BabyVLM-V2: Toward Developmentally Grounded Pretraining and Benchmarking of Vision Foundation Models

Supplementary Material

A. Model training

A.1. "baby model" architecture

We build upon the original BabyLLaVA-Llama model introduced in BabyVLM-V1 [59], by giving it the capability to process multiple images as input and conduct multi-turn visual–linguistic interactions. The model architecture consists of a compact language backbone, a visual encoder, and a lightweight multilayer perceptron (MLP) connector that projects visual features into the language space. Unlike BabyVLM-V1, which also experimented with smaller backbones (GPT-2 [45] + ResNeXt-50 [65]), we only adopt the larger variant composed of a LLaMA-1.1B [56, 68] language model and a ViT-L-16 [8] visual encoder (300M params). We find that the smaller variant often struggles to complete complex downstream tasks such as memory, primarily due to its limited model capacity, whereas our current configuration achieves a better balance between developmental plausibility and expressive capability. We also train a larger variant which doubles the size of the language model, and no significant gain is observed; thus, we only use this configuration for the paper.

A.2. "baby model" training paradigm

We train the entire model from scratch using a four-stage pipeline, as summarized in Table 7.

Stage 0: Unimodal Training. In the first stage, the language and vision backbones are trained independently to acquire the basic representational abilities for each modality. The language backbone is trained on all transcribed utterances using a standard autoregressive loss [44]. Its tokenizer is initialized via Byte-Pair Encoding (BPE) [49] trained on the same corpus, with a fixed vocabulary size of 6000. The vision backbone is trained using a DINOv2 [43] objective on SAYCam frames. We do not apply any filtering during this stage—except restricting samples to the training split—since the filtering procedures are primarily designed to enforce image–utterance alignment, which is irrelevant to unimodal representation learning.

Stage 1: Feature Alignment. This stage corresponds to Phase 1 training in LLaVA [31]. Both the vision and language backbones are frozen, and only the MLP connector is optimized using an autoregressive loss. The objective is to align visual features with the language embedding space, effectively bridging the two modalities. To maintain training stability, we use only the image–utterance subset of the pretraining data in this stage, postponing exposure to multi-image inputs until later phases.

Stage 2: Joint Pretraining. In this stage, the vision backbone remains frozen, while the MLP connector and language backbone are trained jointly on the full mixed-format pretraining dataset, as described in Section 3.1. This allows the model to learn multimodal grounding over diverse input structures.

Stage 3: Instruction Fine-tuning. Finally, we fine-tune the model using the mixed instruction dataset, which is a combination of all the instruction samples mentioned in Table 3. This step enables the model to perform various downstream tasks through natural-language prompts. The vision backbone, MLP connector and language backbone are all updated to learn instruction-following behavior and context-dependent reasoning. We apply two different learning rates for different modules in this stage: the learning rate of the vision backbone is 1e-4, while that of the MLP connector and language backbone is 5e-4.

Main hyperparameters of all 4 stages are summarized in Table 7. All experiments are conducted on four NVIDIA A6000 GPUs with 48 GB of VRAM each. Language backbone training completes in less than one hour, while the vision backbone completes in 4 days. Next, training the MLP connector requires approximately 5 hours. Joint pretraining on the mixed-format dataset takes roughly 34 hours to converge. Finally, instruction tuning takes $\sim$24 hours.

A.3. Open-source model fine-tuning

We conduct LoRA finetuning experiments on two open-source models, LLaVA-OneVision-7B and Qwen2.5-VL-7B, to evaluate the effectiveness of our instruction-finetuning dataset. Each task is finetuned separately. We set the LoRA rank to 64, use a scaling factor of 64, and apply a dropout rate of 0.05. Training is performed for 5 epochs with a global batch size of 128, a learning rate of 1e-4, a weight decay of 0.1, a warmup ratio of 0.03, and a cosine learning-rate schedule.

B. Developmentally aligned benchmarks

In Appendix B, we adopt the following organization: Subsection B.1 describes general implementation details that are shared by several tasks, including details on the vocabulary used in *DevCV Toolbox*, acquisition of SAYCam annotations, acquisition of Ego4d annotations, and important distinction between SAYCam and Ego4d. Then, each sub-

[1]The training dataset, the model checkpoints, the training scripts and the evaluation samples will be released to the public in the near future.

Table 7. **Training stage specification of "baby model"**. Note that for stage 3, different modules have different learning rate, as mentioned in Section A.2.

Stage	Trained modules	Frozen modules	Dataset	Loss	Learning rate	Epoch	Global batch size
0-language	Language backbone	N/A	283k utterance only	Autoregressive	2e-4	10	16
0-vision	Vision backbone	N/A	1085k image only	DINOv2	1e-4	100	64
1	MLP connector	Language backbone + vision backbone	768k image-utterance	Autoregressive	3e-3	5	128
2	MLP connector + language backbone	Vision backbone	768k image-utterance + 181k video-utterance + 63k multi-turn	Autoregressive	2e-4	5	128
3	MLP connector + language backbone + vision backbone	None	113k instruction finetune	Autoregressive	5e-4 | 1e-4	5	128

section between 2 and 11 describes how these annotations are used to construct one task in *DevCV Toolbox* each, and are each broken up into *Original Toolbox Task, Adaptation, Data Collection, and Example Prompt*. Some of these also include information on *Evaluation* or *Data Composition*.

B.1. Data collection procedures common to all tasks

Vocabulary filtering

To ensure that all benchmarks in this work focus on developmentally appropriate vocabulary, we draw on the *MacArthur–Bates Communicative Development Inventories (MAB–CDI): Words and Gestures* [37]. The MAB–CDI is a standardized instrument assessing early vocabulary comprehension and production in infants and toddlers, covering familiar words across core semantic categories (e.g., animals, foods, body parts, actions).

Because it is widely regarded as a gold-standard reference for early lexical development, we restrict our benchmark vocabulary to words that appear in—or are closely aligned with—those in the MAB–CDI. Accordingly, during visual concept mining from SAYCam and Ego4D, we retain only crops whose labels fall within this developmentally grounded lexical domain, ensuring that every keyword used across tasks reflects concepts young children could plausibly understand.

SAYCam annotations

To support all SAYCam-based benchmarks in this work, we build the following unified preprocessing pipeline that extracts high-quality image crops for every object and action concept appearing in the corpus. This pipeline is reused (with task-specific modifications described in the corresponding benchmark sections) across tasks and provides consistent visual grounding for all downstream datasets.

- **Frame-level detection and indexing:** We first sample SAYCam videos at 1 FPS and run an open-vocabulary detector (Grounding–DINO [33]) using the GPT-annotated labels associated with each frame as the open set. Let $\mathcal{S}$ denote the set of all such SAYCam labels. For each label $s \in \mathcal{S}$, we construct an index $\mathrm{Index}(s)$ that maps s to all frames in which it is detected, together with its proportionally buffered bounding boxes and GPT-derived blurriness scores. This $\mathrm{Index}(s)$ structure serves as the master lookup table for retrieving visual instances of any concept.
- **Normalizing label variants:** Raw SAYCam labels $s \in \mathcal{S}$ often include plural forms, paraphrases, or compositional descriptions. To ensure consistent visual grounding, we cluster lexically or semantically equivalent labels into small groups based on lexical similarity, plural equivalence, and phrase containment heuristics. Each label s is assigned to its cluster $\mathcal{M}(s)$. This allows us to treat variants of s such as "shoes", "a shoe", or "pair of shoes" as a single underlying concept by retrieving visual instances from $\{\mathrm{Index}(s')|s' \in \mathcal{M}(s)\}$.
- **Quality filtering:** Because SAYCam contains naturalistic video frames from children's head-cam footage, many detections are of low-quality due to motion blur, wrong/irrelevant detector predictions, small/partial bounding boxes. Therefore, we score each detection result using four broad signals: (1) detector confidence, (2) CLIP image–text alignment, (3) crop size, and (4) spatial clarity (e.g., centeredness). These signals are normalized per-concept and combined into a single quality measure. We also employ additional light-weight adjustments to ensure that within $\mathcal{M}(s)$, rare labels are not overwhelmed by frequent ones and that exact label matches are preferred over looser variants.
- **Ensuring lexical and visual diversity:** To avoid selecting many near-duplicate frames of the same scene, we apply simple diversity controls. We first ensure that different lexical variants of a concept are represented, and then enforce a minimal temporal spacing between chosen frames. From this diversified pool, we keep only a small number (≤ 10) of final crops per concept, prioritizing clarity and representativeness.
- **Final output:** The result is a compact, high-quality set of image crops for every object or action concept in SAY-

Table 8. **Comparison between SAYCam and Ego4d**. Object Size is reported in terms of the average % of the frame's area filled.

Data Source	participants	Number of Pixels	Object Size
SAYCam	infants	307k (fixed)	57%
Ego4d	adults	over 2M (average)	4%

Cam. These curated crops act as the visual foundation for the majority of benchmarks built from SAYCam in this paper. They guarantee concept fidelity, diversity of visual contexts, and consistent quality standards across tasks.

Ego4D Annotations

For Ego4D, we do not perform any heavy data cleaning or processing due to the native, high-quality annotations of the dataset. For most of our benchmarks, we use image data from the `egotracks` split, which contains densely annotated egocentric video tracks. In addition, *Picture Vocabulary* (see Section B.2) also draws image crops from `fho_lta`—a subset of Ego4D focused on future hand–object interactions—providing additional object-centric visual diversity.

Overall differences between SAYCam and Ego4d

Here, we analyze the differences between SAYCam and Ego4d which result in "baby model"'s very poor generalization to Ego4d. Specifically, SAYCam was filmed by 3 babies across 4 homes, while Ego4d was filmed by 923 participants across 74 sites. There's also a significant domain shift in the size of the frames and the sizes of the objects relative to the frames. See Table 8 for a summary.

In addition, although all of the Ego4d examples constructed in *DevCV Toolbox* are directly based on objects listed in SAYCam's vocabulary, their backgrounds may still include objects that "baby model" never saw in its training, and thus detract from its' overall understanding of the scene. For example, we might construct an example from Ego4d that asks about the location of a *hand*, and although "baby model" saw examples of *hand* during training, the frame is full of other objects to which "baby model" can attribute no meaning. In such a case, context clues learned by "baby model" about where gloves are usually found relative to their scene, such as at the end of an arm or holding onto a known object, are lost, and performance drops correspondingly. Further, we conjecture that this lack of generalization stems from not only the explicit action categories included in Ego4d that a baby would never have seen (like fixing a car or performing a laboratory experiment), but also from the inherently wider field of view captured adult demonstrators relative to babies. Further, we argue that even if Ego4d had been filmed of the same locations and actions as SAYCam, we would still observe a domain shift caused solely because the demonstrators are adults, perceiving the world from a higher point of view than babies. This point reinforces the uniqueness of the baby domain in the space of egocentric computer vision.

B.2. Picture Vocabulary

Original Toolbox Task

Our task is directly adapted from the NIH Baby Toolbox® Picture Vocabulary Test (PVT), which evaluates a participant's receptive vocabulary by presenting a spoken target word alongside four images (one correct, three distractors) [11]. The goal is to touch the picture matching the target word. Distractors in the original PVT are designed to be *plausible but incorrect*, typically encompassing coarse-categorical, fine-categorical, or phonological similarity. While the full PVT, taken directly from the NIH Toolbox® [12, 15], includes 373 examples, we identify 52 examples intended for early childhood receptive vocabulary evaluation through combining `all-MiniLM-L6-v2` embedding similarity [60] comparison to vocabulary in [37] and manual inspection.

While the Baby Toolbox PVT uses an IRT-based computer-adaptive score that converts response patterns into age-normed ability estimates [15], our adaptation simplifies this to straightforward 4-way accuracy since all items in the benchmark are evaluated rather than adaptively selected.

Our adaptation preserves the original developmental intent while replacing controlled illustrations with naturalistic egocentric visual inputs (SAYCam/Ego4D), providing a grounded benchmark for modeling baby-level vocabulary comprehension in realistic developmental environments.

Adaptation

To adapt the original PVT design to naturalistic corpora, we first map MAB-CDI words $r \in \mathcal{R}$ to corpus vocabularies $\mathcal{S}$: GPT-annotated labels for SAYCam and native objects/actions labels for Ego4D. This produces a set of visually grounded targets $\mathcal{G}_r \subset \mathcal{S}$ for each CDI anchor r, forming a one-to-many mapping $r \to \mathcal{G}_r$.

We then analyze the 52 baby-level NIH PVT items to quantify the original distractor structure. We define three categories: fine-categorical, coarse-categorical, and phonological. We manually annotate every NIH distractor to one or more of these types accordingly. Note that the original PVT includes unrelated distractors and we exclude those given the difficulty in controlling the quality of unrelated distractors in naturalistic imagery. We obtain the unnormalized distractor-type weights:

$$w_{\text{coarse}} = 0.5643, \quad w_{\text{fine}} = 0.1472, \quad w_{\text{phon}} = 0.0321.$$

Using these proportions, we construct corpus-specific distractor pools from the entire corpus $\mathcal{S}$ (because the model is only required to identify the correct target concept, not to correctly recognize or label the distractors):

- **Fine-categorical:** We use similarity scoring based on CLIP text embeddings [46]. For SAYCam, candidates above a similarity threshold of 0.7 is considered belonging to the same fine-grained category while for Ego4D we use a quantile band $[0.997, 0.99973]$ to also filter out overly similar and thus indistinguishable words.
- **Coarse-categorical:** We use Kmeans clustering based on CLIP text embeddings (SAYCam: $K = 100$; Ego4D: $K = 150$).
- **Phonological:** We use Soundex-based string similarity for both datasets.

For each CDI anchor r, we select a ground-truth label $g \in \mathcal{G}_r$ and sample three distinct distractors from these pools using the weights. For SAYCam examples, we perform a final round of manual screening to filter out the infeasible examples, while Ego4D examples are filtered with a hybrid procedure combining `Gemini2.5-flash` checks with a lightweight manual review.

Data Collection

To produce high-quality 4-way visual choices, we collect image crops corresponding to every target and distractor label.

For SAYCam, because *Picture Vocabulary* requires extremely precise, semantically clear images, we modify the fully automated pipeline in Section B.1 with the following changes:

1. Candidates come *directly* from $\mathrm{Index}(g)$ where $g \in \mathcal{G}_r$ given anchor r.
2. Human annotators manually filter irrelevant, ambiguous, or blurry crops and refine bounding boxes, replacing automated quality scores.

This process yields a compact, high-precision crop inventory used for all SAYCam examples.

For Ego4D, for the objects, we use the bounding boxes from the `visual_crop` field of the `EgoTracks` benchmark, applying a deterministic buffer ($1.2\times$ + 8px margin) and requiring a post-buffer normalized area > 0.03. For actions, we use the `fho_lta` benchmark which contains abundant action annotations. As there are no explicit bounding box annotations, we sample frames from the middle 25% of each action frame interval and apply minimal center-biased cropping to maintain clarity. For each label, we keep 10 candidates while preserving diversity and visual fidelity, and we apply a `Gemini2.5-flash` pass to eliminate unusable crops.

Dataset composition

As shown in Figure 7, We obtain 1181 SAYCam examples, covering 344 unique GT labels, 1311 unique distrator labels, and 1660 unique crops. Due to manual filtering, its distractor distribution only loosely follows NIH proportions (shown in Figure 8). Similarly, we obtain 346 Ego4D examples over 124 unique GT labels, 343 unique distractor labels, and 633 unique images (shown in Figure 7) with the corresponding distractor distribution shown in Figure 8.

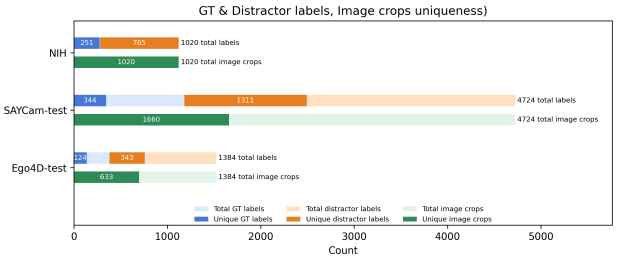

Figure 7. Label/Image crop uniqueness comparison between Picture Vocabulary Test in NIH Baby Toolbox®, SAYCam, and Ego4D.

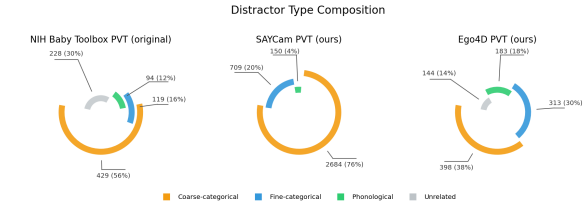

Figure 8. Distractor type composition for the Picture Vocabulary Test in NIH Baby Toolbox®, SAYCam, and Ego4D. The original PVT contains multi-type overlaps, while our sampling assigns each distractor a single type even though some satisfy multiple cues. Ego4D uses unrelated distractors only as a rare fallback.

Example Prompt

Each finalized example is a prompt embedded with 4 image choices for which the following is an example:

```
" Touch the image of 'foot' (A)
<image> (B) <image> (C) <image>
(D) <image> "
```

The model needs to output one of `A`, `B`, `C`, or `D` to be evaluated.

B.3. Looking While Listening

Original Toolbox Task

The Looking while listening test (LwL) from NIH Baby Toolbox® aims to evaluate comprehension for object labeling and receptive language [15]. The infant is shown two clipart images which is followed by an audio prompt describing one of them. Eye tracking is used to detect whether the participant is looking at the ground-truth image. Similar to PVT, we simplify the original metric to accuracy only.

Adaptation

To adapt LwL to our benchmark in SAYCam, We replace

clipart with naturalistic image crops from SAYCam, and eye tracking with multiple choice, similar to *Picture Vocabulary*.

Data collection
Examples for *Looking While Listening* are taken directly from *Picture Vocabulary* examples.

Example Prompt
Each finalized example is a prompt embedded with 2 image choices for which the following is an example:

```
" Touch the image of 'foot'
(A) <image> (B) <image>"
```

The model needs to output one of `A` or `B` to be evaluated.

B.4. Localization

Original Toolbox Task
Much like Picture Vocabulary, the Mullen Receptive Language test #19 tests infants on their ability to point at sketched *target* objects as they are named, avoiding confusing them with the *distractor* objects. Specifically, after gesturing to a group of sketched objects, the psychologist asks: *Look at these. Where is the cat?* If the child points in the direction of the cat, they pass the test.

Adaptation
Localization makes a significant modification to the original NIH Baby Toolbox® measure- In *DevCV Toolbox*, we find it meaningful to test pointing to objects *in their naturalistic environments*, namely, we treat the objects naturally occurring in the background of the frame as distractors rather than inserting unrelated objects. Additionally, because if is infeasible to ask a model to 'point', the answer choices are always *top left, top right, bottom left, bottom right*.

Again, the objects in this task are real objects from SAYCam and Ego4d rather than the sketches used in the NIH Baby Toolbox®, and just like in the NIH Baby Toolbox®, the prompt is the full frame and the name of the object to be localized.

Data collection
The examples for both SAYCam and Ego4d are generated using the centers of the bounding boxes annotated in B.1 and Ego4d's egotracks, respectively.

To avoid including test examples where a bounding box stretches across two answers ambiguously (for example, an object in the bottom middle that could reasonably be called either *bottom left* or *bottom right*), we 1) crop each frame so that its closest corner is flush with the edges of the object's bounding box, and 2) enforce a maximum bounding box area of 1/4 of the frame's area (see Figure 9), which filters out 5.2k of the 7.3k possible test examples. In practice, we find that both of these steps are needed to ensure fair, reasonably unambiguous examples. We enforce no minimum confidence in the SAYCam object annotations and use all object names generated in B.1.

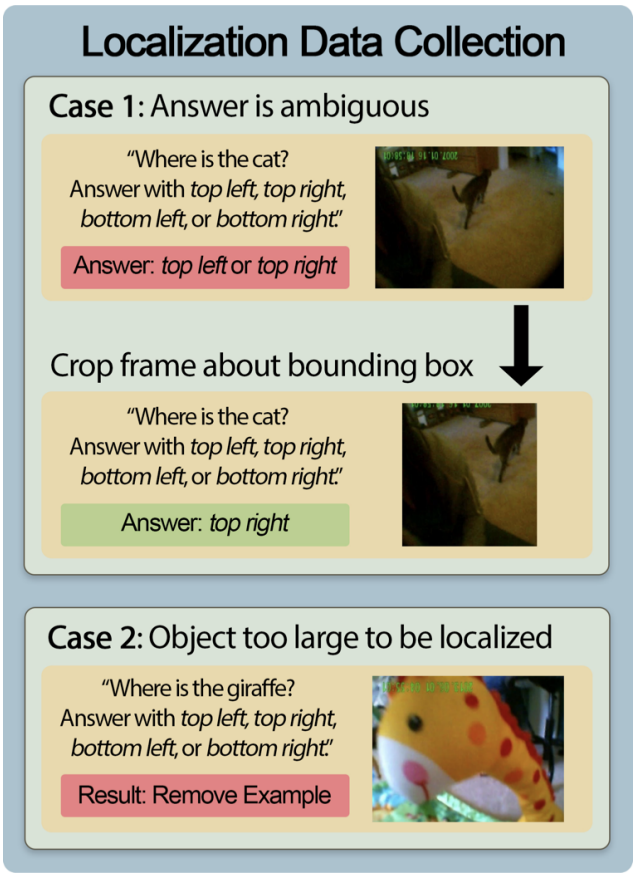

Figure 9

Example Prompt
Each finalized example is a prompt embedded with one image and the same four choices, for which the following is an example:

```
"<image>
Point at the cup.  Is it in (A)
the top left of the image, (B)
the top right, (C) the bottom
left, or (D) the bottom right?"
```

The model needs to output one of `top left`, `top right`, `bottom left`, or `bottom right` to be evaluated.

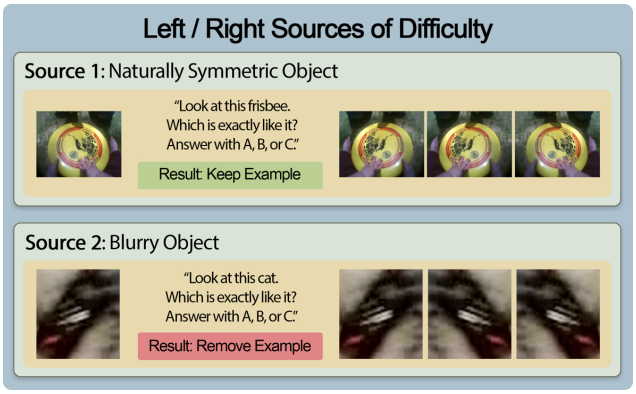

Figure 10

B.5. Left/Right

Original Toolbox Task
Left/Right is adapted directly from Mullen Visual Reception test #29, in which a psychologist shows a child an object, then instructs the child to match it with the identical one. If the child correctly points to the identical object, avoiding confusing it with its own mirror image, they pass the test.

Adaptation
The only modification made while adapting VR test #29 to *DevCV Toolbox* is replacing the clipart objects with real objects from SAYCam and Ego4d. In *DevCV Toolbox*, the basic format is preserved: a prompt image, followed by a correct answer and two distractor choices in some random order, are presented to the model. The target image is a duplicate of the prompt, and the incorrect answers are the mirror image of the target.

Some examples in *Left/Right* are harder than others; we conjecture that difficulty in this task can result from either 1) naturally symmetric objects, or 2) low resolution objects (see Figure 10). Naturally symmetric objects are difficult because they require an encoding of fine-grained details. However, low resolution objects are difficult because even though there might be some spatial clues to discriminate the target from its mirror image, if have models can't ascribe any semantic meaning to the image, they won't encode any semantic meaning to its details. By filtering out small bounding boxes, we aim to remove the examples that are difficult solely due to low resolution.

Data collection
For the SAYCam variant, use the object names and bounding boxes generated in B.1. We enforce no minimum or maximum object size, and for the val and test splits, we enforce a minimum confidence in the bounding box of .85. In both variants, all object crops are zero-padded to (640, 480).

For the Ego4d variant, we use object names and bounding boxes from the published Ego4d egotracks annotations and include only objects that belong to the vocabulary defined in B.1. To remove examples with poor resolution, we require either a minimum bounding box height or width of one fifth of the frame, which filters out about half of the otherwise qualifying examples.

Example Prompt
Each finalized example is a prompt embedded with 1 image prompt and 3 image choices for which the following is an example:

```
"<image>
Which of the following is the
same as this?  (A) <image> (B)
<image>, or (C) <image>?"
```

The model needs to output one of `A`, `B`, or `C` to be evaluated.

B.6. Spatial Details

Original Toolbox Task
Similarly, Mullen Visual Reception test #25 also tests understanding of details in images. In this test, the child is presented with a sketch of a *tulip*, and the psychologist asks: *See this flower. Find one just like this. Look for it here*, while tracing their finger along a page filled with sketches of a *tulip, a sunflower, a clover, and a daisy*. The child is allowed to refer back to the *tulip* while choosing their answer. If the child points to the *tulip*, they pass the test.

Adaptation
Again, the objects in our benchmark are real, cropped objects from SAYCam and Ego4d rather than clipart, and they come from more categories than just *flower*. Additionally, because the models cannot "point" to the choices, the choices are passed as separate images and the correct answer is the index (A, B, or C) of the matching image.

Our final modification to the original measure is that to make it more difficult for a computer, we present the answer choices in their naturalistic backgrounds rather than cropped as in the NIH Baby Toolbox®. In practice, we find the final modification necessary to make *Spatial Details* require a fine-grained understanding of detail, as matching identical images is trivial even for a small vision model.

Data collection
To construct examples from both SAYCam and Ego4d, we match objects with the label, but require that they come from different videos. The labels for each come from B.1 and egotracks, respectively. Note that the same object can appear multiple times within an example- for instance, the same *chair*, captured in two separate videos, can show up

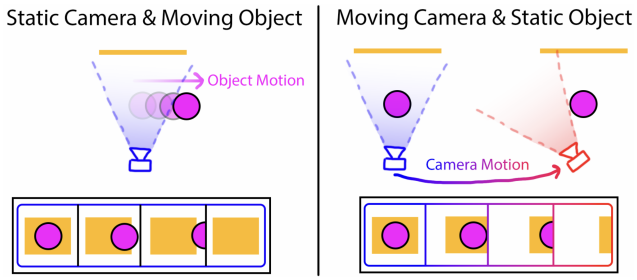

Figure 11. Comparison between sources of occlusion. **Left**: object occlusion from a static camera and moving object. **Right**: object occlusion from a moving camera and static object. Each panel shows a top-down view of the scene along with the corresponding projected 2D video depicting the occlusion event.

as two of the choices. In such cases, the model is forced to rely on spatial details such as orientation, perspective, and lighting, to match identical occurrences.

To ensure quality, we enforce a minimum object confidence of .92 in the SAYCam annotations. To increase difficulty, we also require that objects have an area of less than half of the frame's area.

Example Prompt

Each finalized example is a prompt embedded with 1 image prompt and 3 image choices for which the following is an example:

```
"<image>
Which of the following is the
same as this? (A) <image> (B)
<image>, or (C) <image>?"
```

The model needs to output one of `A`, `B`, or `C` to be evaluated.

B.7. Visual Delayed Response

Original Toolbox Task

Inspired by Visual Delayed Response in the NIH Baby Toolbox®, we introduce an evaluation task designed to assess the spatiotemporal reasoning capabilities of vision-language models. More specifically, our task focuses on object tracking and spatial localization over time, requiring models to process multi-frame/video input to infer spatial trajectory and disappearance of a designated object.

Adaptation

In the original NIH Baby Toolbox® task, a cartoon creature is placed in a frame with grey walls to its left and right. The creature moves from the center of the frame to behind either wall, and the child must identify which wall the creature hid behind. (See Figure 12)

Translating this task to real-world videos is challenging, as the synthetic examples from the toolbox portray an unrealistically ideal scenario. Each toolbox example depicts

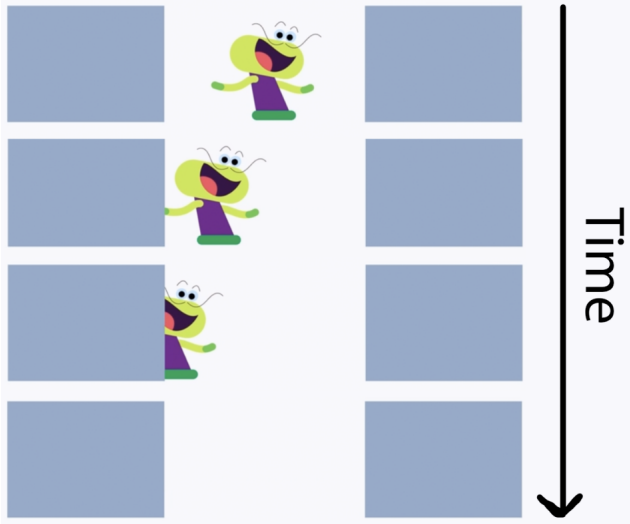

Figure 12. Example of Visual Delayed Response task, taken directly from the NIH Baby Toolbox.

a moving object observed from a static camera perspective, with simplified backgrounds and perfectly smooth motion trajectories. Such controlled scenarios are rare in real-world footage, especially in egocentric videos captured from a toddler's perspective.

To address this challenge we exploit the frequent head movements captured in SAYCam footage, together with the fact that many objects in real-world scenes are largely stationary. By inverting the source of 2D object motion from a static camera with moving objects to a moving camera with stationary objects (see Figure 11), we are able to expand the dataset by over an order of magnitude.

Formally, the model is provided a video $V = \{f_1, f_2, ...f_T\}$ and designated key object k. The video depicts the key object k moving within the field of view and eventually exiting the visible frame at time $t^* \leq T$. The model's objective is to predict the exit region $r \in \mathcal{R}$, where $\mathcal{R}$ denotes the set of possible frame boundaries through which the object may leave.

We define two variants of this task, which differ in the set of selectable exit regions $\mathcal{R}$ provided to the model:

- **Multi-choice setting**: $\mathcal{R}_m$ = {left, right, top, bottom, top-left, top-right, bottom-left, bottom-right}
- **Binary setting**: $\mathcal{R}_b$ = {correct, opposite}, $\mathcal{R}_b \subseteq \mathcal{R}_m$

The multi-choice variant provides a comprehensive set of possible exit regions, where the model is given eight regions as selectable options. The binary variant is a simplified version of the task, where the model only chooses between two options: the correct exit region or the region directly opposite to it.

The overall task can be summarized as a mapping $f_{VDR}(V, k) \rightarrow r$, where $r \in \mathcal{R}$. Here, f_{VDR} represents

the function that, given a video V and designated key object k, predicts the exit region $r \in \mathcal{R}$ through which the object leaves the frame.

Data Collection

SAYCam. Collecting examples for the SAYCam variant of *Visual Delayed Response* can be split into 3 stages: filtering with GPT annotations, filtering with object tracking, and manual labeling.

Stage 1. We first use the 1 FPS annotations provided by GPT in B.1, where each frame is labeled with a "key object" and "objects" attribute. The "key object" denotes a singular object being attended to in a particular frame (if any), and "object" denotes a list of all visible objects within the frame of view. We do an initial filtering for candidate clips by using a sliding window over the 1 FPS frames of each long-range video. For a clip to pass the filter, the first half of frames in the window must have the same "key object", k. In addition, the second half of frames must not have k listed as a "key object" or be present in the "objects" list. From the 422990 initial clips, 17443 are passed as candidate clips to the next stage.

Stage 2. We then perform open-set object detection [58] over the 1 FPS frames sampled from each candidate clip, where the only object class to be detected is the "key object" itself. An object tracking algorithm [70] is also used to track the "key object" over the full fps video. The clips are filtered according to the object tracks, where each track must satisfy all of the following:

- Start within the middle 70% of the frame
- Appear in at least 10 consecutive frames
- Disappear for at least 10 consecutive frames before the full clip ends

To help account for errors in the object detection/tracking, we purposefully loosen the filters and add additional measures for sporadic/false detections. From the 17443 initial clips, 3908 are passed as candidate clips to the next stage.

Stage 3. The final stage involves manually reviewing and hand-labeling each candidate example from the previous stage. We label not only for the ground truth exit direction, but also for a variety of annotations related to overall quality of the clip. In total, we annotate for camera motion, scene visibility, camera stability, occlusion, exit direction, and presence of multiple objects. A breakdown for each is provided as follows:

- **Occlusion**: {Fully Occluded, Partially Occluded, Remains in View}
- **Camera Motion**: {Static, Moving}
- **Direction of Exit**: [Up, Down, Left, Right]
- **Scene Visibility**: {Excellent, Good, Fair, Poor}
- **Camera Stability**: {Very Stable, Stable, Shaky, Very Shaky}
- **Multiple Objects**: {True, False}

We then filter for valid high-quality clips according to the following criteria:

- Object must become fully occluded
- Direction of exit cannot be contradicting (both left & right, or both up & down)
- Scene visibility better than "Poor"
- Camera stability better than "Very Shaky"

From the 3908 initial clips, 2380 are passed as final clips for the dataset.

Ego4D. Data collection for Ego4D follows a very similar structure to the SAYCam process, with the addition of tracked object annotations being already provided by the Ego4D dataset. We use a sliding window over each long-range video's object tracks and filter for all of the following:

- Object is present in first half of window and disappears in second half
- Object bounding box $\geq$ 40000 pixels ($\sim$13% of screen)
- Starts within the middle 50% frame

Each clip is also manually reviewed/labelled according to the same procedure as SAYCam data collection

Multi-frame versions. Since the average clip can range from 100-150 frames, we manually create multi-frame counterparts to each example. More specifically, we look to obtain 1 representative object frame and 3-9 linearly sampled frames that best showcase the object motion/disappearance in a given clip. To do this, we first find the specific frame for three different fields: full object frame, start occlusion frame, and end occlusion frame. The full object frame is always used as the first frame in the multi-frame sequence, and shows the key object in clear view. The start/end occlusion frames mark the interval with which the key object becomes occluded. A random number of frames (3-9) are linearly sampled along this interval to complete the multi-frame sequence for a given clip.

Evaluation

Evaluation is performed over three separate variants: **Exact** and **Adjacent** in the multi-choice setting, and **Binary** in the binary setting (see Figure 13). Accuracy is used as the metric for evaluation, defined as the fraction of predictions considered correct across all trials for a given variant.

In the multi-choice setting:

- **Exact**: Only the labelled ground truth region is counted as correct.
- **Adjacent**: Both the labelled ground truth region and its two adjacent regions are counted as correct. This helps account for small ambiguities in the ground truth label.

In the binary setting:

- **Binary**: A prediction is correct if it matches the "correct" region rather than the "opposite" region.

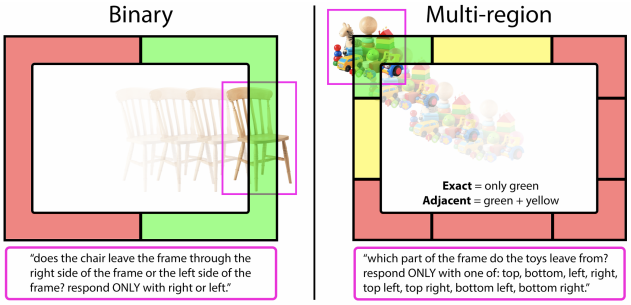

Figure 13. Visualization of evaluation methods for Visual Delayed Response task. **Left**: Binary evaluation for the binary setting, where there is only a correct and opposite incorrect option. **Right**: Exact and Adjacent evaluation for the multi-region setting, where the correct region for Exact is defined by only the green region, and the correct region for Adjacent is defined by both the green and yellow regions.

Example Prompt

Each finalized example includes a series of `<image>` tags or singular `<video>` tag, followed by the prompt. To be properly evaluated, the model must output exactly one option from the choices given in the prompt.
Example from binary setting with multi-frame input:

```
"<image><image><image><image>
does the bottle leave the frame
through the right side of the
frame or the left side of
the frame?  respond ONLY with
'right' or 'left'."
```

Example from multi-choice setting with video input:

```
"<video>
which part of the frame do the
toys leave from?  respond ONLY
with one of:  'top', 'bottom',
'left', 'right', 'top right',
'top left', 'bottom right', or
'bottom left'."
```

B.8. Memory

Original Toolbox Task

The Memory task in the NIH Toolbox is designed to measure how well toddlers (22–42 months old) learn and remember new information using a touchscreen. Children play a short game where they "feed" hungry cartoon animals by touching them on the screen. The test is divided into the learning phase and the test phase.

- **Learning phase:** children see pairs of animals and are told to touch the new animal—the one they have not fed before. They complete 10 trials and receive feedback so they can learn the rules and memorize the animals seen in this phase.
- **Testing phase:** children again see pairs of animals and told to touch the new animal, where each old animal from the learning phase appears twice, each time paired with a different new animal. They complete 20 trials and receive no feedback so correct responses reflect their memory for animals in learning phase.

The animals were selected based on how many 24-month old infants were familiar with them according to data from the MB-CDI Wordbank. Performance is scored based on whether the child touches the correct animal in the testing phase, along with optional reaction time measures to show how quickly they respond.

Adaptation

To simplify the problem and enlarge the potential dataset size, we define the set of word labels used in the learning phase as

$$\mathcal{W}_{\text{learn}} = \{w_1, w_2, \ldots, w_k\},$$

where each w_i corresponds to an image $x_i \in \mathcal{X}_{\text{learn}}$. These image–label pairs (x_i, w_i) serve as the stimuli to be memorized during the learning phase. We further sample $2k$ additional word labels for the testing phase,

$$\mathcal{W}_{\text{test}} = \{w_{k+1}, w_{k+2}, \ldots, w_{3k}\},$$

each associated with a novel image $x_j \in \mathcal{X}_{\text{test}}$.
At each round t, the Vision–Language Model (VLM) receives an input consisting of two images and a text prompt:

$$I_t = \{x_{p_t}, x_{q_t}, P_t\},$$

where x_{p_t}, x_{q_t} are the image inputs and P_t is the corresponding prompt.

- **Learning phase:** The learning phase contains k rounds:

$$I_t^{\text{learn}} = \begin{cases} \{x_1, P_1\}, & t = 1, \\ \{x_{t-1}, x_t, P_t\}, & 2 \le t \le k, \end{cases}$$

  where the two images in the second case are presented in random order. This setup enables the model to incrementally associate visual concepts across consecutive rounds within a single context window.
- **Testing phase:** The testing phase consists of $2k$ rounds, each comparing a learned stimulus with a new one:

$$I_t^{\text{test}} = \{x_{i(t)}, x_{j(t)}, P_t^{\text{test}}\}, \quad x_{i(t)} \in \mathcal{X}_{\text{learn}},\ x_{j(t)} \in \mathcal{X}_{\text{test}}.$$

  Here, $x_{i(t)}$ is a previously seen image and $x_{j(t)}$ a novel one. The model must identify which image corresponds to the new concept described in P_t^{test}.

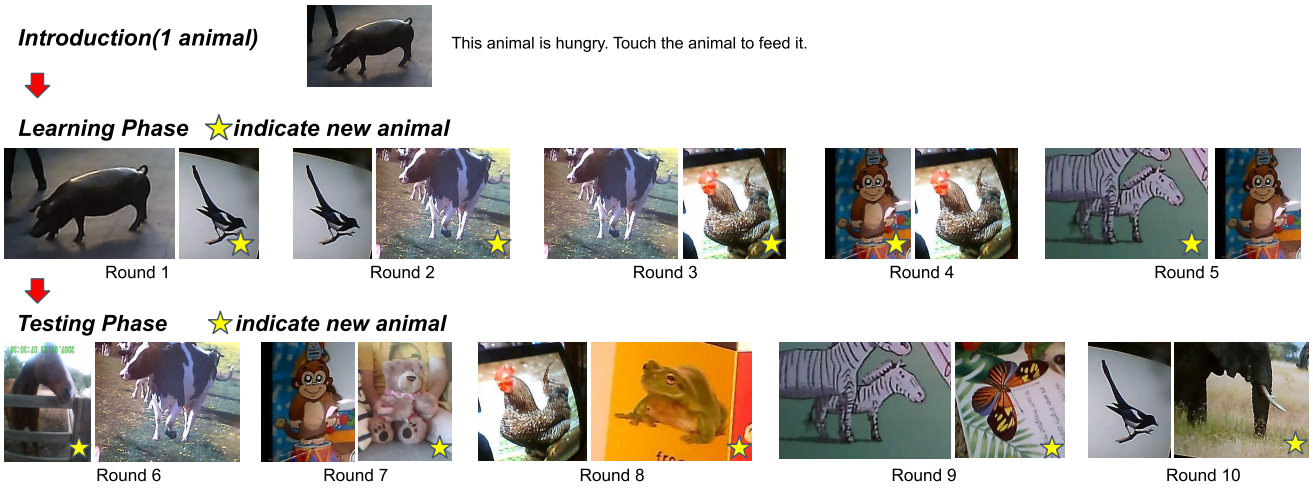

Figure 14. A sample of our memory task adaptation. We use the MAB-CDI words detected in SAYCam as the images to be memorized.

Evaluation

Each learned concept $w_i \in \mathcal{W}_{\text{learn}}$ is paired with two distinct new concepts:

$$(w_i, w_{a(i)}),\ (w_i, w_{b(i)}), \quad a(i), b(i) \in \{k{+}1, \ldots, 3k\}, \quad a(i) \neq b(i), \tag{1}$$

forming two dyads per old stimulus and a total of $2k$ dyads in the testing phase. To mitigate the influence of random guessing, an old stimulus w_i is considered successfully *remembered* only if both of its dyads are answered correctly:

$$r_i = \begin{cases} 1, & \text{if both dyads for } w_i \text{ are correct,} \\ 0, & \text{otherwise.} \end{cases}$$

The overall memory accuracy is then computed as

$$\text{Acc}_{\text{mem}} = \frac{1}{k} \sum_{i=1}^{k} r_i.$$

In all experiments, we set $k = 5$, resulting in a total of $3k = 15$ distinct word–image pairs. This design preserves the spirit of the original Toolbox while adapting the procedure to the VLM's limited context window. When designing the evaluation metric, we follow the structure of the original Toolbox with appropriate simplifications. Specifically, we remove the original intermediate 6–8 min delay settings between the learning and testing phases in our benchmark design. Future extensions may incorporate external memory mechanisms such as Retrieval-Augmented Generation (RAG), or introduce irrelevant contexts between the two phases to simulate real-world temporal gaps. In this work, however, we focus exclusively on assessing the model's in-context retrieval ability.

Data collection

For the scalability of the memory task, we expanded the image set from the cartoon animals in the original Toolbox to the objects in the SAYCam dataset, which also ensures that the items are familiar to children. We used a combination of annotation-based search scripts and automated vision models, including CLIP for object–text similarity and SAM for object segmentation as shown in B.1, to find and isolate frames where these objects appeared clearly. Manual screening was also done after auto-filtering. This process allowed us to gather real-world visual examples of common objects seen by young children, supporting the creation of new learning and memory trials for our benchmark. The visual objects collected from SAYCam dataset will serve as our stimuli in the memory task.

Example Prompt

Each finalized example is a list of prompts each embedded with 2 image choices, for which the following is an example:

```
"Let's try more.
Touch the new image.
(A) <image> or (B) <image>."
```

The model needs to output one of `A` or `B` to be evaluated.

B.9. Who Has More

Original Toolbox Task

In the NIH Baby Toolbox®, the Who Has More Measure is poised as a simple narrative: there are two animals; each of them is pictured with some number of the same object.

Which animal has more?

Adaptation
In *DevCV Toolbox*, we remove the narrative aspect and replace the clipart objects with naturalistic SAYCam and Ego4d objects. In the *Naturalistic* adaptation, the objects are not necessarily identical and appear in their naturalistic backgrounds; in the *Synthetic* adaptation, the objects are perfectly identical, cropped, and pasted onto black backgrounds in matching layouts. The model is prompted to identify whether the *first* or *second* has more.

Data collection
In the synthetic variants, to pick the two quantities to compare, we first sample a number between one and ten. Then, from the numbers remaining that are *lower than the first one*, we sample the second quantity. We do this to ensure a balanced distribution in the *differences in numbers being compared* for each answer. The objects being compared come from the annotations in B.1 and egotracks for SAYCam and Ego4d, respectively.

For the test sets in the naturalistic adaptations, each example is hand-annotated by two separate human experts to cross-validate annotation quality. Specifically, the first human expert labels video frames with an object type and the number of that object. Next, for each of the frames that the first annotator labeled, the second annotator labels the number of the named object in each, *without access to the first annotator's annotation*.

With both labels for each frame, we construct an example for every pair of frames of with objects of the same type for which *both annotators would have arrived at the same answer answer as to which has more had they based their decision solely on their count annotation*. As an example, say the first annotator labels frame A as having 5 cups, and frame B as having 6 cups. If the second annotator labels 5 cups in frame A and 7 cups in frame B, we construct a *Who Has More* example from frames A and B (despite the annotators giving frame B two different labels) because 5<7 *and* 6<7. However, if the second annotator instead labeled frame B as having 5 cups, we *do not* construct a *Who Has More* example from frames A and B, because the two annotators would have given different answers for such an example.

In constructing *Who Has More*, we observe that some objects occur in multiples more than others, and each object follows a unique (and usually nonuniform) distribution of quantity- for example, the number of hands visible in a frame is usually one or two and rarely another number, while an object like books could reasonably be seen in any quantity between one and ten. Additionally, we observe that given the differences in settings and scene perspective, the distributions of object types as well as quantity per object is inherently different for SAYCam and Ego4d.

Example Prompt
Each finalized example is a prompt embedded with 2 image choices for which the following is an example:

```
Which of the following has more
of shoe?  (A) <image>, or (B)
<image>?"
```

The model needs to output one of `A` or `B` to be evaluated.

B.10. Subitizing

Original Toolbox Task
In the NIH Baby Toolbox®, the infant sees one to four colored dots for only one second, then an audio prompt requests the number of dots. Importantly, the dots are not shown for long enough to be counted one at a time- Subitize is intended to measure the ability to *quickly identify small quantities, without counting.*

Adaptation
To construct *Subitizing* in *DevCV Toolbox*, we paste objects onto random locations on black frames, in random quantities between one and four. To simulate the "one second flash", we insert empty frames before and after the frame including the objects.

Data collection
In the SAYCam variant, the objects being pasted come from frames cropped by the bounding boxes obtained in Section B.1, subjected to a minimum confidence of .95. In the Ego4d variant, the bounding boxes come from egotracks, and only objects in the MAB-CDI vocabulary are included.

Example Prompt
Each finalized example is a prompt embedded with 1 blank frame, 1 image prompt, and 1 blank frame for which the following is an example:

```
<image> <image> <image>
How many of apple did you see?
Answer with 1, 2, 3, or 4."
```

The model needs to output one of `1`, `2`, `3`, or `4` to be evaluated.

B.11. Object Counting

Original Toolbox Task
In the NIH Baby Toolbox®, infants are shown some number of an object on a screen, and asked to count them. Unlike the Subitize measure, there is no time limit- participants have time to count each item individually.

Adaptation
In *DevCV Toolbox*, the examples are constructed in the same way as the *Subitizing*examples, except the quantities are between one and twelve, and there are no blank frames corresponding with the lack of a time limit.

Data collection
The data collection for *Object Counting*is the same as for *Subitizing*.

Example Prompt
Each finalized example is a prompt embedded with 1 image prompt, for which the following is an example:

```
<image>
How many of chair did you see?
Answer with a number 1-12."
```

The model needs to output a number between `1` and `12` to be evaluated.

C. Human survey

C.1. Small-scale human adult test

To confirm the validity of *DevCV Toolbox*, we collect small-scale adult performance data on eight of the ten tasks. We omit *Looking While Listening* and *Subitizing* as their examples are directly taken from *Picture Vocabulary* and *Object Counting*, respectively. In total, we have data from n=11 adult participants, each completing 10 trials per task for the SAYCam variants of *Picture Vocabulary*, *Localization*, *Left/Right*, *Spatial Details*, *Visual Delayed Response*, and *Object Counting*, and 5 trials per task for the SAYCam variants of naturalistic *Who Has More* and synthetic *Who Has More*, and as well as the Ego4d variants of all tasks other than *Memory*. Participants completed 30 consecutive rounds of each *Memory* variant, requiring a maximum memory of 29 distinct images.

Results for each task can be found in the *Human performance* rows of Tables 4 and 9. In summary, our participants achieved an average accuracy of 93.0 on all SAYCam tasks and 93.5 on all Ego4d tasks, for both of which they far outperform any model. From this, we conclude 1) *DevCV Toolbox* is a valid discriminator of vision FMs with adult performance as a strong upper bound, and 2) the SAYCam and Ego4d variants have roughly similar complexity and ambiguity for humans.

C.2. Children Helping Science tests

To further examine the developmental fidelity of *DevCV Toolbox*, an IRB review process is currently underway to extend this survey to a *large scale children survey*, where we plan to collect response data for each task from children of the ages recommended for the corresponding NIH Baby Toolbox® measure.

To this end, we collaborated with expert psychologists to develop child-friendly web interfaces for selected tasks and prepared them for deployment on the online developmental research platform Children Helping Science (CHS) [52]. CHS is a widely used, home-based platform through which families can participate in browser-based developmental studies run by researchers worldwide. By adapting our SAYCam-based tasks (PV, VDR and Memory) to CHS, we aim to collect performance from young children under conditions analogous to the NIH Baby Toolbox®. At the time of writing, the studies are under review and not yet live. We show two examples of our task UI design in Figures 15 and 16.

Taking PV as an example (Figure 15), to approximate the modality of the original NIH Baby Toolbox® task, which relies on audio-visual interaction with spoken prompts and observed child responses, we design an *audio&video test page* to verify that instructions and target words can be delivered clearly via audio and that the child's webcam setup is functioning for basic participation monitoring. The *instruction page* provides caregiver-friendly guidance in both text and spoken form. Finally, the *trial pages* present each example in a clean 2×2 grid of four large image options, paired with an audio prompt of the target word, optimizing engagement and accessibility for infants and toddlers while staying faithful to the original task format.

Following the PV setup, VDR also has an initial audio & video test page, along with an instruction page to provide context of the experiment to the caregiver. The trial page for this task (see Figure 16) displays the object that should be tracked, along with the video clip itself and two selectable arrows to submit an answer. Since MP4 with interactive display is not yet supported on the website, a GIF is created in its place. The beginning 5 seconds of the GIF show the first frame with a countdown, then the clip is played as normal and followed by another 5-second buffer to show that the video has ended. To help the caregiver and child understand the experiment, an interactive demo is played as the first 3 trials to showcase how each one should be properly done.

D. Additional experiments & details

D.1. Out-Of-Domain evaluation

To test "baby model"'s capability of generalizing to unseen data domain, we further evaluate it on a set of out-of-domain (OOD) tasks that share the same structure as the in-domain benchmarks but differ in their visual domains. We consider two OOD settings: (1) **Ego4D-based tasks**

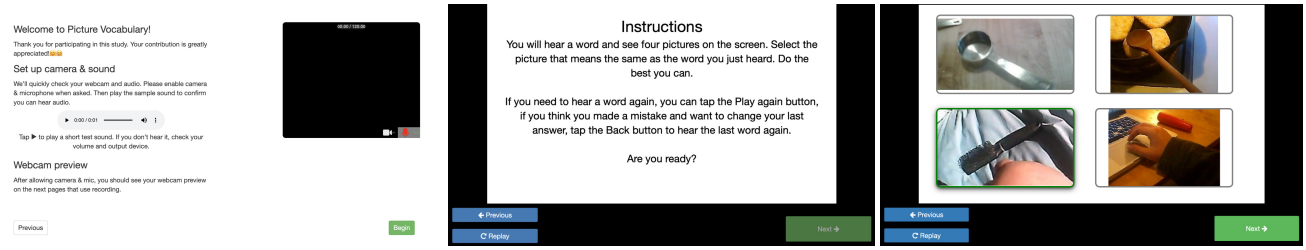

Figure 15. User interface design for our CHS-adapted Picture Vocabulary task.

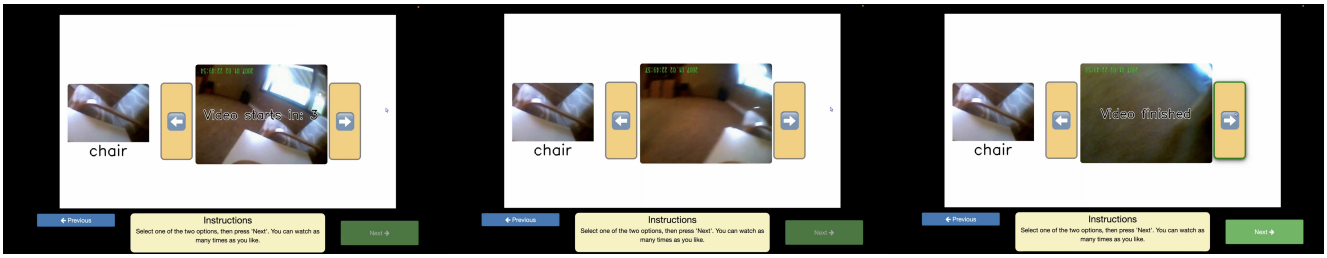

Figure 16. User Interface design for the trial page of Visual Delayed Response task on CHS.

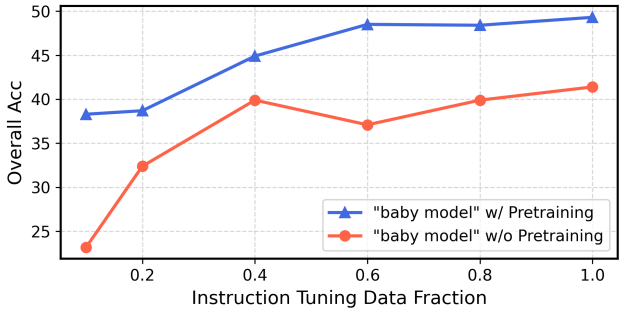

Figure 17. *DevCV Toolbox* overall performance on different instruction tuning data fraction. Results are reported on epoch 1.

use egocentric videos from the Ego4D dataset [13], which remain first-person and naturalistic but introduce distinct environments and contexts. (2) **BabyToolbox-based tasks** correspond directly to standardized developmental psychology and clinical assessments, where the visual stimuli are abstract, non-egocentric cartoon images. The detailed test results are reflected in Table 9 and Table 10.

D.2. Importance of the pretraining stage

To evaluate the contribution of the pretraining stage, we compare two variants of "baby model": (1) the full model trained with Stage 0–2 pretraining before instruction tuning, and (2) a randomly initialized model that skips pretraining and is trained only with Stage 3. For both variants, we fine-tune using different fractions of the instruction dataset and evaluate each model on in-domain tasks.

As shown in Figure 17, the pretrained model consistently outperforms the non-pretrained variant across all data fractions. The gap is especially pronounced when the instruction data is limited, demonstrating that pretraining provides a strong and sample-efficient initialization for downstream instruction tuning. As the instruction data fraction increases, both models improve, reflecting a clear scaling-like trend qualitatively consistent with observations in large-scale model studies [17, 22]. This suggests that data-dependent performance gains also exist in compact, developmentally inspired models, while pretraining remains a crucial component for achieving data-efficient learning.

D.3. Effect of preprocessing pretraining dataset

To further study the effect of applying various filters to curate the pretraining dataset, we constructed two more pretraining datasets: one without any preprocessing (no filter), and the other is preprocessed by stricter filters, keeping less than half the size of the original pretraining dataset. The evaluation results of "baby model" pretrained on these additional pretraining datasets are shown in Table 11. "baby model"-no filter degrades the performance a lot, which verifies the effectiveness of our pretraining dataset preprocessing pipeline; "baby model"-strict filter shows some further improvement, suggesting potential room to refine the data preprocessing pipeline.

D.4. Instruction fine-tuning strategy ablation

We compare two fine-tuning strategies using our instruction tuning data on "baby model": (1) fine-tuning a separate

Table 9. **Performance comparison across models on *DevCV Toolbox* out-of-domain tasks (Ego4D).** Different background colors denote different model families. We report accuracy (%) for all tasks.

Model	Overall	Count	LeftRight	Spatial	PV	Memory	Localization	Visual Delay Response			Who Has More	
								binary	multi-exact	multi-adjacent	synthetic	naturalistic
Upper Bound												
Human performance	93.5	96.4	98.2	96.4	96.4	98.8	90.9	100	58.2	100	100	92.7
Proprietary models												
GPT-4o	67.6	62.1	45.1	94.7	85.3	**100**	80.4	45.5	13.2	48.3	84.3	84.6
GPT-5	86.7	77.5	**88.0**	**96.8**	91.9	**100**	88.7	**94.4**	**50.3**	82.6	94.6	88.5
Gemini-2.5-flash	77.7	72.9	49.6	86.7	**92.5**	99.2	88.4	80.6	37.1	70.2	**97.8**	80.1
Gemini-2.5-pro	**88.2**	**81.9**	**88.0**	94.8	91.9	**100**	**90.2**	91.3	**50.3**	**87.9**	96.5	**97.8**
Open-source models - Similar Size as Ours												
LLaVA-OneVision-0.5B	39.4	43.9	32.6	33.3	27.7	22.6	21.6	73.0	15.2	67.7	46.8	49.4
InternVL3.5-1B	43.7	34.7	34.0	34.1	33.8	24.9	60.7	73.9	16.9	68.5	49.0	49.9
Qwen2.5-VL-3B	48.1	35.7	32.6	44.1	41.9	25.7	86.7	79.8	28.9	51.1	50.2	53.4
Ours												
"baby model"	48.8	42.1	82.2	47.1	23.4	72.4	27.7	39.9	22.8	36.2	92.5	50.9
Lower Bound												
Random guess	31.8	8.33	33.3	33.3	25.0	25.0	25.0	50.0	12.5	37.5	50.0	50.0

Table 10. **Performance on NIH Baby Toolbox out-of-domain tasks.** We report the #correct/#total for all tasks.

Model	Who Has More	Count	Mullen Visual Reception
"baby model"	13/24	2/6	3/12

model for each task, and (2) jointly fine-tuning a single model on the merged instruction dataset. Table 12 presents the comparison. Overall, the task-specific strategy performs slightly better than the mixed strategy. However, the performance gap is generally small across most tasks. Notably, certain tasks, such as *Left/Right*, *Spatial Details*, and *Memory*, benefit significantly from mixed fine-tuning, suggesting potential knowledge transfer or regularization effects across tasks.

D.5. Synthetic caption generation

We study the impact of noisy visual-alignment in the naturalistic child-directed utterances transcribed in the pretraining dataset by replacing them with video captions generated by GPT-4o. To encourage diversity in the generated captions and ensure they remain close to the style of the original dataset, we include 10 randomly sampled transcriptions in each prompt. The transcriptions are sampled from a pool of the 1,000 highest confidence transcriptions in the original dataset that contain at least one noun and more than three words. These heuristic filters help ensure that the sampled transcriptions contain stylistic information rather than simple phrases that are common in the dataset like "wow" or "let's go". The pool of 1,000 transcriptions are manually screened to remove uninformative transcriptions that passed the filtering step. The full prompt to GPT-4o is shown in Figure 18 and an example of a generated caption is shown in Figure 19.

```
messages =
[{"role": "system", "content": f"You will be given the frames from a short video clip. Your task is to write a short, accurate caption for the video (1 sentence max). In addition, you should only use realistic child-directed language, like something a parent would say to a toddler. The sentences should be easy to understand for a two-year-old child. Avoid specific names of people, places, or brands. You should also stay in the style of the existing child-directed captions that we have transcribed. To help you with this, you will be given 10 captions from the dataset. Make sure to adhere to their style while still grounding in the visual content of the frames. Here are the 10 captions: {sampled_captions}"},
{"type": "image_url", "image_url": {video_frames}},
{"role": "user", "content": f"Do not use ambiguous or hedging phrases. You should also not output phrases like 'I cannot see anything' or 'I don't know'. If you cannot tell what is happening in the video, make your best guess. Output only the caption (1 sentence max) with no other text: "}]
```

Figure 18. Full prompt for pretraining data ablation

D.6. Prompting experiment

Finally, we complete a prompting experiment to show the stability of *DevCV Toolbox* examples with respect to commercial models, the results of which are shown in Table 13. We select *Left/Right* and *Object Counting* for this experiment, as we found that commercial models had the lowest and most variable performance on these. For both tasks, 100 examples are randomly selected and presented to Gemini-2.5-flash with a standard prompt, a one-shot prompt, and two variations of the standard prompt, called *alternate prompt 1* and *alternate prompt 2*. The standard

Table 11. **Ablation of pretraining data filtering threshold on *DevCV Toolbox.*** "baby model" is the model pretrained on the regular pretraining dataset described in the main paper, "baby model"-no filter and "baby model"-strict filter are models pretrained on different pretraining datasets described in Section D.3.

Model	Overall	Count	LeftRight	Spatial	PV	Memory	Localization	Visual Delay Response			Who Has More	
								binary	multi-exact	multi-adjacent	synthetic	naturalistic
"baby model"	63.9	47.3	96.4	92.8	32.4	90.8	37.8	54.6	38.1	**52.5**	99.7	60.5
"baby model"-no filter	56.0	46.5	41.8	**95.9**	32.6	63.1	38.2	54.2	36.1	49.5	**99.8**	58.6
"baby model"-strict filter	**67.3**	**53.2**	**99.7**	95.8	**33.3**	**94.0**	**44.7**	**58.5**	**40.2**	52.0	**99.8**	**68.8**

Table 12. **Two supervised fine-tuning strategies**. "baby model"-separate denotes models fine-tuned on each task's instruction dataset separately, and "baby model"-mixed is a single model fine-tuned on the mixed instruction set.

Model	Overall	Count	LeftRight	Spatial	PV	Memory	Localization	Visual Delay Response			Who Has More	
								binary	multi-exact	multi-adjacent	synthetic	naturalistic
"baby model"-separate	56.9	45.2	54.9	79.8	**32.5**	73.6	36.3	**55.7**	37.0	49.9	98.6	**62.1**
"baby model"-mixed	**63.9**	**47.3**	**96.4**	**92.8**	32.4	**90.8**	**37.8**	54.6	**38.1**	**52.5**	**99.7**	60.5

Figure 19. Example of caption generated by GPT-4o

Table 13. **Comparison between Gemini-2.5-flash performance with different prompting strategies**

Prompt Type	Count	LeftRight
standard	69	55
one-shot	66	82
alternate prompt 1	55	54
alternate prompt 2	67	56

prompt is the one used in all other experiments, and the one shot-prompt is a prompt that includes one other example, with its correct answer, prepended to the standard prompt.

For *Object Counting*, *alternate prompt 1* does not give the object's name to be counted, e.g. `"<image> How many objects do you see?"`, which we see drops performance, which is intuitive because large models thrive on context, in this case the name of the object to be counted. *Alternate prompt 2* gives more detail, e.g. `<image> count the flora very closely, starting from one. Keep track of which ones have already been counted and what number you've counted to thus far. Then, report how many flora you counted."`. Unsurprisingly, *alternate prompt 2* does not improve performance, showing that 1) the *standard* prompt was sufficient and 2) Gemini-2.5-flash has capable instruction-following capabilities. For *Object Counting*, we find that a one-shot prompt does not boost performance.

For *Left/Right*, the *standard* prompt gives each image token interleaved with their answer labels, e.g. `"<image> Which of the following is the same as this? (A) <image> (B) <image> (C) <image>"`. In *alternate prompt 1*, we undo this interleaving, resulting in `"<image><image><image><image> Which of the following is the same as the first one? (A) the second one, (B) the third one, or (C) the fourth one?"`. In *alternate prompt 2*, we interleave even more, by giving some descriptive text before the prompt image, e.g. `"Here is an image: <image>. Which of the following is the same as it? (A) <image>, (B) <image>, or (C) <image>?"`. Intuitively we expect *alternate prompt 2* to be the easiest, *alternate prompt 1* to be the hardest, and the *standard* prompt to fall in between. However, we find that none of these prompts elicits significantly different performance, however, the one-shot prompt *significantly* boosts performance. These two findings show the robustness of Gemini-2.5-flash, and the complexity of *Left/Right*, respectively.

D.7. Effect of reasoning chain on proprietary models

Several proprietary models evaluated on *DevCV Toolbox* rely on built-in reasoning chains by default, including Gemini-2.5-Flash/Pro and GPT-5. To investigate the effect

of reasoning depth, we vary the *thinking budget*, a hyperparameter specifying the number of tokens to use for reasoning, of Gemini-2.5-Flash from 0 to 25,000 tokens. The results reveal a striking task-dependent divergence: performance on *Spatial Details*, which demands fine-grained perceptual matching, monotonically increases with thinking budget, from 87.5% to 92.1%. In contrast, *Who Has More* (Naturalistic), which relies on direct visual perception, sees performance decrease, dropping from 92.0% at zero budget to 87.5% at the maximum budget. This suggests that *DevCV Toolbox* tasks differ meaningfully in the reasoning depth they require, and that extended chain-of-thought is not universally beneficial as pure perceptual tasks may be better served by direct, fast responses.

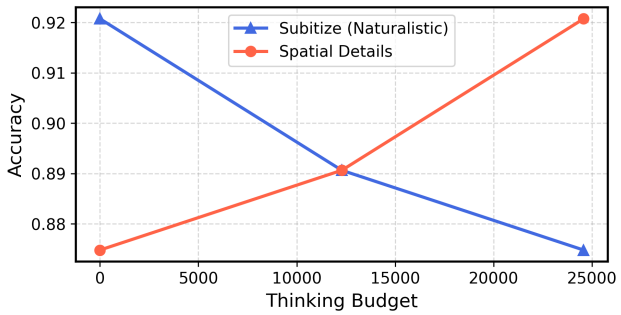

Figure 20. Gemini-2.5-Flash accuracy on *DevCV Toolbox* tasks across different thinking budget configurations.

E. Ethical considerations

SAYCam provides a clear license and practice guidelines for us to follow throughout this work. Whenever ambiguities arose, we consulted directly with the SAYCam authors, who graciously served as consultants to the project through multiple meetings. We have confirmed that the experiments and data collection procedures described in this paper do not require IRB approval. However, the large-scale children survey on *DevCV Toolbox* that we plan as future work does require IRB, and we are actively working with a psychologist and the IRB office to obtain the necessary approvals.