Supplementary Material

A Erorr Analysis across the Video Corpus

Videos in our dataset vary substantially in terms of what OP tasks they involve. This has a large effect over localization accuracy, because it is much harder to localize a carried target than a visible one. To gain more insight into the performance of the leading models, we compare the localization IoU on a video-by-video basis.

Figure S1 depicts per-video IoU of OPNet and two other strong baselines. Each point corresponds to one video and the color reflects the type of frames in that video. Figure S1(a) shows how OPNet outperforms Transformer on videos including *carried* frames (colored in orange). Clearly, videos with carried frames are clustered in the lower half of the figure, where OPNet is superior.

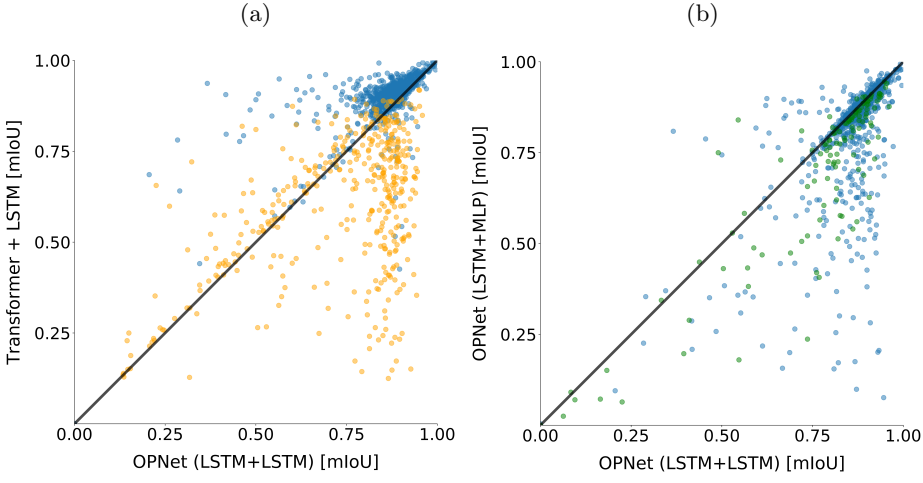

Fig. S1. Sample-by-sample comparison of OPNet with two strong baselines. Each point represents the IoU of a video from the test set, achieved by OPNet and a baseline. **(a)** Videos with more than 7% *carried* frames are colored in orange. **(b)** Videos with more than 7% *occlusion* frames are colored in green. Points in the lower part corresponds to videos in which OPNet is superior.

Similarly, Figure S1(b) compares OPNet with the OPNet (LSTM + MLP) baseline, which contains only the first reasoning component (see Our Approach section). It shows that OPNet outperforms the baseline on videos including a high number of *occlusion* frames (colored in green). It also emphasizes that for

most videos, OPNet is superior, as illustrated by the great number of points in the lower half of the figure.

B Model Comparison

We show two videos comparing OPNet with baselines and other variants. In both videos, four competing methods are applied to the same video scene. We recommend playing videos at a slow speed.

- The first model comparison video[1] shows one visual scene analyzed by four methods. OPNet (ours) successfully localizes the target throughout the video. When the target is "carried", the *Transformer* model (bottom left) fails to switch and instead of tracking the carrying object it keeps predicting the last seen location of the target. The *Tracker* model (top left) switches to a wrong object. The *Heuristic* model (top right) successfully tracks the object containing the target, adjusting well to the target size. See Figure S2(a).
- The second model comparison video[2] shows a visual scene analyzed by four methods. In this video, the target is being occluded by multiple objects, including full occlusion, which makes it challenging to track. The *Tracker*, *Heuristic* and *OPNet MLP* models occasionally drift from the target when it is fully occluded by a large object. OPNet (ours) successfully localizes the target throughout the video. See Figure S2(b).

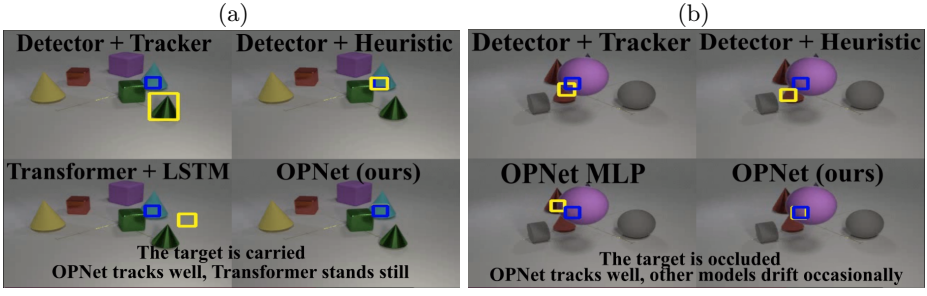

Fig. S2. Screenshots from the model comparison video files. Blue boxes denote the ground truth location. Yellow boxes denote the predicted location. OPNet (ours) is at the bottom right panel. **(a)** The target is contained and then *carried* by the blue cone and is captured successfully by OPNet. **(b)** The target is occluded by the red cone and purple ball. These occlusions confuse all baselines, while OPNet localizes the target accurately.

[1] https://youtu.be/TZgoxoKcGrE
[2] https://youtu.be/KoxbhgalazU

C Qualitative Analysis

Further insight may be provided by comparing the attention mask of the OPNet "Who to Track" module and the ground-truth mask of the containing or carrying object. Figure S3 compares these masks for success and failure cases. It can be seen that OPNet nicely tracks the correct object for most of the frames.

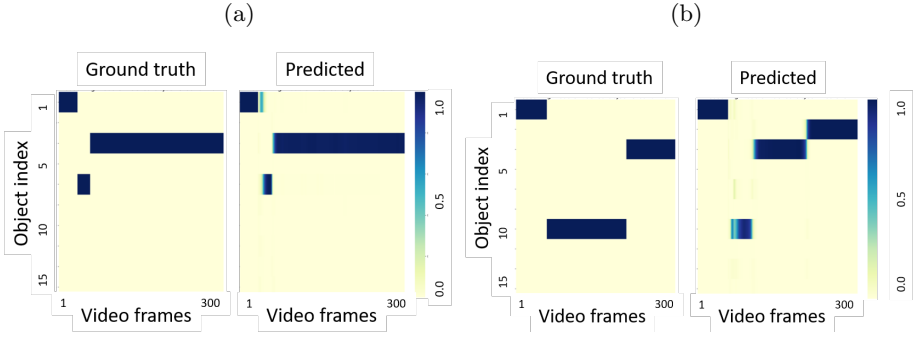

Fig. S3. Switching attention across objects. In each pair of panels, each row traces the probability assigned to an object along the video in the ground truth (left) and predicted attention (right). (a) The system successfully switches attention from object 1 (target) when it is contained by object 6 and then carried by object 3. (b) After a successful switch from the object 1 to 10, the system incorrectly witches to object 3.

D Implementation Details

We trained OPNet and baseline variants using L_1 loss optimized using Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 1e-08$, and using a batch size of 16. We initialized the learning rate to 0.001 and employed a learning rate decay policy, which reduced the learning rate by a factor of 0.8 every 3 epochs without loss improvement. We tuned all hyperparameters using the validation set. We experimented with using a higher initial learning rate of $1e-2$, but it turned out to be too noisy for the relatively small loss induced by the L_1 loss. We also tried lower learning rate ($1e-4$), but it did not converge to a good minimum.

The model was trained for 160 epochs, which we verified via manual inspection to be sufficient for convergence of all models. Early stopping was based on the validation-set mean IoU.

For comparisons with CATER [1] (Table 5 of the main paper), we used the accuracy values reported in their paper.

For the *learning only from visible frames* setup (Section 7.2 and Table 3 of the main paper) we used the values $\alpha = 1$ and $\beta = 0.5$. We used these values to normalize the different scales of $\mathcal{L}_{localization}$ and $\mathcal{L}_{consistency}$. We verified

via manual inspection that (1) for the first 60-70 epochs the loss component $\mathcal{L}_{localization}$ is significantly greater than the loss component $\mathcal{L}_{consistency}$. Thus, in this phase the model improves its prediction when the target is visible; (2) After 60-70 epochs the two loss components have the same scale. Thus, in this phase the model improves its prediction also when the target is not-visible.

E Significance of Performance Improvement

Tables S1 and S2 present the result of two-sided t-tests comparing OPNet against other strong baseline models. Each t-test is designed to test whether the two compared models have the same *mean* IoU, where the mean is computed across all the videos in the test set. Table S1 presents the p-value result of five t-tests comparing OPNet and Transformer on the four OP subtasks, and overall. The results show that OPNet significantly outperforms Transformer on the *contained* and *carried* subtasks, which is the key goal of this paper. OPNet also significantly outperforms Transformer overall with a p-value of 0.01. Table S2 presents the result of two-sided t-tests comparing OPNet with its baseline version, OPNet (LSTM + MLP). The results show that OPNet significantly outperforms its baseline on all subtasks, except for the *carried* subtask, where the difference between models is not statistically significant. OPNet also significantly outperforms the baseline over all frames, with p-value < 0.001.

Table S1. p-values computed using paired t-tests. In all subtasks, except occluded and visible, OPNet outperforms the Transformer baseline, also in the overall score

Subtask	OPNet (LSTM + LSTM)	Transformer + LSTM	p-value
Visible	88.89	**90.82**	**<0.001**
Occluded	78.83	80.40	0.183
Contained	**76.79**	70.71	**<0.001**
Carried	**56.04**	28.25	**<0.001**
Overall	**81.94**	80.27	**0.010**

Table S2. p-values computed using a paired t-tests. OPNet significantly outperforms OPNet (LSTM+MLP) in all subtasks except for carried

Subtask	OPNet (LSTM+LSTM)	OPNet (LSTM + **MLP**)	p-value
Visible	**88.89**	88.11	**<0.001**
Occluded	**78.83**	55.32	**<0.001**
Contained	**76.79**	65.18	**<0.001**
Carried	56.04	57.59	0.429
Overall	**81.94**	78.85	**<0.001**

F LA-CATER Dataset Preparation

Our new LA-CATER dataset augments the CATER dataset [1] with ground-truth locations of all objects and with detailed frame level annotations. Also, instead of using the videos released by CATER we generated new videos using their configuration, and expanded their code to add ground-truth locations and frame-level annotations.

We now describe how we classify frames into the four corresponding OP subtasks. The CATER dataset annotates each frame with the *actions* occurring for each object in that frame. These actions are defined as follows:

- *Slide*. Object changes its position by sliding on the XY-plane.
- *Pick-Place*. Object is picked up in the air along the Z-axis, moved to a new location and placed down.
- *Contain*. A special action performed by cones only, in which cone execute *Pick-Place* action and positioned on top of another object.

- **Contained Frames**. We classify a frame as *Contained* when the target is contained by a cone. Explicitly, a frame is classified as *Contained* when it is annotated with the "contain" action in CATER, with a cone marked as the containing object and the target marked as the contained object. A frame with recursive containment, namely, a containing cone is itself contained by another cone, is also considered to be a *contained* frame. Frames are marked as contained from the moment the target is covered and until the containing object is picked up as part of *pick-place* action.
- **Carried Frames**. We mark a frame as *Carried* when the target is *contained* by a cone (its action is marked in CATER as contained) and *slides* along with it. Frames are marked as *carried* from the beginning of the *slide* action until the end of the *slide* action. Thus, only frames corresponding to the *slide* action are marked as *carried*.
- **Occluded Frames**. For frame t, we define the *occlusion rate (OR)* of object x by object y as

$$OR_t^x(y) = \begin{cases} \frac{Area_t^x \cap Area_t^y}{Area_t^x} & Area_t^x \leq Area_t^y \\ 0 & Otherwise \end{cases} \tag{S1}$$

Where $Area_t^x$ is the area of object x in frame t.
We define the *distance from camera (DC)* of object x

$$DC_t^x = \left\| loc_t^x - loc_t^{CA} \right\|^2 \tag{S2}$$

loc_t^x, loc_t^{CA} denote the 3D coordinates location of object x and the camera in frame t respectively.
We define an indicator for a *fully occluded (FO)* object:

$$FO_t^x = \begin{cases} 1 & \exists\ y\ \ s.t\ \ OR_t^x(y) = 1\ \text{ and }\ DC_t^x \geq DC_t^y \\ 0 & Otherwise \end{cases} \tag{S3}$$

We then mark frame t as *Occluded* when the target is fully occluded by another object. e.g $FO_t^{target} = 1$

- **Visible Frames**. Finally, we define frame as *Visible* when the target is not *Contained*, *Carried* or *Occluded*. Thus, the target needs to be only partially visible to be considered as *visible*. For instance, the target is still considered *visible* when it is 20% occluded (e.g $\exists \;\; y \;\; s.t \;\; OR_t^x(y) = 0.2$)

G Annotating Frames in Perfect Perception

For the perfect-perception setup, we extend the definition of *fully occluded (FO)* objects from Eq S3. We define an object to be *partially occluded (PO)* with respect to the rate p as follows:

$$PO_t^x(p) = \begin{cases} 1 & \exists \;\; y \;\; s.t \;\; OR_t^x(y) \geq p \;\; \text{and} \;\; DC_t^x \geq DC_t^y \\ 0 & Otherwise \end{cases} \tag{S4}$$

We say that object x is non-visible in frame t with respect to p if $PO_t^x(p) = 1$. We use the value $p = 0.7$ to decide which objects are non-visible. Contained objects are defined as non-visible, regardless of their PO value.

Objects are represented by a 5-coordinate vector, containing 4 bounding box coordinates in (x_1, y_1, x_2, y_2) format and an additional visibility bit. Visible objects are represented by their ground-truth bounding boxes and a turned-on visibility bit. Non-visible objects are represented by a four-zeros bounding box coordinates and a turned-off visibility bit.

References

1. Girdhar, R., Ramanan, D.: Cater: A diagnostic dataset for compositional actions and temporal reasoning. arXiv preprint arXiv:1910.04744 (2019)