6 Appendix section

6.1 Benchmark task descriptions

The tasks are categorized into single-view and multi-view settings, covering depth estimation, distance prediction, spatial relations, and spatial imagination. The descriptions of each task are as follows:

Single-view tasks Single-view tasks test a model's ability to infer spatial properties from a single image. These tasks include:

- **Depth estimation (OC, OO, NA)**: Predicting absolute or relative depth values for objects
- **Distance prediction (OC, OO, NA)**: Estimating the Euclidean distance between objects or from an object to the camera.
- **Object center distance inference (OO, MCA)**: Given objects A, B and C, determine which of B and C is farther or closer to A.
- **Object spatial relation (OO, MCA)**: Determining relative positioning (e.g., left, right, in front of).
- **Spatial imagination (OC, OO, MCA)**: Predicting unseen spatial relationships based on limited visual information.

Multi-view tasks Multi-view tasks require reasoning across multiple images to infer spatial relationships. These tasks include:

- **Viewpoint change inference (NA)**: Given two perspectives, output how the camera should be moved to see the second perspective.
- **Multi-view depth estimation (OC, OO, NA)**: Predicting depth across multiple perspectives.
- **Multi-view distance prediction (OC, OO, NA)**: Estimating object distances across different views.
- **Multi-view object matching (MCA)**: Identifying the same object across multiple views.
- **Camera pose inference (MCA)**: Predict the position of the camera corresponding to the second perspective in the first image.
- **Multi-view object spatial relation (OC, OO, MCA)**: Determining object relationships across multiple images.
- **Spatial imagination (OC, OO, MCA)**: Reasoning about spatial structure beyond visible views.

A tiny version of our *SPAR-Bench* evaluation results are shown in table 8.

6.2 Image subsampling

We propose an efficient image filtering method based on camera poses to reduce redundant images with high similarity, so that can improve data processing efficiency. Given a scene S with a set of image sequence $\mathcal{I}$, our goal is to filter out similar images based on a translation threshold d_{trans} and a rotation angle threshold d_{rot}, obtaining a compact image sequence $\mathcal{I}' \subseteq \mathcal{I}$.

Specifically, for a given image sequence $\mathcal{I}$, we first load the corresponding camera intrinsic and extrinsic parameters. Each camera pose is represented by a 4×4 transformation matrix $\mathbf{T}_i$, consisting of a rotation matrix $\mathbf{R}_i$ and a translation vector $\mathbf{t}_i$:

$$\mathbf{T}_i = \begin{bmatrix} \mathbf{R}_i & \mathbf{t}_i \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{1}$$

where $\mathbf{t}_i \in \mathbb{R}^3$ and $\mathbf{R}_i \in \text{SO}(3)$. The world-to-camera transformation by inverting the given pose.

Translation filtering For each image $i (i = 1, .., n)$, we compute the Euclidean distance between its translation vector $\mathbf{t}_i$ and the translation vector $\mathbf{t}_j$ of a candidate image $j (j = i + 1, .., n)$:

$$d_{ij}^{trans} = \|\mathbf{t}_i - \mathbf{t}_j\| \tag{2}$$

If $d_{ij}^{trans} > d_{th}$, we believe that the difference between these two frames is significant enough and we will preserve the current frame j. If $d_{ij}^{trans} < d_{th}$, we will further perform rotation filtering.

Rotation filtering For images with smaller d_{ij}^{trans}, we compute the relative rotation matrix: $\mathbf{R}_{ij} = \mathbf{R}_i^{-1}\mathbf{R}_j$ The rotational difference is determined by the angle θ_{ij}, computed as:

$$\theta_{ij} = \cos^{-1}\left(\frac{\text{Trace}(\mathbf{R}_{ij}) - 1}{2}\right) * \frac{180}{\pi} \tag{3}$$

If $\theta_{ij} < \theta_{th}$, image j is considered redundant and removed. After iterating through all images, the final filtered image set is as follows:

$$\mathcal{I}' = \{i \in \mathcal{I} \mid \text{satisfies filtering criteria}\} \tag{4}$$

This method can filter out approximately 90% of redundant images, which ensures that only images with sufficiently distinct poses are retained, reducing redundancy while preserving viewpoint diversity.

In the experimental setup, we set the threshold parameter of the ScanNetPP dataset [27] with $d_{th} = 0.5$ and $\theta_{th} = 45$ and ScanNet [26] dataset with $d_{th} = 0.5$ and $\theta_{th} = 15$. For the Structured3D Dataset, we did not perform filtering and subsampling operations since the images in the dataset were sparse enough.

Methods	**Avg.**	**Low**	Depth-OC	Depth-OC-MV	Depth-OO	Depth-OO-MV	Dist-OC	Dist-OC-MV	Dist-OO	Dist-OO-MV	**Medium**	PoseMatch	CamMotion	ViewChgI	**High**	Dist-I-OO	Dist-I-OO-MV	ObjRel-OC-MV	ObjRel-OO	ObjRel-OO-MV	Spatmg-OC	Spatmg-OC-MV	Spatmg-OO	Spatmg-OO-MV
Baseline																								
Chance Level (Random)	-	-	-	-	-	-	-	-	-			22	18	-		80	32	26	28	22	32	12	30	28
Chance Level (Frequency)	37.80	36.33	42.89	51.78	25.78	27.11	35.33	46.89	35.33	25.56	41.14	30	40	53.42	38.00	60	58	32	30	30	32	34	28	38
Human Level	67.27	55.31	72.75	74.25	28.75	36.25	78.25	52.25	66.5	33.50	72.32	92	64	60.97	76.22	80	94	70	92	80	78	82	50	60
GPT-4o	36.39	29.25	53.80	45.00	15.00	13.60	37.40	34.40	23.40	24.40	24.93	30	16	28.80	45.11	64	64	58	46	46	32	44	30	22
Claude-3.7-Sonnet	21.77	25.43	41.00	45.40	11.20	12.20	42.60	19.60	26.00	5.40	7.33	16	6	0.00	23.33	40	48	22	36	14	12	20	6	12
Qwen2-VL-72B	35.62	35.28	45.40	49.80	13.80	10.00	54.60	49.40	36.80	22.40	23.39	42	18	10.16	40.00	60	68	50	38	44	18	28	18	36
Qwen2.5-VL-72B	39.40	35.35	53.20	46.80	17.80	29.00	49.60	57.40	14.40	14.60	23.05	40	16	13.16	48.44	74	74	60	56	50	20	34	24	44
InternVL2-2B	29.51	21.85	15.00	31.40	17.80	18.80	13.40	27.40	26.40	24.60	25.81	44	26	7.44	37.56	46	56	54	42	18	50	42	14	16
InternVL2-4B	32.10	29.55	22.02	28.40	18.80	14.20	47.60	52.60	26.00	26.60	33.88	52	30	19.64	33.78	46	54	44	30	30	26	26	26	22
InternVL2-8B	32.95	24.10	24.60	39.00	16.00	16.80	35.40	33.40	13.40	14.20	35.43	58	28	20.28	40.00	68	42	40	46	34	34	46	16	34
InternVL2.5-2B	31.81	27.85	44.80	42.20	11.20	7.00	40.20	35.40	24.20	17.80	22.48	40	22	5.44	38.44	68	48	50	48	26	18	38	20	30
InternVL2.5-4B	33.99	30.38	31.20	36.20	26.20	30.00	24.20	36.40	31.40	27.40	34.27	58	38	6.80	37.11	48	58	54	40	30	24	42	18	20
InternVL2.5-8B	37.27	28.38	27.40	31.80	19.60	19.00	40.40	48.80	15.00	25.00	31.47	66	22	6.40	47.11	58	54	50	52	52	44	58	22	34
Qwen2-VL-2b	26.88	23.45	44.20	50.00	25.20	17.40	7.40	12.60	20.60	10.20	28.01	22	24	38.04	29.56	52	50	20	24	10	40	30	24	16
Qwen2-VL-7b	32.84	27.98	37.80	36.20	23.60	7.00	28.00	31.80	31.60	27.80	16.36	26	18	5.08	42.67	58	54	26	40	54	34	36	40	42
Qwen2.5-VL-7b	33.48	31.25	27.80	37.20	27.40	19.80	50.00	47.60	17.60	22.60	19.84	26	24	9.52	40.00	52	50	44	56	28	28	36	32	34
LLaVA-OV-0.5B	30.84	33.20	55.40	51.60	22.80	10.00	35.20	28.20	36.60	25.80	15.08	24	20	1.24	34.00	52	56	40	36	16	30	40	22	14
LLaVA-OV-7B	34.73	27.95	42.80	44.60	25.20	24.00	12.80	12.60	38.40	23.20	27.69	48	22	13.08	43.11	64	62	26	58	42	24	40	32	40
llava-v1.5-7b	25.76	13.02	4.80	15.40	17.60	17.60	8.80	7.80	17.60	14.60	33.69	28	40	33.08	34.44	52	54	18	22	26	42	38	18	40
llava-v1.6-7b	13.50	9.00	10.60	0.00	20.40	0.00	16.20	0.00	24.80	0.00	6.00	8	10	0.00	20.00	46	14	12	30	6	42	6	20	4
ours	66.65	70.33	87.00	83.20	45.80	43.20	81.00	84.00	78.80	59.60	60.13	78	66	36.40	65.56	86	90	72	78	58	48	48	42	68

Table 8: **Performance of different models on *SPAR-Bench*.** All results are obtained on tiny *SPAR-Bench*. Shaded cells indicate best scores in each category.

6.3 Image item construction

Given a scene S, we construct image items by extracting 3D object data and projecting it onto 2D images.

Data loading and initialization For each scene, we load the corresponding 3D mesh, camera intrinsic and extrinsic parameters, and instance annotations. The scene mesh is represented as: $M = (\mathcal{V}, \mathcal{F})$ where $\mathcal{V}$ is the set of vertices and $\mathcal{F}$ is the set of triangular faces.

To determine the visibility of 3D faces in the image, we perform rasterization to obtain a mapping from image pixels to face indices: $pix\_to\_face_{(x,y)} = f_k$. where $f_k \in \mathcal{F}$ and pix_to_face stores the corresponding face index for each pixel (x, y). If a pixel does not correspond to any face, it will be marked as -1.

Object projection and bounding box computation For each 3D object, we compute the set of visible vertices and project them into the 2D image plane using:

$$p_{2D} = K(R \cdot p_{3D} + t) \tag{5}$$

where p_{3D} is a vertex in the 3D space, and p_{2D} is its projected 2D coordinate. The bounding box of the projected object is computed as:$B_{obj} = [x_{min}, y_{min}, x_{max}, y_{max}]$.To ensure a valid projection, we also enforce some constraints as follows:

- The fraction of visible object vertices f_v in the image must exceed a threshold τ_v, where $f_v = \frac{|V_{visible}|}{|V_{total}|}$.
- The projected object area must be above a minimum threshold A_{min}.
- The depth values in the z-buffer must be within a reasonable range. where $z_{min} = \min(Z_{obj}), \quad z_{max} = \max(Z_{obj})$

Each 3D object is associated with an oriented bounding box, defined by its centroid c, axis-aligned lengths l_x, l_y, l_z. Finally, the extracted image item dictionary, including object data, is used for downstream task generation.

6.4 Task data generation

In this section, we describe the detailed information on multi-task generation. We generate questions based on the template. These questions can be of three types: select, fill, or sentence. In each case, the goal is to generate a question that involves the spatial relationship between two objects. We will provide a Q&A format in the form of a template and fill in key information and answers in it.

Obj spatial relation This task is to describe the spatial relationships between objects in the 3D scene based on their spatial positions. The process involves several key steps: (1) Transforming 3D object coordinates from the original camera view into a common view. This transformation ensures that all spatial calculations are relative to the main camera view. Let c denotes the 3D center of an object in the world coordinate system, and $\mathbf{T}$ denotes the camera pose. The transformation is carried out as: $c'_{homo} = \mathbf{T}^{-1} \cdot c_{homo}$. where c_{homo} means homogeneous coordinate of object 3D center point. (2) Spatial Relationship Description. We describe their spatial relationships in terms of several key factors: above-below, left-right, near-far, and front-behind (relative to two objects). These relationships are determined based on their spatial coordinates and distance from the camera center. The distance is calculated by $d = \|c' - T_{trans}\|_2$. We set the relationship threshold at 0.1m. If the difference in coordinates or distances is less than 0.1m, we consider the corresponding spatial relationship to be indistinguishable (empty).

Depth prediction Given an image I containing a set of detected objects $\mathcal{O} = \{o_1, o_2, ..., o_n\}$, we transform each object's 3D center point c into the camera coordinate system as $c' \in \mathbb{R}^3$. Then the transformed depth values $d$are extracted from the z-component of their transformed coordinatesc', which means $d = c'(z)$.

For the absolute depth prediction task, we use this value as the standard answer. For the relative depth estimation task, we calculate the depth difference between objects by: $\Delta d = |d_i - d_j|$. We will skip that case if two objects have overlapping bounding boxes or similar values.

Distance infer Given an image I containing a set of objects $\mathcal{O} = \{o_1, o_2, ..., o_n\}$, we define the 3D center of each object o_i in the world coordinate system as $\mathbf{c}_i \in \mathbb{R}^3$ and transformed them into the camera coordinate system as $\mathbf{c}'_i$. For the object-object type task, we random sample two objects o_A with $\mathbf{c}'_A$ and o_B with $\mathbf{c}'_B$ in the same scene. The Euclidean distance between them is given by: $d_{AB} = \|\mathbf{c}'_A - \mathbf{c}'_B\|_2$, where $\|\cdot\|_2$ represents the L_2-norm. For the object-camera type task, we calculate the distance with $|\mathbf{c}'_i|_2$. To ensure numerical stability and consistency in question-answer generation, the computed distance is rounded to the nearest 0.1 meter. If the two objects have overlapping 3D bounding boxes or the distance is smaller than the threshold, we will skip that case.

Spatial volume infer For an object o_i in the image, we first obtain its 3D bounding box in the world coordinate system and then transform it into the camera coordinate system by the extrinsic transformation matrix. The center coordinate is denoted as $\mathbf{c}'_i$ and each corner point of 3D bounding box is denoted as $\mathbf{b}'_{ij}$, $j = (1, 2, ..., 8)$. The object's dimensions(length, width, and height) are

derived as follows:

$$
\begin{aligned}
h &= \max_j b_{i,j}^{(z)} - \min_j b_{i,j}^{(z)} \\
l &= \max_{j,k} ||b_{i,j}^{(xy)} - b_{i,k}^{(xy)}|| \\
w &= \min_{j,k} ||b_{i,j}^{(xy)} - b_{i,k}^{(xy)}||
\end{aligned}
\tag{6}
$$

where $b_{i,j}^{(xy)} = (b_{i,j}^{(x)}, b_{i,j}^{(y)})$ represents the 2D projection of the bounding box in the XY plane. To ensure consistency, all dimensions are converted to centimeters. The final estimated volume is given by: $V = h \cdot l \cdot w$.

Spatial imagination Our spatial imagination task aims to evaluate the spatial reasoning capabilities of LLM models by analyzing object relationships before and after camera transformations in a 3D environment. Given an image and corresponding scene metadata, we randomly sample objects and generate structured question-answer (QA) pairs that describe spatial relationships.

For each image I, it is associated with a set of objects $\mathcal{O} = \{o_1, o_2, ..., o_N\}$. We randomly sample objects $o_A, o_B, o_C, o_D \subset \mathcal{O}$ for relational comparisons. To analyze object relationships from different viewpoints, we transform the camera pose $\mathbf{P} \in SE(3)$ so that it moves towards object A and faces object B. The new camera pose is constructed as follows:

$$
\begin{aligned}
\mathbf{t}_{A\to B} = \mathbf{c}_A \quad \mathbf{f} = \frac{\mathbf{c}_B - \mathbf{c}_A}{\|\mathbf{c}_B - \mathbf{c}_A\|} \\
\mathbf{v} = \frac{\mathbf{u}_0 \times \mathbf{f}}{\|\mathbf{u}_0 \times \mathbf{f}\|} \quad \mathbf{u} = \mathbf{f} \times \mathbf{v} \quad \mathbf{r} = -\mathbf{v}
\end{aligned}
\tag{7}
$$

where $\mathbf{c}_A, \mathbf{c}_B$ are 3D center coordinates of object o_A and o_B respectively. $\mathbf{u}_0 = [0, 0, 1]^T$. $\mathbf{f}$ is forward direction vector. $\mathbf{v}, \mathbf{u}, \mathbf{r}$ are left-vector, up-vector and right-vector respectively. So the new camera pose is computed as:

$$
\mathbf{P}_{A\to B} = \begin{bmatrix} \mathbf{r} & \mathbf{u} & \mathbf{f} & \mathbf{t}_{A\to B} \\ 0 & 0 & 0 & 1 \end{bmatrix}
\tag{8}
$$

We extract the up-vector $\mathbf{u}$ from $\mathbf{P}_{A\to B}$ to compute the vertical rotation angle: $\theta = \cos^{-1}(\mathbf{u}_z)$. If $\theta > 60°$, we will discard this viewpoint to maintain a reasonable observation angle.

After that, we compute the spatial relationship between object C and object D before and after camera transformation. The relationships are determined based on their relative positions in the original and transformed coordinate system. We describe their spatial relationships in terms of several key factors: above-below, left-right, near-far, and front-behind (relative to two objects). Please refer to the paragraph **Obj spatial relation** for details. The final step is to generate structured question-answer pairs. The same procedure is also applied after the camera-object type task.

Position matching Position matching aims to identify and compare the positions of the same object across different views. Given an object detected in multiple images, the task is to find its corresponding 2D bounding box in another view based on its known location in one reference image.

We define the set of detected objects as: $\mathcal{O} = \{o_1, o_2, \ldots, o_N\}$. Each object o_i appears in a set of images: $\mathcal{I}_i = \{I_{i_1}, I_{i_2}, \ldots, I_{i_m}\}$. If $m < 2$, the object is discarded. For each valid object, we randomly select two distinct images $I_{i_A}, I_{i_B} \in \mathcal{I}_i$ as a reference frame and target frame. The 3D bounding box of object o_i in world coordinate is denoted as b_i^{3D}. We project the 3D bounding box into the image plane by Eq. 5. The 2D bounding boxes in different images are denoted as $b_{i_A}^{2D}$, $b_{i_B}^{2D}$.

The system formulates the position-matching question as: Qusetion = Given object o_i and 2D bounding box $b_{i_A}^{2D}$in image I_A, find its location in image I_B; Answer = $b_{i_B}^{2D}$.

View chang infer The view change inference task aims to determine the spatial displacement between different images. To ensure images exhibit a co-visibility relationship, we select different perspectives images containing the same object instance.

For two distinct images I_A, I_B containing the same object o, we first compute the center of the 2D bounding box to quantify the object's displacement in image space. If 2D bounding box of object o in image I_A is denoted as $b_A^{2D} = [x_A^{\min}, y_A^{\min}, x_A^{\max}, y_A^{\max}]$, the 2D center coordinate can be

calculated as $\mathbf{c}_A^{2D} = \big((x_A^{\min} + x_A^{\max})/2, (y_A^{\min} + y_A^{\max})/2\big)$. We determined the object's location in image I_A, I_B according to $\mathbf{c}_A^{2D}$ and $\mathbf{c}_B^{2D}$.

To further analyze the view movement in the world coordinate system, We compute the relative pose transformation: $\mathbf{T}_{AB} = \mathbf{T}_A^{-1}\mathbf{T}_B$. Finally, we determine the translation distance and rotation angle of the viewpoint based on the relative pose transformation matrix and designed rules.

Camera pose The camera pose task aims to estimate the relative camera motion and generate questions about the 2D coordinate and depth information.

We select distinct frames I_A, I_B containing the same object o to ensure there is an overlap between them. Firstly, we project the camera position of image I_B in the world coordinate into the coordinate system of image I_A. According to Eq. 5, the projected point coordinate is given by $p_{B|A} = K(R_A \cdot p_B + t_A)$, where R_A and t_A are the rotation matrix and the translation vector of I_A, K is camera intrinsic parameters. Then we calculate 2D image coordinates (u, v) and normalize to 0-1000:

$$u = \frac{p_{B|A}[0]}{p_{B|A}[2]} \cdot \frac{1000}{\text{width}} \quad v = \frac{p_{B|A}[1]}{p_{B|A}[2]} \cdot \frac{1000}{\text{height}} \tag{9}$$

If the projected point lies within the bounds of the image I_A, we compute the depth as: $d_{B|A} = (R_A \cdot p_B + t_A)[2]$

Obj frame location We introduce the object frame location task to identify the frames in which a given object appears. We select a reference frame and determine in which other frames the object is present. This process enables the automatic generation of question-answer pairs related to object appearance across frames.

For an object o, we extract the set of frames: $I_o = \{I_1, I_2, ..., I_n\}$ in which it appears. To generate questions and answers for the object frame localization task, we randomly select one frame I_s as the reference frame. We also add some irrelevant frames as wrong options, and the answer consists of the list of other frame indices $I_o \setminus \{I_s\}$ where the object appears.

Obj frame location This task infers the chronological order in which multiple objects appear within a sequence of frames. By selecting a subset of objects and analyzing their first occurrence across frames, we generate structured question-answer pairs that facilitate the temporal reasoning ability of LLMs.

Given object set $\mathcal{O} = \{o_1, o_2, ..., o_N\}$ and associated image set $\mathcal{L} = \{\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_N\}$, where $\mathcal{I}_i$ means the set of frames that object o_i appears, we extract the first appearance frame of each object as $F_i = \min(f \mid f \in \mathcal{I}_i)$. F_i represents the first appearance frame of object o_i. Then we sort the first appearance frames of all objects to determine the order of appearance:

$$\mathcal{S} = \text{sort}(\{(o_1, F_1), (o_2, F_2), (o_3, F_3)...(o_N, F_N)\}) \tag{10}$$

where $\mathcal{S}$ is the ordered sequence of objects based on their first appearance. For sentence QA type, the ordered sequence and corresponding frame indices are embedded into a sentence. For fill-in-the-blank QA type, the question is to instruct the user to input the ordered sequence as a comma-separated list.

Obj count The object counting task estimates the number of object instances for each label category in the scene and generates structured question-answer pairs to facilitate the numerical reasoning ability of LLMs.

For each label $l \in \mathcal{L}$, the total number of object instances is computed as: $N(l) = |\mathcal{L}(l)|$, where $N(l)$ represents the number of object instances associated with label l. We will exclude labels with fewer than two object instances.

Room size This task is designed for estimating the size of a room and generating corresponding question-answer pairs to facilitate the spatial reasoning ability of LLMs.

Given the room scene with 3D mesh $M = (\mathcal{V}, \mathcal{F})$, where $\mathcal{V}$ is the set of vertices and $\mathcal{F}$ is the set of faces, we first downsample the points by quantizing them into a grid with a specified voxel size $\delta = 0.1$. The quantized points are computed as: $\mathcal{Q} = \lfloor \mathcal{V}/\delta \rfloor$. We then retain only the unique voxels and obtain voxel centers: $\mathcal{P}_d = (\text{unique}(\mathcal{Q}) + 0.5) \cdot \delta$. If the downsampled set contains fewer than 100 points, we revert to the original point cloud. To estimate the room area, we construct a concave hull using the α-shape algorithm:

$$\mathcal{H} = \text{AlphaShape}(\mathcal{P}_d[:, 0:2], \alpha) \tag{11}$$

where $\alpha = 0.1$ controls the concavity of the shape. The final room area is calculated as:

$$A = \sum_{h \in \mathcal{H}} \text{area}(h) \tag{12}$$

To ensure valid QA generation, If the room area is below the threshold $A_{th} = 5$, no QA pairs will be generated.

Navigation We construct visual navigation data based on Matterport3D [51] and Room Across Room (RxR) [52] Dataset. For the navigation instructions and image sequences in RxR, we take the image sequence as input and construct question-answer pairs. We expect the LLM model to complete the absent key action information in the instructions, such as left turn, straight ahead, on the right side, and other keywords.

6.5 More visualization

We visualize the detailed QA of different tasks from our proposed *SPAR-7M* in table 9 - 14.

Table 9: Detailed QA of the Depth Prediction Object-Camera Multi-view Task

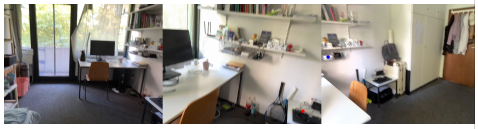
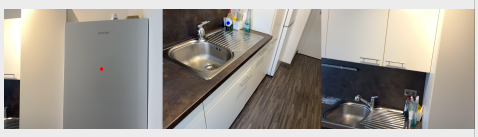
Task	Question	Answer
Depth-OC-MV (fill)	The table (**red point**) is located at a depth of 1.5 meters. Estimate the depth of the food container (**blue point**). Calculate or judge based on the 3D center points of these objects. Ensure your answer contains only one number. 	1.7
Depth-OC-MV (select)	Given the refrigerator (**red point**) is located at a depth of 0.9 meters in the Z-axis of the camera coordinate system, how far in depth is the dish soap bottle (**blue point**) at its center? Calculate or judge based on the 3D center points of these objects. Please select the correct option from the choices provided. **A. 2.2; B. 2.3; C. 2.1; D. 1.4**. Your answer can only include one of options A, B, C or D. 	D
Depth-OC-MV (sentence)	The wardrobe (**red point**) at a depth of 1.0 meters serves as a reference. How deep is the power socket (**blue point**)? Calculate or judge based on the 3D center points of these objects. 	With a central depth of 1.2 meters, power socket is referenced here.

Table 10: Detailed QA of the Distance Inference Object-Object Task

Task	Question	Answer
DistI-OO (fill)	Between cooking pan (**green point**) and plastic bag (**blue point**), which object is positioned closer to coat (**red point**)? Calculate or judge based on the 3D center points of these objects. Submit your response as the name of one object exclusively. 	Cooking pan
DistI-OO (choice)	Which object lies at a closer distance from backpack (**red point**): duffel bag (**green point**) or light switch (**blue point**)? Calculate or judge based on the 3D center points of these objects. Pick the appropriate answer from the options given. **A. duffel bag; B. light switch**. Your answer can only include one of options A, B. 	A
DistI-OO (sentence)	Compare the positions of bed (**green point**) and chair (**blue point**). Which is farer to the heater (**red point**)? Calculate or judge based on the 3D center points of these objects. 	The proximity of heater to bed is 2.0 meters, and to chair, it is 0.6 meters. Hence, the bed is farer to heater.

Table 11: Detailed QA of the Object Spatial Relation Object-Camera Multi-view Task

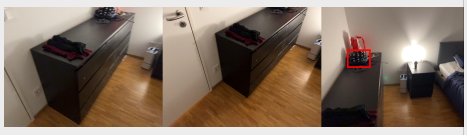
Task	Question	Answer
ObjRel-OC-MV (choice)	What is the direction of object chair (**bbox**) relative to the observer's primary angle? Calculate or judge based on the 3D center points of these objects. We use the first image to reflect the main perspective, which aligns with the observer's viewpoint. The options describe the spatial relationship between object and observer in terms of left-right (left, right, or empty if indistinguishable), above-below (above, below, or empty if indistinguishable), and front-behind (front, behind, or empty if indistinguishable). Select the correct response from the given choices. **A. left, above, front; B. left, above, behind; C. right, below, front; D. right, above, front**. Your answer can only include one of options A, B, C or D. 	C
ObjRel-OC-MV (sentence)	Describe the spatial orientation of object bag (**bbox**) relative to the observer. Calculate or judge based on the 3D center points of these objects. The first image is positioned to serve as the main viewpoint for the observer. 	Relative to the observer's placement, the bag (**red bbox**) appears to the right below. It seems to the front.

Table 12: Detailed QA of the Object Spatial Imagination Object-Camera Multi-view Task

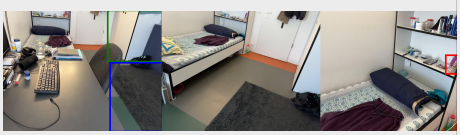
Task	Question	Answer
SpImag-OC-MV (choice)	How does the positional relationship of refrigerator (**red bbox**) to the observer evolve once the observer shifts to the 3D center of map poster (**green bbox**) and faces trash can (**blue bbox**)? Calculate or judge based on the 3D center points of these objects. Base your response on the observer's perspective, with the first image defined as the primary view before movement begins. For multiple-choice questions, consider only the state after the observer has moved. The options describe the spatial relationship between object and observer in terms of left-right (left, right, or empty if indistinguishable), above-below (above, below, or empty if indistinguishable), and front-behind (front, behind, or empty if indistinguishable). Select the appropriate option from the given choices. **A. left, , behind; B. right, below, ; C. right, above, ; D. right, below, behind**. Your answer can only include one of options A, B, C or D. 	D
SpImag-OC-MV (sentence)	How does the observer's shift to the 3D center of wardrobe (**green bbox**) and orientation toward rug (**blue bbox**) affect the positioning of bread packet (**red bbox**)? Calculate or judge based on the 3D center points of these objects. Frame your answer with the observer's perspective, assigning the first image as the main view before any motion. 	Initially, the position of bread packet appears to the left below to the observer. It is also to the front. After moving to wardrobe and orienting toward rug, bread packet changes to to the right below. It is now to the behind.

Table 13: Detailed QA of the View Change Infer Task

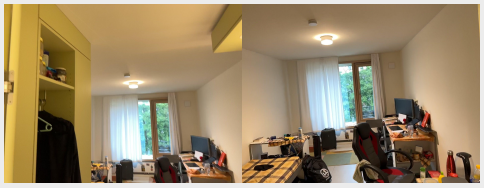
Task	Question	Answer
ViewChgI (fill)	If starting from the first image, how would the observer's camera need to move to recreate the second image? Provide the camera movement and rotation in the following format: move (right or left):(meters), move (down or up):(meters), move (forward or back):(meters), rotate (down or up):(degrees), rotate (right or left):(degrees) - The first three values are in meters. - The last two values are in degrees. - Use commas to separate each parameter. - Do not include any additional text. Example: move left: 2.6, move down: 0.1, move forward: 0.2, rotate up: 10, rotate left: 0. 	move right: 1.2, move up: 0.4, move forward: 1.4, rotate up: 5, rotate left: 90
ViewChgI (sentence)	What changes in position or angle would the camera need to make to transition from the first image to the second? 	In the initial frame, suitcase is located in the bottom-right and moves left and up to the bottom-left in the next image. Observing these changes, it appears the camera movement is as follows: To realign the second image: Move 0.6 meters forward. Shift sideways by 0.2 meters right. Rotate down by 10 degrees. Turn right by 10 degrees.

Table 14: Detailed QA of the Camera Pose Task

Task	Question	Answer
CamPos (fill)	Estimate the image-plane location and depth in meters of the second image's observer as it appears in the first image's coordinate space. Please ensure your answer is limited to a 2D coordinate and a depth, for instance: $(200, 500)$, 1.2. 	$(113, 182)$, 3.8
CamPos (choice)	Where would the second image's observer be seen in the first image's space? Provide 2D image-plane coordinates and depth in meters. Choose the correct response from the given choices. **A. Image Coor: (514, 95), Depth:4.8 meters; B. Image Coor: (258, 537), Depth: 6.9 meters; C. Image Coor: (108, 214), Depth: 6.3 meters; D. Image Coor: (921, 261), Depth: 4.6 meters**. Your answer can only include one of options A, B, C or D. 	A
ViewChgI (sentence)	How does the camera's movement between the two images affect its position in the first image? Provide (X, Y) image-plane coordinates and depth in meters. 	Relative to the first image, the second image's observer occupies the position $(650, 457)$, at a depth of 4.4 meters.