10. Supplementary Material

10.1. Replica Class Mappings

The classes in the Replica dataset differ from those in the Matterport3D dataset used to train the embodied object detectors in this work. To enable testing on the Replica dataset, the following mapping from Replica to Matterport3D classes was used:

```
replica_to_mp3d_mapping = {
    'chair': 'chair',
    'cushion':'cushion',
    'table':'table',
    'pillow':'cushion',
    'cabinet':'cabinet',
    'shelf':'shelving',
    'rack':'chest_of_drawers',
    'sofa': 'sofa',
    'sink': 'sink',
    'base-cabinet':'cabinet',
    'wall-cabinet':'cabinet',
    'bed':'bed',
    'comforter':'bed',
    'desk': 'table',
    'bathtub': 'bathtub',
    'blinds': 'curtain',
    'curtain': 'curtain',
    'monitor': 'tv_monitor',
    'nightstand': 'table',
    'picture': 'picture',
    'toilet': 'toilet',
    'tv-screen': 'tv_monitor'
    }
```

10.2. ScanNet Class Mappings and Scenes

To enable testing on the ScanNet dataset, the following mapping from ScanNet to Matterport3D classes was used:

```
scannet_to_mp3d_mapping = {
    'bathtub': 'bathtub',
    'bed': 'bed',
    'cabinet': 'cabinet',
    'chair': 'chair',
    'desk': 'table',
    'dresser': 'chest_of_drawers',
    'night stand': 'table',
    'pillow': 'cushion',
    'picture': 'picture',
    'sink': 'sink',
    'sofa': 'sofa',
    'table': 'table',
    'toilet': 'toilet',
    'tv': 'tv_monitor',
    'towel': 'towel'
    }
```

The ten apartment and office scenes used for evaluation are *0000_00*, *0040_00*, *0050_01*, *0131_00*, *0207_00*, *0264_00*, *0377_00*, *0377_01*, *0549_01*, *0663_02*.

To enable testing on the ScanNet++ dataset, long-tail class labels that contain a top 100 class are re-labelled with the corresponding top 100 class. For example, *dining chair* is re-labelled as *chair*. The following mapping from ScanNet++ top 100 classes to Matterport3D classes was then applied:

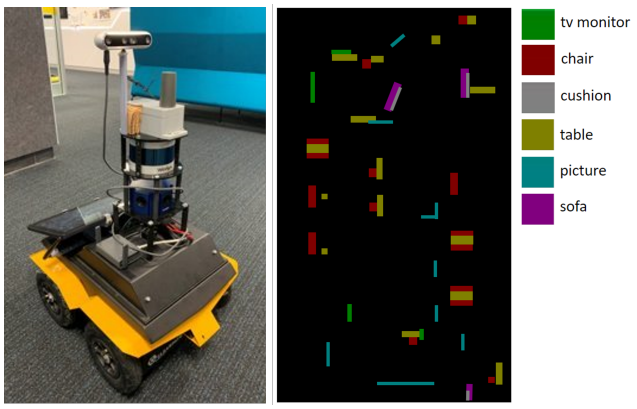

Figure 4. The mobile robot platform used to evaluate embodied object detection under real-world conditions (left). Birds-eye-view semantic map of the office environment used to test embodied object detection performance (right).

```
scannetpp_to_mp3d_mapping = {
    'bathtub': 'bathtub',
    'bed': 'bed',
    'cabinet': 'cabinet',
    'storage cabinet': 'cabinet',
    'kitchen cabinet': 'cabinet',
    'chair': 'chair',
    'office chair': 'chair',
    'desk': 'table',
    'dresser': 'chest_of_drawers',
    'night stand': 'table',
    'pillow': 'cushion',
    'cushion': 'cushion',
    'picture': 'picture',
    'poster': 'picture',
    'painting': 'picture',
    'sink': 'sink',
    'sofa': 'sofa',
    'table': 'table',
    'toilet': 'toilet',
    'monitor': 'tv_monitor',
    'tv': 'tv_monitor',
    'towel': 'towel'
    }
```

The ten apartment and office scenes used for evaluation are *1ada7a0617*, *21d970d8de*, *25f3b7a318*, *27dd4da69e*, *31a2c91c43*, *3864514494*, *38d58a7a31*, *5748ce6f01*, *5942004064*, *5ee7c22ba0*.

10.3. Visualisation of Spatial Feature Memory

To supplement the discussion in the paper, we provide a visualisation of the classification and occupancy confidence implicitly expressed by our spatial feature memory. To visualise classification confidence, we generate class-likelihood scores $s_{u,v} \in \mathbb{R}^{a \times l \times C}$ for each location in spatial memory

Figure 5. Visualisation of the classification entropy $H_{u,v}$ for each feature in spatial memory after all 50 episodes have been processed in Matterport3D scene *YFuZgdQ5vWj_1*. For visualisation purposes, the entropy has been normalised between 0 and 1 with lighter colours denoting lower entropy and hence higher classification confidence.

(u, v) via the following equation:

$$s_{u,v} = softmax\left(\|M_{u,v}\|_2 \cdot z_l\right) \tag{13}$$

where $\|M_{u,v}\|_2$ is the L2 normalisation of the memory feature at location (u, v) and z_l is the class-specific embedding from the shared language-image embedding space. We then calculate the entropy $H_{u,v} \in \mathbb{R}^{a \times l}$ of the predicted class-likelihood scores to quantify the uncertainty at each location:

$$H_{u,v} = -\sum_{i=1}^{C} s^i_{u,v} \ln\left(s^i_{u,v}\right) \tag{14}$$

Figure 5 visualises the classification entropy $H_{u,v}$ after all 50 episodes have been processed in Matterport3D scene *YFuZgdQ5vWj_1*.

To visualise the occupancy confidence $o_{u,v} \in \mathbb{R}^{a \times l}$, we calculate the magnitude of the normalised spatial memory feature $|M_{u,v}|$ using the following equation:

$$o_{u,v} = \sqrt{|M_{u,v}|_1^2 + |M_{u,v}|_2^2 + \cdots + |M_{u,v}|_{d_2}^2} \tag{15}$$

where d_2 is the length of the normalised spatial memory feature $|M_{u,v}|$. Figure 6 visualises the occupancy confidence $o_{u,v}$ after all 50 episodes have been processed in Matterport3D scene *YFuZgdQ5vWj_1*.

10.4. Additional Experiments

10.4.1 Impact of Spatial Resolution

We conduct experiments using the Matterport3D test set to investigate the influence of spatial resolution on the performance of the proposed system. The setting of this parameter

Figure 6. Visualisation of the occupancy confidence $o_{u,v}$ for each feature in spatial memory after all 50 episodes have been processed in Matterport3D scene *YFuZgdQ5vWj_1*. For visualisation purposes, the occupancy confidence has been normalised between 0 and 1 with lighter colours denoting higher confidence.

Table 7. Impact of spatial resolution on detection quality and inference time on the Matterport3D test set. Inference is averaged across all images in the test set. It also does not include geometric projection, as this is handled by a separate data loading thread.

Resolution (m)	AP50	Inference (ms)
0.1×0.1	41.49	99.11
0.2×0.2	41.57	71.54
0.5×0.5	41.31	65.52
1.0×1.0	40.51	63.83

impacts both the accuracy of the detector and the inference time (Table 7). Generally, a high resolution spatial memory will better express the boundaries between different objects, leading to improved detection quality. However, this comes at the cost of increased inference time. In particular, we find that the time to read and write high-dimensional object features to spatial memory increases exponentially with the resolution. To balance these conflicting requirements, we utilise a spatial resolution of $0.2m \times 0.2m$. This setting retains strong detection performance, with minimal increase in inference time. Note that the values in Table 7 do not include geometric projection, as this is handled by a separate data loading thread. Thus, the actual inference time of our system is slightly higher (approximately 100ms) when deployed on the robot.

10.4.2 Systematic Assessment of Sensor Noise

In addition to our real world experiments, we systematically explore the resilience if our system to noise in the geometric projection process (Figure 7). For example, the Intel RealSense D455 camera guarantees an error less than 0.08m at a distance of 4m, while state-of-the-art SLAM systems such as ORB-SLAM3 (2) return localisation accuracy

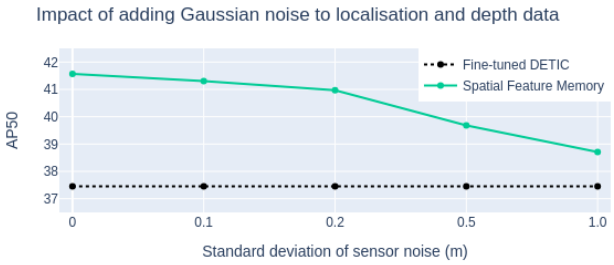

Figure 7. Impact of spatial resolution on the resistance to sensor noise of the proposed system. We vary the resolution of the spatial feature memory across values of 0.1, 0.2 and 0.5. Then, Gaussian noise with a standard deviation of 0.1m is added to the depth and position reading, and 0.01 radians to the robot heading. The standard deviation is scaled by a factor of 2 and 5 until performance degradation is realised. The dotted line represents the performance of the fine-tuned DETIC model, which is unaffected by noise in geometric projection. The remaining lines show the performance of the spatial feature at different resolutions.

within 0.04m on complex scenes. We add Gaussian noise with a standard deviation of 0.1m to the depth and position readings, and 0.01 radians to the heading. This noise is scaled until performance degradation is realised. When using our spatial feature memory under 0.2m of depth and position noise, only a minor performance drop occurs (Figure 7). Evidently, by maintaining a low-resolution representation of the scene, the impact of noise in the projection process is attenuated. This is further helped by our reliance on both image features and spatial memory for performing detection.

10.4.3 Domain Shift During Exploration

We conduct further experiments exploring the performance of our system when low quality detections are initially placed in memory. To simulate this scenario, image quality is degraded using a gamma correction factor of 0.2 during the exploration phase. In this setting, the quality of detections used to establish the memory are significant lower than those seen at test time. As such, we see a drop in performance relative to when exploration is conducted in good conditions. However, our spatial feature memory continues to improve the base detector in this challenging setting (Table 8). We attribute this to the ability of our approach to implicitly express where the spatial memory is reliable (Figure 5, 6). In turn, the detector can learn during training how to effectively balance the use of image and memory features, limiting the potential for erroneous detections to bias the model at future timesteps.

Table 8. Benefit of the spatial feature memory when low quality detections are initially placed in memory. To simulate this scenario, image quality is degraded using a gamma correction factor of 0.2 during the exploration phase, while testing images remain of high quality.

	MP3D	Replica
Fine-tuned DETIC + Memory	**40.62 (+3.17)**	**54.33 (+1.83)**
Fine-tuned DETIC	37.45	52.50

10.5. Hyper-parameter Selection and Sensitivity

10.5.1 Embodied Object Detection

The following hyper-parameters were used to train the embodied object detectors in this work. A batch size of 2 episodes (40 samples) is used to train all methods, as this is the maximum batch size that fits on a single A100 GPU. SGD is used as the optimiser for all experiments. When performing vanilla training, a learning rate of 0.0001 is used. When fine-tuning from DETIC weights, we use a learning rate of 0.00001. The base model is then fine-tuned end-to-end with the spatial memory, using a learning rate of 0.00001 for layers in the base detection network and 0.0001 for new layers associated with the spatial memory. All models are trained for a total of 10000 iterations on the Matterport3D train set, with a checkpoint saved every 1000 iterations. To select a model for testing, the checkpoint with the highest performance on the validation set is selected. Where an earlier checkpoint reaches within 0.3 mAP of the best performing model, it is instead used to mitigate overfitting. Additionally, the following settings are used to implement the spatial memories:

- **Spatial Feature Memory.** We pre-compute the spatial feature memory using the base object detector to speed up training. A confidence threshold of 0.3 is used to select detections for updating memory. The weighting coefficient for fusing memory and image features is set to 5.
- **3D Semantic Mapping.** We closely follow the implementation of (46) to generate the semantic map. A confidence threshold of 0.5 is used to select confident detections for updating the map. The weighting coefficient for fusing memory and image features is set to 50.
- **Implicit Pixel Memory.** We closely follow the publicly available implementation of (3) to build the implicit pixel memory, using a single layer GRU with hidden dimension 256 to update the spatial memory. The weighting coefficient used to combine implicit pixel memory and image features is set to 20.
- **MAMBA External Memory.** We closely follow the publicly available implementation of (49) to imple-

ment the MAMBA external memory, changing only the optimiser, batch size and confidence threshold to align with the other baselines.

The sensitivity of the spatial feature memory, 3D semantic mapping and implicit pixel memory to key parameters on the Matterport3D test set are shown in Figure 8. While there is clear benefit to tuning the weighting coefficients λ, performance remains strong for each method across a range of values. We also assess the sensitivity of the spatial feature memory to the threshold τ_s used to select confident detections for memory update. Performance remains strong across a range of values, showing minor variability when set between 0.2 and 0.5 (Figure 8).

10.5.2 Panoptic Multi-TSDFs

We follow the implementation proposed by (46) to perform semantic mapping. Where possible, we leave all hyperparameters unchanged from the original work. However, several parameters needed to be defined for integration with our embodied object detectors. Firstly, we increase the dynamic voxel size used in the original work from 0.02-0.04mm to 0.03-0.07m to reduce memory consumption. We also implement a threshold τ_s of 0.5 to select confident detections for updating the map. This is slightly higher than our implicit object memory (which uses a value of 0.3) as increasing the number of object instances also increases computational complexity. Specifically, a voxel size of 0.03-0.07m and threshold of 0.5 enables all Matterport and Replica scenes to be mapped using 16Gb of RAM.

We also implement an approach to produce dense segmentation masks m_d^t from the sparse masks m_s^t that result from re-projecting voxels into the image frame. To do so, we apply a mean filter to each segmentation mask, which is commonly used to blur or remove noise from an image. The dense segmentation masks m_d^t are generated simply by applying a threshold τ_m to the blurred mask. The kernel size and threshold τ_m need to be defined based on the voxel size of the map. To explore this relationship, we use the simple scenario of two adjacent voxels of size v that are situated a distance of e from the camera. If the center of each voxel is projected into the camera frame, the pixel distance between the resulting points can be calculate via:

$$d_p = \frac{w}{h_{fov}} * arctan\left(\frac{v}{2e}\right) \tag{16}$$

where d_p is the pixel distance, w is the image width in pixels, and h_{fov} is the horizontal field of view of the camera in radians. For voxels of size 0.05m that are 1m from the camera, an image width on 640 pixels and a horizontal field of view of $\frac{\pi}{2}$, this returns a distance of approximately 10 pixels. Across an entire image, this corresponds to 1 in

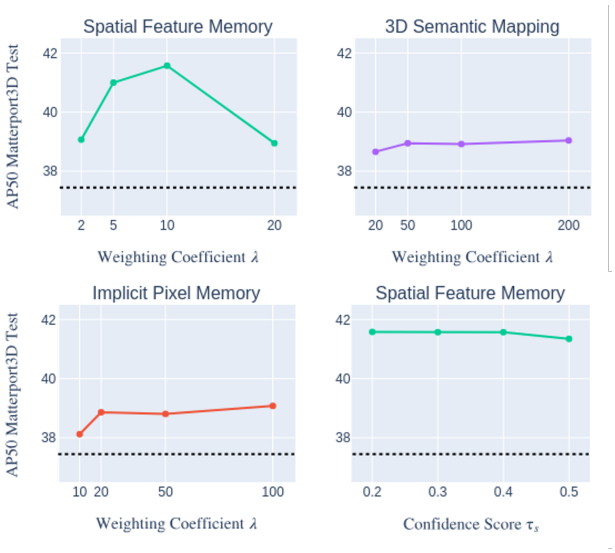

Figure 8. Sensitivity of proposed external memories to key hyperparameters on the Matterport3D test set. In each test, the isolated parameter is swept while keeping all other aspects of implementation constant and fine-tuned DETIC is used as the base model. The dotted line represents the performance of the fine-tuned DETIC model.

100 pixels being associated with a voxel. To associate unlabelled pixels to points of this sparsity, at least one point needs to fall within the range of the mean kernel. Thus, a minimum mean kernel size of 10×10 and threshold τ_m of 0.01 must be used. However, a larger kernel and lower threshold should be used to deal with misclassified voxels, larger voxel sizes and closer objects. The cost of a larger kernel and lower threshold is a reduction in precision at the edge of the segmentation mask. We use a mean kernel size of 50×50 and a threshold τ_m of 0.003 for all experiments to achieve a trade-off between these considerations.