Human-Inspired Topological Representations for Visual Object Recognition in Unseen Environments

Ekta U. Samani[1] and Ashis G. Banerjee[2]

***Abstract*— Visual object recognition in unseen and cluttered indoor environments is a challenging problem for mobile robots. Toward this goal, we extend our previous work [1] to propose the TOPS2 descriptor, and an accompanying recognition framework, THOR2, inspired by a human reasoning mechanism known as object unity. We interleave color embeddings obtained using the Mapper algorithm [2] for topological soft clustering with the shape-based TOPS descriptor to obtain the TOPS2 descriptor. THOR2, trained using synthetic data, achieves substantially higher recognition accuracy than the shape-based THOR framework and outperforms RGB-D ViT [3] on the UW-IS Occluded dataset recorded using commodity hardware. Therefore, THOR2 is a promising step toward achieving robust recognition in low-cost robots.**

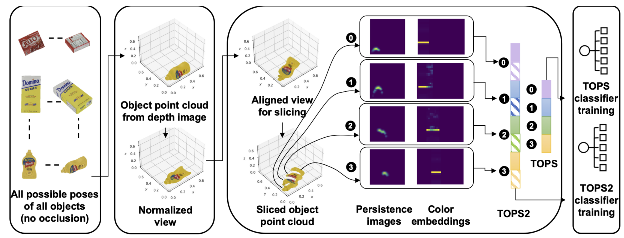
I. Introduction

Object recognition is crucial for successful manipulation of objects in unseen environments. Deep learning-based methods run into difficulties in such environments due to the covariate shift in the data distribution [4]. Further, domain adaptation methods require data from the target domain, and domain generalization methods require abundant real-world training data, which poses deployment barriers on robots with commodity hardware. [5]. Instead, we consider a single synthetic source domain to obtain object representations suitable for recognition across multiple target domains. Considering this paradigm, our previous work [1] proposes a topological descriptor, TOPS, computed from depth images. When used with the human-inspired recognition framework, THOR, it shows promising robustness to occlusions, but recognition using 3D shape alone is challenging [6], [7].

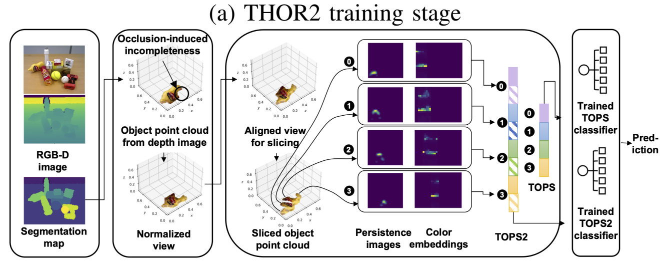
Multimodal convolutional neural networks [8] and transformer-based approaches [3] for recognition using shape and color have been proposed. However, our paradigm requires color representations that transfer well from simulation to the real world. Obtaining them is challenging because the observed chromaticity of objects varies with the lighting conditions [9]. To account for this variation, we follow an approach inspired by the MacAdam ellipses [10] in humans (regions containing indistinguishable colors; we identify the color regions using a topological soft clustering technique, known as the Mapper algorithm [2], and use them to compute the representations. Similar to classical approaches [6], [11]–[15], we then fuse them with TOPS to obtain TOPS2. Our key contributions are:

[1]E. U. Samani was with the Department of Mechanical Engineering, University of Washington, Seattle, WA 98195, USA, during this work `eusamani@gmail.com`

[2]A. G. Banerjee is with the Department of Industrial & Systems Engineering and the Department of Mechanical Engineering, University of Washington, Seattle, WA 98195, USA, `ashisb@uw.edu`



(a) THOR2 training stage



(b) THOR2 testing stage

Fig. 1: Proposed framework, THOR2, for 3D shape and color-based recognition using object unity [18], facilitated by the similarity in the TOPS and TOPS2 descriptors of unoccluded and occluded objects.

- We identify color regions (clusters of similar colors) in the standard RGB color space using the Mapper algorithm [2] and capture their connectivity in a color network.
- We propose a color network-based computation of color embeddings to obtain the TOPS2 descriptor for 3D shape and color-based recognition of occluded objects using an accompanying framework THOR2.
- We show that THOR2, trained with synthetic data, outperforms a state-of-the-art transformer adapted for RGB-D object recognition in unseen cluttered environments.

II. Method: THOR2

Given a real-world RGB-D image of an unseen cluttered scene and the corresponding instance segmentation map [16], [17], our goal is to recognize all the objects in the scene. First, we obtain colored point clouds for every object in the scene and compute the corresponding TOPS and TOPS2 descriptors. The color network used for obtaining TOPS2 is pre-computed (offline) using the Mapper algorithm. We then perform recognition using two classifiers (one for each descriptor) trained on synthetic RGB-D images (see Fig. 1).

A. Color Network Generation

We consider the colors represented by the standard RGB (sRGB) color space, where values for each color channel range from 0 to 255. Let X_{rgb} denote the set of these colors. But, the sRGB color space is not perceptually uniform. Therefore, following [9], we convert all the elements in X_{rgb} to the CIELAB (i.e., $L^*a^*b^*$) color space to obtain X_{lab}.

We then perform topological soft clustering on X_{lab} using the Mapper algorithm. Specifically, let k_{L^*}, k_{a^*}, and k_{b^*} represent the L^*, a^*, and b^* components of a color k in X_{lab}. We use a chroma and hue-based *lens* (i.e., projection) function, f_l, to transform the three-dimensional data in X_{lab} to a two-dimensional space. We define f_l as follows.

$$f_l(k) = \left(\sqrt{k_{a^*}^2 + k_{b^*}^2}, \xi + \arctan\left(\frac{k_{b^*}}{k_{a^*}}\right)\right), \tag{1}$$

where ξ is a constant offset selected based on the cover. Following Chazal et al. [19], we build a cubical *cover* $\mathcal{U}$ by considering a set of regularly spaced intervals of equal length covering the set $f_l(X_{lab})$. Let r_1 and r_2 (where $r_1, r_2 > 0$) denote the lengths of the intervals (i.e., the *resolution* of the cover) along the two dimensions of $f_l(X_{lab})$, respectively. Let g_1 and g_2 denote the respective percentages of overlap (i.e., the *gain* of the cover) between two consecutive intervals. For each $U \in \mathcal{U}$, a clustering algorithm is applied to $f_l^{-1}(U)$ to obtain the *refined pullback cover* $\mathcal{R}$ of X_{lab}. We use the HyAB distance metric [20], defined as follows, to compute the distance between two colors m and n during clustering.

$$\mathrm{HyAB}(m, n) = |m_{L^*} - n_{L^*}| + \sqrt{(m_{a^*} - n_{a^*})^2 + (m_{b^*} - n_{b^*})^2}. \tag{2}$$

Next, the *nerve* of $\mathcal{R}$ is constructed by collapsing each cluster $R \in \mathcal{R}$ into a vertex and creating a p-simplex to represent each $(p+1)$-way intersection of R's. Therefore, in the resulting network, the vertices represent color regions, and the edges represent the overlap between the corresponding color regions. Since a cubical cover does not capture the cyclic nature of the chroma-related dimension of $f_l(X_{lab})$, we add edges connecting the vertices corresponding to the first and last intervals along that dimension. We then eliminate the redundant vertices to obtain the final color network (Fig. 2).

Let $G = (V, E)$ be the color network, representing a non-empty set of vertices V and a set of edges E. Let n_c represent the number of vertices in G, i.e., the number of color regions. We define a similarity matrix, Δ, of size $n_c \times n_c$ to capture the similarity and connectivity between the different color regions in G. We define Δ as follows

$$\Delta = \begin{bmatrix} \delta_{11} & \delta_{12} & \dots & \delta_{1n_c} \\ \delta_{21} & \delta_{22} & \dots & \delta_{2n_c} \\ \vdots & \vdots & \vdots & \vdots \\ \delta_{n_c1} & \delta_{n_c2} & \dots & \delta_{n_cn_c} \end{bmatrix} \tag{3}$$

where $\delta_{i'j'}$ represents the similarity between the i'-th and j'-th nodes. Since every edge in E does not represent the same perceptual difference between the color regions, first, we assign a weight to each edge. Let $\eta_{i'j'}$ be an edge in E that connects the i'-th and j'-th nodes. To assign the weight, first, we compute the mean color [7] of the i'-th and j'-th colors nodes. The edge $\eta_{i'j'}$ is then assigned a weight equal to the HyAB distance between the mean colors of the i'-th and j'-th nodes. We then set $\delta_{i'j'} = \frac{1}{1+l_{i'j'}}$, where $l_{i'j'}$ is the weight of minimum weight path connecting the i'-th and j'-th nodes. This matrix Δ is used for computing TOPS2.

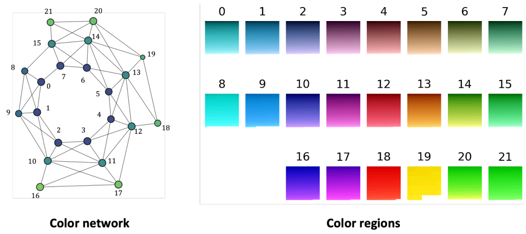


Fig. 2: The color network that captures the connectivity among color regions identified using the Mapper algorithm.

B. TOPS2 Descriptor Computation

Consider a colored object point cloud $\mathcal{P}$ in $\mathbb{R}^3$. Similar to [1], first, we reorient the point cloud to a reference orientation by performing view normalization. Next, we rotate the view-normalized point $\tilde{\mathcal{P}}$ by an angle α about the y-axis to obtain a suitably aligned point cloud $\hat{\mathcal{P}}$. As in [1], we slice $\hat{\mathcal{P}}$ along the z-axis to get slices $\mathcal{S}^i$, where $i \in \mathbb{Z} \cap [0, \frac{h}{\sigma_1}]$. Here, h is the dimension of the axis-aligned bounding box of $\hat{\mathcal{P}}$ along the z-axis, and σ_1 is the thickness of the slices. Let $s = (s_x, s_y, s_z)$ represent a point in $\mathcal{S}^i$. For every slice $\mathcal{S}^i$, we modify the z-coordinates $\forall s \in \mathcal{S}^i$ to s_z', where $s_z' = i\sigma_1$.

Next, we perform further slicing of $\mathcal{S}^i$ along the x-axis to obtain *strips* Ω^j, where $j \in \mathbb{Z} \cap [0, \frac{w}{\sigma_2}]$. Here, w is the dimension of the axis-aligned bounding box of the slice along the x-axis, and σ_2 represents the 'thickness' of a strip. For every strip Ω^j, we obtain corresponding color vectors $\Phi^j = \begin{bmatrix} \phi_1 & \phi_2 & \dots & \phi_{n_c} \end{bmatrix}^T$ as follows.

$$\phi_\lambda = \sum_{\omega \,\in\, \Omega^j} \frac{\mathbb{1}_{X_{rgb}^\lambda}(\omega)}{\sum\limits_{\lambda=1}^{n_c} \mathbb{1}_{X_{rgb}^\lambda}(\omega)}, \tag{4}$$

where $\lambda \in \{1, \dots, n_c\}$ represents the λ-th color region, ω represents the color of a point in Ω^j (in the sRGB color space), $\mathbb{1}$ denotes the indicator function of a set, and X_{rgb}^λ represents the set of colors (in the sRGB color space) belonging to the λ-th color region. Consequently, the color vectors Φ^j approximately represent the color constitution (in terms of the color regions) of the strips Ω^j.

We then stack the color vectors (with appropriate zero padding) to obtain an $n_s^{max} \times n_c$ dimensional color matrix $\mathcal{C}^i$. Let $\mathcal{C}^i = \begin{bmatrix} \mathbf{O} & \dots & \Phi_1 & \Phi_2 & \dots & \Phi_{n_s} & \dots & \mathbf{O} \end{bmatrix}^T$, where n_s is the number of strips in the corresponding slice $\mathcal{S}^i$, n_s^{max} is the maximum number of strips in any given slice, and $\mathbf{O}$ represents a $n_c \times 1$ dimensional zero matrix. Consequently, the color matrix $\mathcal{C}^i$ approximately represents the color constitution (in terms of the color regions) of the slice $\mathcal{S}^i$ in a spatially-aware manner. Last, we obtain a corresponding embedding $\mathcal{E}^i$ as follows.

$$\mathcal{E}^i = (\mathcal{C}^i \Delta)^T \tag{5}$$

Fig. 3 shows this embedding generation for a sample object's slice. We then vectorize the color embeddings and interleave

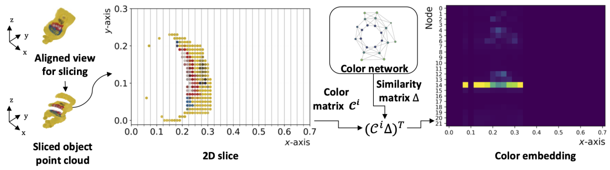


Fig. 3: Visualization of the color embedding computation for obtaining the TOPS2 descriptor. Example of an aligned object point cloud, the slices obtained from it, and the color embedding for one of its slices.

them with the vectorized persistence images from the corresponding TOPS descriptor to obtain the TOPS2 descriptor.

C. THOR2: Training and Testing

Similar to [1], we consider a training set comprising synthetic RGB-D images corresponding to all the possible views of all the objects. We do not consider any object occlusion scenarios in our training set owing to the slicing-based design of the TOPS2 descriptor; it embodies object unity [18], enabling an association between the visible part of an occluded object with the original unoccluded object. We generate colored object point clouds from the RGB-D images, scale them by a factor of σ_s, perform view normalization, and compute the TOPS and TOPS2 descriptors for them. We train one classifier using the TOPS descriptor and another using the TOPS2 descriptor.

When testing on a real RGB-D image of a cluttered scene, first, we generate the individual colored point clouds of all the objects using the instance segmentation maps and scale them by a factor of σ_s. Consider a scaled object point cloud $\mathcal{P}_t$. To recognize $\mathcal{P}_t$, first, we perform view normalization to obtain $\tilde{\mathcal{P}}_t$. Next, we determine if the object corresponding to $\tilde{\mathcal{P}}_t$ is occluded, as described in [1]. If it is occluded, we rotate $\tilde{\mathcal{P}}_t$ by π about the z-axis. We then compute the TOPS and TOPS2 descriptors, and use the corresponding classifier models to obtain two predictions. We choose the prediction with the highest probability as our final prediction.

III. Experiments and Results

We use the CIE standard illuminant D65 for obtaining X_{lab} from X_{rgb} and use the Kepler Mapper library [21], [22] to compute the color network from X_{lab}. We set $\xi = \frac{\pi}{8}$, and build a cover by choosing $g_1 = 10\%$ and $g_2 = 25\%$. We set r_1 and r_2 to divide the corresponding dimensions into three and eight equally-spaced intervals, respectively. For clustering, we use the DBSCAN algorithm [23]. In the case of THOR and THOR2, we use multi-layer perceptrons trained using synthetic training data as described in [1].

We compare the performance of THOR, THOR2, and a Vision Transformer (ViT) adapted for RGB-D object recognition [3] in different environments of the UW-IS Occluded dataset [1] under varying degrees of occlusion. Table I shows that THOR2, which uses both 3D shape and color, achieves higher recognition accuracy than the exclusively shape-based THOR framework in all the scenarios of the UW-IS Occluded dataset. Moreover, THOR2 outperforms

TABLE I: Comparison of mean recognition accuracy (in %) in two different environments of the UW-IS Occluded dataset under varying degrees of occlusion.

Env.	Occlusion	THOR	THOR2	RGB-D ViT [3]		
		Synthetic	Synthetic	Synthetic	S+20% YCB	S+100% YCB
Warehouse	None	51.62 ± 0.53	**61.40 ± 0.37**	43.65 ± 0.70	48.14 ± 1.72	49.39 ± 2.57
	Low	48.07 ± 0.28	**58.00 ± 0.49**	45.11 ± 1.37	47.56 ± 1.62	47.25 ± 3.01
	High	44.26 ± 0.25	**59.38 ± 0.35**	42.52 ± 1.07	48.38 ± 2.37	46.45 ± 2.87
Lounge	None	56.72 ± 0.60	**64.29 ± 0.34**	39.14 ± 2.07	44.72 ± 2.16	46.84 ± 2.66
	Low	54.45 ± 0.24	**65.87 ± 0.64**	43.06 ± 0.60	47.41 ± 0.76	47.51 ± 2.26
	High	51.88 ± 0.46	**59.95 ± 0.52**	43.50 ± 1.07	46.68 ± 1.75	47.11 ± 2.90
All		52.22 ± 0.33	**62.58 ± 0.36**	42.96 ± 0.87	47.04 ± 1.44	47.41 ± 2.70

Note: S + x% YCB indicates that x% real images from the YCB dataset are used along with the entire synthetic dataset for training and validation.

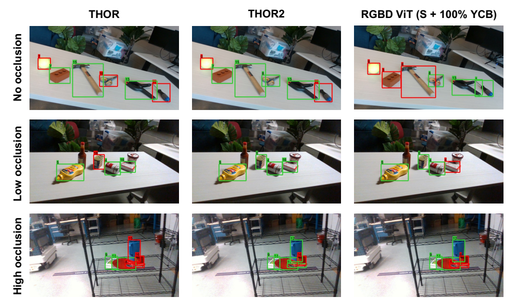


Fig. 4: Sample results (green and red boxes indicate correct and incorrect recognition, respectively) from the UW-IS Occluded dataset.

RGB-D ViT, irrespective of the amount of real-world data (from the YCB dataset [24]) used to train it. Fig. 4 shows a few sample results. These results also demonstrate that the TOPS and TOPS2 descriptors transfer well to the real world, unlike representations learned using an RGB-D ViT trained on synthetic and limited real-world training data. However, we note that similar to THOR, THOR2 faces difficulty in the case of under-segmentation errors and specific heavy occlusion scenarios [1]. We also successfully implement THOR2 on a LoCoBot equipped with an Intel RealSense D435 camera and an NVIDIA Jetson AGX Xavier processor. THOR2 runs at an average rate of $0.7s$ per frame in a scene with six objects on this platform.

IV. Conclusions

This work presents the TOPS2 descriptor and an accompanying human-inspired recognition framework, THOR2, for 3D shape and color-based recognition of occluded objects in unseen indoor environments. In addition to the persistence images in the TOPS descriptor, TOPS2 comprises color embeddings based on the similarity and connectivity among different colors in a color network obtained using the Mapper algorithm. Our slicing-based approach ensures similarities between the descriptors of the occluded and the corresponding unoccluded objects, facilitating object unity-based recognition. Results show that THOR2 benefits from color information and outperforms RGB-D ViT [3] trained using synthetic and limited real-world data. In the future, we plan to extend THOR2 to incorporate multiple viewpoints to improve the recognition of heavily occluded objects.

REFERENCES

[1] E. U. Samani and A. G. Banerjee, "Persistent homology meets object unity: Object recognition in clutter," *IEEE Trans. Rob.*, vol. 40, pp. 886–902, 2024.

[2] G. Singh, F. Mémoli, G. E. Carlsson *et al.*, "Topological methods for the analysis of high dimensional data sets and 3D object recognition," *PBG@ Eurographics*, vol. 2, pp. 091–100, 2007.

[3] G. Tziafas and H. Kasaei, "Early or late fusion matters: Efficient RGB-D fusion in vision transformers for 3d object recognition," in *IEEE Int. Conf. Intell. Robot. Syst.*, 2023, pp. 9558–9565.

[4] E. U. Samani, X. Yang, and A. G. Banerjee, "Visual object recognition in indoor environments using topologically persistent features," *IEEE Rob. Autom. Lett.*, vol. 6, no. 4, pp. 7509–7516, 2021.

[5] P. Antonik, N. Marsal, D. Brunner, and D. Rontani, "Human action recognition with a large-scale brain-inspired photonic computer," *Nat. Mach. Intell.*, vol. 1, no. 11, pp. 530–537, 2019.

[6] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset," in *IEEE Int. Conf. Rob. Autom.*, 2011, pp. 1817–1824.

[7] S. H. Kasaei, M. Ghorbani, J. Schilperoort, and W. van der Rest, "Investigating the importance of shape features, color constancy, color spaces, and similarity measures in open-ended 3D object recognition," *Intell. Service Rob.*, vol. 14, no. 3, pp. 329–344, 2021.

[8] M. Gao, J. Jiang, G. Zou, V. John, and Z. Liu, "RGB-D-based object recognition using multimodal convolutional neural networks: a survey," *IEEE Access*, vol. 7, pp. 43 110–43 136, 2019.

[9] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in Python*. Springer Nature, 2023, vol. 146.

[10] D. L. MacAdam, "Visual sensitivities to color differences in daylight," *Josa*, vol. 32, no. 5, pp. 247–274, 1942.

[11] B. Browatzki, J. Fischer, B. Graf, H. H. Bülthoff, and C. Wallraven, "Going into depth: Evaluating 2D and 3D cues for object classification on a new, large-scale object dataset," in *IEEE Int. Conf. Comput. Vis. Workshops*, 2011, pp. 1189–1195.

[12] L. Bo, X. Ren, and D. Fox, "Depth kernel descriptors for object recognition," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2011, pp. 821–826.

[13] S. S. Bucak, R. Jin, and A. K. Jain, "Multiple kernel learning for visual object recognition: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1354–1369, 2013.

[14] D. Paulk, V. Metsis, C. McMurrough, and F. Makedon, "A supervised learning approach for fast object recognition from RGB-D data," in *Int. Conf. Pervasive Tech. Related Assistive Env.*, 2014, pp. 1–8.

[15] D. Fehr, W. J. Beksi, D. Zermas, and N. Papanikolopoulos, "Covariance based point cloud descriptors for object detection and recognition," *Comput. Vis. Image Understanding*, vol. 142, pp. 80–93, 2016.

[16] C. Xie, Y. Xiang, A. Mousavian, and D. Fox, "Unseen object instance segmentation for robotic environments," *IEEE Trans. Rob.*, vol. 37, no. 5, pp. 1343–1359, 2021.

[17] Y. Lu, N. Khargonkar, Z. Xu, C. Averill, K. Palanisamy, K. Hang, Y. Guo, N. Ruozzi, and Y. Xiang, "Self-supervised unseen object instance segmentation via long-term robot interaction," *arXiv preprint arXiv:2302.03793*, 2023.

[18] S. P. Johnson and R. N. Aslin, "Perception of object unity in young infants: The roles of motion, depth, and orientation," *Cognitive Development*, vol. 11, no. 2, pp. 161–180, 1996.

[19] F. Chazal and B. Michel, "An introduction to topological data analysis: Fundamental and practical aspects for data scientists," *Frontiers Artificial Intell.*, vol. 4, p. 667963, 2021.

[20] S. Abasi, M. Amani Tehran, and M. D. Fairchild, "Distance metrics for very large color differences," *Color Research & Appl.*, vol. 45, no. 2, pp. 208–223, 2020.

[21] H. J. van Veen, N. Saul, D. Eargle, and S. W. Mangham, "Kepler mapper: A flexible python implementation of the mapper algorithm." *J. Open Source Software*, vol. 4, no. 42, p. 1315, 2019. [Online]. Available: https://doi.org/10.21105/joss.01315

[22] ——, "Kepler Mapper: A flexible Python implementation of the Mapper algorithm," Oct. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.4077395

[23] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Int. Conf. Knowl. Discovery Data Mining*, vol. 96, no. 34, 1996, pp. 226–231.

[24] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: Using the Yale-CMU-Berkeley object and model set," *IEEE Rob. Autom. Mag.*, vol. 22, no. 3, pp. 36–52, 2015.