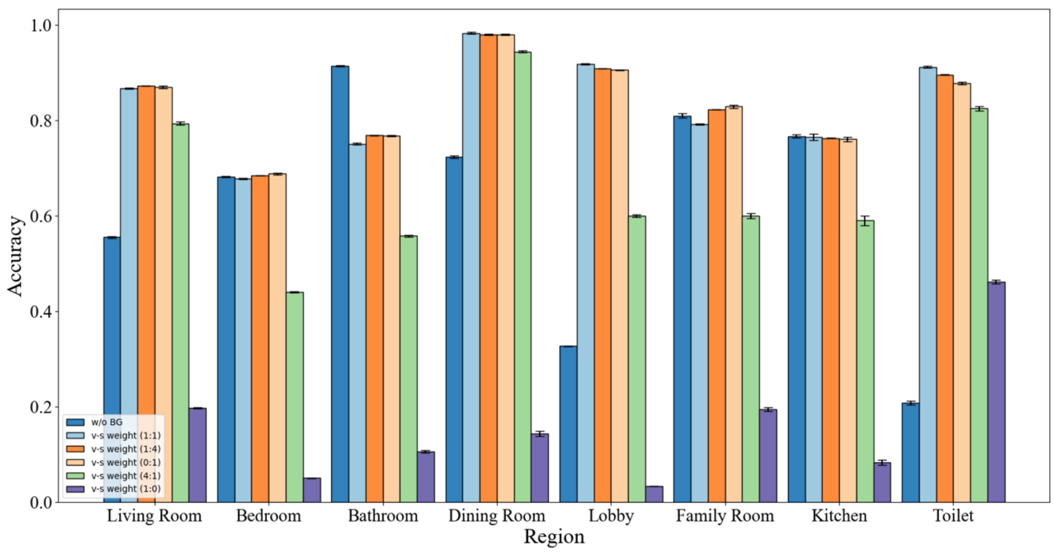


Figure A3: Ablation results on the accuracy of region prediction on Matterport3D**?** with 3D positions input. The w/o BG stands for not encoding background pixels to get region embeddings, and v-s weight ablates the weight of vision-language and semantic embeddings in the embeddings similarity contribution. Error bars show the results among samples from different scenes in Matterport3D**?**.

```
        2.53566175
    ],
    "bbox_center": [
        -8.821845,
        2.6915385,
        1.259409125
    ],
    "class": "bedroom",
    "caption": "A bedroom at the northwest of the house with warm
    lighting. Main objects include a bed in the center, a large closet,
    and a dresser at the corner."
},
```

Listing 1: Region node

```
{
    "id": 1,
    "node_type": object,
    "bbox_extent": [
        0.3569,
        0.2297,
        0.101.8
    ],
    "bbox_center": [
        0.3222,
        -1.1108,
        -0.5062
    ],
    "class": "picture",
    "caption": "A white framed picture hanging on the wall."
},
```

Listing 2: Object node

```
{
```

```
    "id": 0,
    "node_type": Entrance,
    "bbox_extent": [
        0.5,
        1.6,
        2.8,
    ],
    "bbox_center": [
        -3.244,
        -0.276,
        0.487
    ],
    "class": "Entrance",
    "caption": "Entrance connecting bedroom and living room."
},
```


Listing 3: Entrance node


```
{
    "id": 2,
    "edge_type": region_entrance,
    "start_node": {
        "id": 0,
        "node_type": region,
        "bbox_extent": [
            4.163309999999999,
            4.207343,
            2.53566175
        ],
        "bbox_center": [
            -8.821845,
            2.6915385,
            1.259409125
        ],
        "region_tag": "bedroom"
    },
    "end_node": {
        "id": 0,
        "node_type": Entrance,
        "bbox_extent": [
            0.5,
            1.6,
            2.8,
        ],
        "bbox_center": [
            -3.244,
            -0.276,
            0.487
        ],
        "class": "Entrance",
        "caption": "Entrance connecting bedroom and living room."
    },
    "relationship": connected,
    "position_relation": "b to the southeast of a",
    "position_reason": "The x-coordinate of the center of bbox of
    end_node (-3.244) is larger than that of start_node (-8.821845), and
    the y-coordinates of the center of bbox of end_node (-0.276) is less
    than that of start_node (4.207343). Therefore, b is to the southeast
    of a."
    "caption": "The pathway from bedroom to living room."
},
```


Listing 4: Region entrance edge


```
{
```

```
    "id": 2,
    "node_type": object_region,
    "start_node": {
        "id": 7,
        "node_type": object,
        "bbox_extent": [
            2.155,
            2.052,
            0.883
        ],
        "bbox_center": [
            5.598,
            2.566,
            0.136
        ],
        "class": "bed",
        "caption": "a bed with a white comforter and a pillow"
    },
    "end_node": {
        "id": 0,
        "node_type": region,
        "bbox_extent": [
            4.163309999999999,
            4.207343,
            2.53566175
        ],
        "bbox_center": [
            -8.821845,
            2.6915385,
            1.259409125
        ],
        "class": "bedroom"
        "caption": "A bedroom at the northwest of the house with warm
    lighting. Main objects include a bed in the center, a large closet,
    and a dresser at the corner."
    },
    "relationship": belong,
    "position_relation": "a in the center of b",
    "caption": "According to the bbox center position and extent, the bed
     is in the center of bedroom."
},
```

Listing 5: Object region edge

A.7 Prompt Example for Region Node Connectivity Description

With topometric mapped nodes, we leverage LLM to describe the connectivity of nodes according to the general knowledge and bounding box 3D position. In listing 5, here we provide a prompt example to describe the connectivity relationship between content objects and regions and set up the edge.

```
{
DEFAULT_PROMPT_POST = """
You are an excellent graph managing agent. Given a graph nodes set of an
    environment,
you can explore the relationships of nodes with their attributes and
    build edges among
them.

The input is a list of JSONS describing two types of nodes, including the
     object and
region. You need to produce a JSON string (and nothing else) and set up
    edges between them with keys: "relationship", "position_relation" and
     "caption".
```

```

Each of the JSON fields will have the following fields:
1. id: a unique number
2. node_type: type of this node
3. bbox_extent: the 3D bounding box extents
4. bbox_center: the 3D bounding box center
5. class: an extremely brief description
6. caption: a sentence describing node attributes in detail

Produce a "relationship" field that best describes the relationship of the object node and region node. Set "false" if the object is not related to the area or is not reasonable, the relationship is refused. Produce a
"position_relation" field describing the position relationship between object and region according to their
bounding box information in the 3D space. Before producing the "position_relation" field, produce a "caption" field that explains why the "position_relation" field is reasonable.

The built edges should include following fields:
1. id: a unique number of each edge in order
2. node_type: according to the connected node type in the form "start_node\_end_node"
3. start_node: keep JSON values of the object node unchanged
4. end_node: keep JSON values of the region node unchanged
5. relationship
6. position_relation
7. caption
"""
```

Listing 6: Prompt example to set up edge with nodes.

A.8 ADDITIONAL EXPERIMENT RESULTS

Additional experiments results of object localization using text query inputs and view localization using image query inputs. Also, a table is provided showing the metric on exactly each region class from 4 scenes in Matterport3D dataset.

Regions	**Scene1**			**Scene2**			**Scene3**			**Scene4**		
	Acc.	Pre.	F1	Acc.	Pre.	F1	Acc.	Pre.	F1	Acc.	Pre.	F1
Living Room	0.948	0.970	0.959	0.870	0.881	0.875	0.778	0.810	0.793	0.902	0.949	0.925
Bedroom	0.943	0.825	0.880	0.925	0.923	0.924	0.687	0.767	0.725	0.920	0.870	0.894
Bathroom	0.466	0.680	0.554	0.903	0.898	0.901	0.875	0.463	0.605	0.797	0.831	0.814
Dining Room	-	-	-	0.961	0.794	0.870	0.774	0.732	0.752	0.933	0.887	0.910
Lobby	0.681	0.941	0.790	0.853	0.951	0.899	0.978	0.510	0.671	0.855	0.698	0.769
Family Room	-	-	-	-	-	-	0.903	0.571	0.700	0.926	0.936	0.931
Kitchen	0.994	0.654	0.789	0.788	0.836	0.811	0.833	0.833	0.833	0.758	0.854	0.803
Office	-	-	-	0.969	0.848	0.905	-	-	-	0.953	0.883	0.917
Toilet	-	-	-	-	-	-	0.900	0.711	0.795	-	-	-
Avg. Acc./Samples	0.886 / 169k			0.900 / 185k			0.884 / 111k			0.894 / 112k		

Table 1: Region prediction results on the test set of different scenes from the Matterport3D**?** dataset. Accuracy, precision, and F1 score are used as metrics.

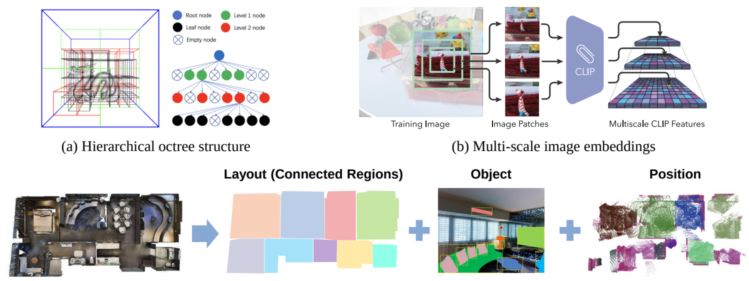


Figure A4: The comparison of the hierarchical scene representation strategy against previous works.

Figure A5: Text query localization on scene 2t7WUuJeko7**?**.

Figure A6: Text query localization on scene 17DRP5sb8fy**?**.

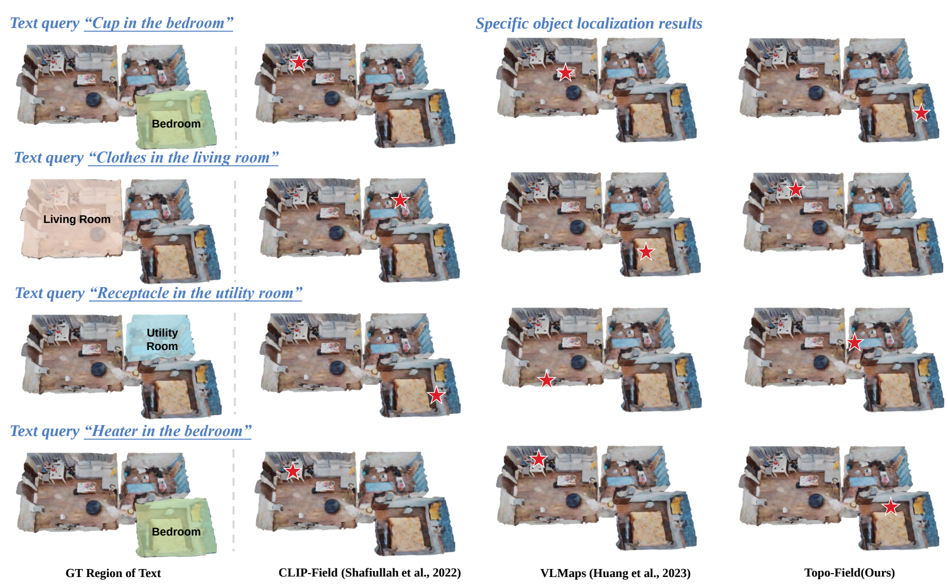


Figure A7: Text query localization on scene Apartment**?**.

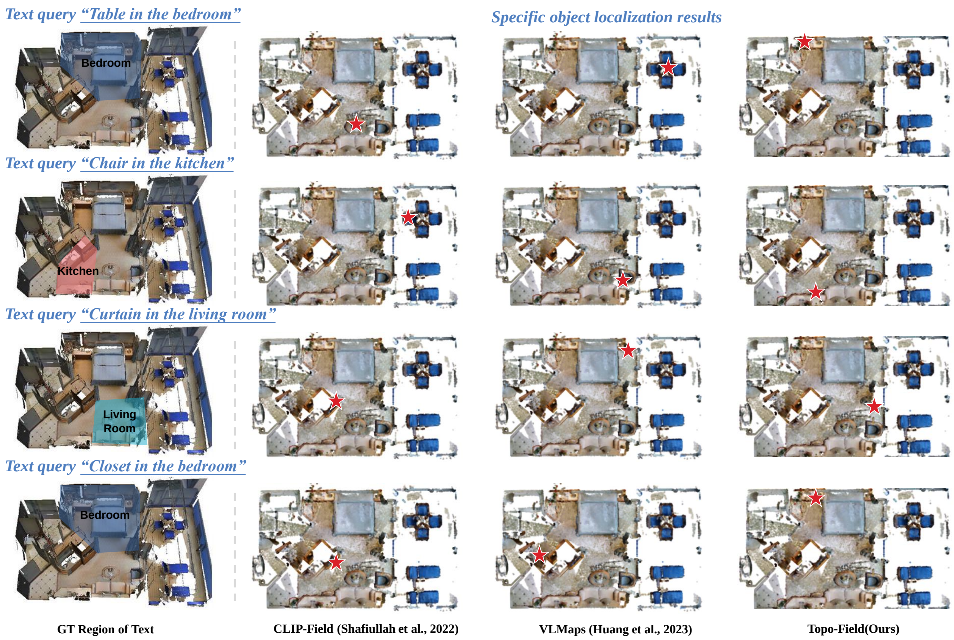


Figure A8: Text query localization on scene HxpKQynjfin**?**.

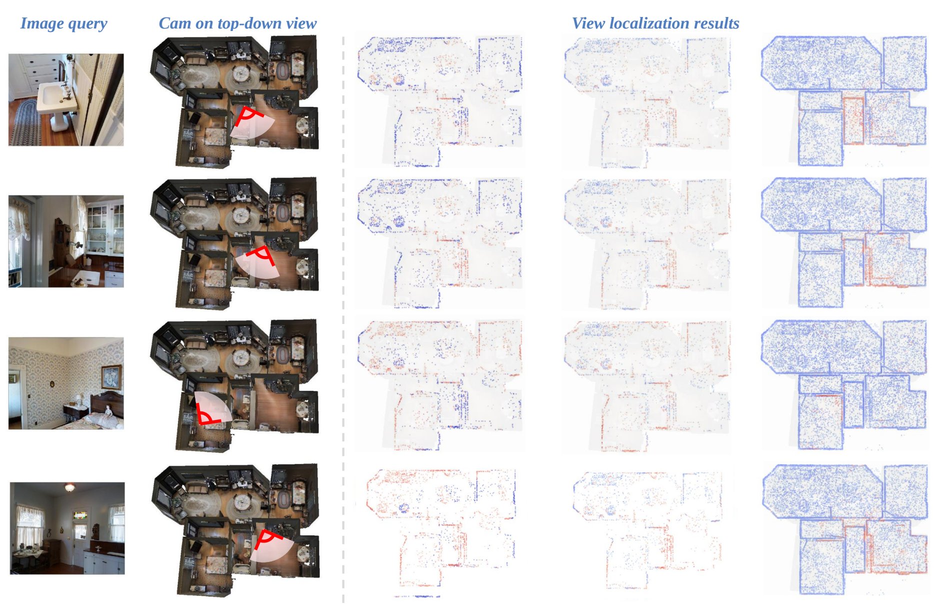


Figure A9: Image query localization on scene 2t7WUuJeko7**?**.

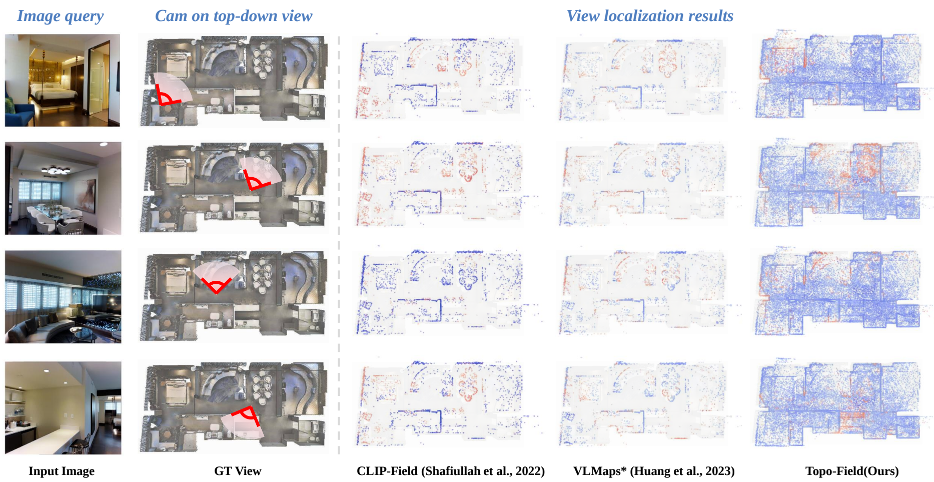


Figure A10: Image query localization on scene 17DRP5sb8fy**?**.

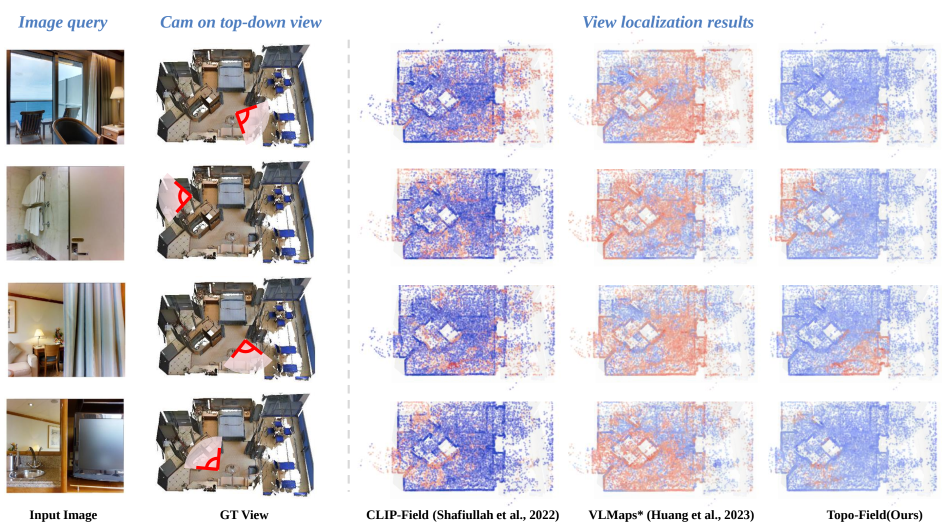


Figure A11: Image query localization on scene HxpKQynjfin**?**.