Modality-invariant Visual Odometry for Embodied Vision

Marius Memmel$^1$* Roman Bachmann$^2$ Amir Zamir$^2$

$^1$University of Washington $^2$Swiss Federal Institute of Technology (EPFL)

https://vo-transformer.github.io

Abstract

Effectively localizing an agent in a realistic, noisy setting is crucial for many embodied vision tasks. Visual Odometry (VO) is a practical substitute for unreliable GPS and compass sensors, especially in indoor environments. While SLAM-based methods show a solid performance without large data requirements, they are less flexible and robust w.r.t. to noise and changes in the sensor suite compared to learning-based approaches. Recent deep VO models, however, limit themselves to a fixed set of input modalities, e.g., RGB and depth, while training on millions of samples. When sensors fail, sensor suites change, or modalities are intentionally looped out due to available resources, e.g., power consumption, the models fail catastrophically. Furthermore, training these models from scratch is even more expensive without simulator access or suitable existing models that can be fine-tuned. While such scenarios get mostly ignored in simulation, they commonly hinder a model’s reusability in real-world applications. We propose a Transformer-based modality-invariant VO approach that can deal with diverse or changing sensor suites of navigation agents. Our model outperforms previous methods while training on only a fraction of the data. We hope this method opens the door to a broader range of real-world applications that can benefit from flexible and learned VO models.

1. Introduction

Artificial intelligence has found its way into many commercial products that provide helpful digital services. To increase its impact beyond the digital world, personal robotics and embodied AI aims to put intelligent programs into bodies that can move in the real world or interact with it [15]. One of the most fundamental skills embodied agents must learn is to effectively traverse the environment around them, allowing them to move past stationary manipulation tasks and provide services in multiple locations instead [40]. The ability of an agent to locate itself in an environment is vital to navigating it successfully [12, 64]. A common setup is to equip an agent with an RGB-D (RGB and Depth) camera and a GPS+Compass sensor and teach it to navigate to goals in unseen environments [2]. With extended data access through simulators [28, 39, 40, 47, 57], photorealistic scans of 3D environments [7, 28, 46, 56, 58], and large-scale parallel training, recent approaches reach almost perfect navigation results in indoor environments [55]. However, these agents fail catastrophically in more realistic settings with noisy, partially unavailable, or failing RGB-D sensor readings, noisy actuation, or no access to GPS+Compass [6, 64].

Visual Odometry (VO) is one way to close this performance gap and localize the agent from only RGB-D observations [2], and deploying such a model has been shown to be especially beneficial when observations are noisy [12, 64]. However, those methods are not robust to any sensory changes at the test-time, such as a sensor failing, underperforming, or being intentionally looped out. In practical applications [43], low-cost hardware can also experience serious bandwidth limitations, causing RGB (3 channels) and Depth (1 channel) to be transferred at different rates. Furthermore, mobile edge devices must balance battery usage by switching between passive (e.g., RGB) and active (e.g., LIDAR) sensors depending on the specific episode. Attempting to solve this asymmetry by keeping...
separate models in memory, relying on active sensors, or using only the highest rate modality is simply infeasible for high-speed and real-world systems. Finally, a changing sensor suite represents an extreme case of sensor failure where access to a modality is lost during test-time. These points demonstrate the usefulness of a certain level of modality invariance in a VO framework. Those scenarios decrease the robustness of SLAM-based approaches [32] and limit the transferability of models trained on RGB-D to systems with only a subset or different sensors.

We introduce "optional" modalities as an umbrella term to describe settings where input modalities may be of limited availability at test-time. Figure 1 visualizes a typical indoor navigation pipeline, but introduces uncertainty about modality availability (i.e., at test-time, only a subset of all modalities might be available). While previous approaches completely neglect such scenarios, we argue that explicitly accounting for “optional” modalities already during training of VO models allows for better reusability on platforms with different sensor suites and trading-off costly or unreliable sensors during test-time. Recent methods [12, 64] use Convolution Neural Network (ConvNet) architectures that assume a constant channel size of the input, which makes it hard to deal with multiple "optional" modalities. In contrast, Transformers [51] are much more amenable to variable-sized inputs, facilitating the training of models that can optionally accept one or multiple modalities [4].

Transformers are known to require large amounts of data for training from scratch. Our model’s data requirements are significantly reduced by incorporating various biases: We utilize multi-modal pre-training [4, 17, 30], which not only provides better initializations but also improves performance when only a subset of modalities are accessible during test-time [4]. Additionally, we propose a token-based action prior. The action taken by the agent has shown to be beneficial for learning VO [35, 64] and primes the model towards the task-relevant image regions.

We introduce the Visual Odometry Transformer (VOT), a novel modality-agnostic framework for VO based on the Transformer architecture. Multi-modal pre-training and an action prior drastically reduce the data required to train the architecture. Furthermore, we propose explicit modality-invariance training. By dropping modalities during training, a single VOT matches the performance of separate unimodal approaches. This allows for traversing different sensors during test-time and maintaining performance in the absence of some training modalities.

We evaluate our method on point-goal navigation in the Habitat Challenge 2021 [1] and show that VOT outperforms previous methods [35] with training on only 5% of the data. Beyond this simple demonstration, we stress that our framework is modality-agnostic and not limited to RGB-D input or discrete action spaces and can be adapted to various modalities, e.g., point clouds, surface normals, gyroscopes, accelerators, compass, etc. To the best of our knowledge, VOT is the first widely applicable modality-invariant Transformer-based VO approach and opens up exciting new applications of deep VO in both simulated and real-world applications. We make our code available at github.com/memmelma/VO-Transformer.

2. Related Work

SLAM- vs Learning-based Navigation: Simultaneous Localization and Mapping (SLAM) approaches decompose the navigation task into the components of mapping, localization, planning, and control [49]. These methods rely on explicit visual feature extraction and, therefore, fail in realistic settings with noisy observations [64], while learning-based methods are more robust to noise, ambiguous observations, and limited sensor suites [27, 32]. However, learning-based methods require an order of magnitude more data, e.g., available through simulation [40]. To deal with the large data requirements, SLAM- and learning-based methods can be combined [5, 8, 9, 11, 48, 61, 63].

Visual Odometry for Realistic Indoor Navigation: While most VO methods estimate an agent’s pose change from more than two frames [52, 53] or optical flow [66], subsequent frames in indoor environments share almost no overlap and contain many occlusions due to the large displacement caused by the discrete action space [64]. Datta et al. [12] propose to estimate the pose change from consecutive frames via a ConvNet architecture and decouple learning the VO from the task-specific navigation policy to allow for retraining modules when dynamics change or the actuation experiences noise. Zhao et al. [64] improve the model’s robustness to observation and actuation noise through geometric invariance losses [54], separate models for moving and turning, pre-process observations, and introduce dropout [44]. Finally, Partsey et al. [35] explore the need for explicit map building in autonomous indoor navigation. They apply train- and test-time augmentations and concatenate an action embedding similar to Zhao et al. [64] to the extracted visual features. A trend is to exploit simulators to gather large datasets (1M [64], 5M [35]). While this is a reasonable progression, it is infeasible to re-train the VO model whenever dynamics or sensor configurations change.

Multi-modal Representation Learning: The availability of multi-modal or pseudo-labeled [4] data [13, 16, 34, 38, 59, 65], e.g., depth, video, and audio, makes it possible to learn feature-rich representations over multiple modalities. Together with Transformer’s [51] ability to process a token sequence of arbitrary length, this leads to general-purpose architectures that can handle various modalities [23] like video, images, and audio [30] or single-view 3D geometry [17]. In particular, Multi-modal Multi-task Geometric Autoencoder (MultiMAE) [4] is a multi-modal pre-training
strategy that performs masked autoencoding [19] with RGB, Depth, and Semantic Segmentation (SemSeg). We show that fine-tuning a pre-trained MultiMAE model can significantly increase VO performance using only 5% of the training data amount of previous methods [35].

3. Proposed Method

3.1. Preliminaries

In the realistic PointGoal Navigation task [2], an agent spawns at a random position in an unseen environment and is given a random goal location $g_t$ relative to its starting position. At each time step $t$ of an episode, the agent perceives its environment through observations $o_t$ and executes an action $a_t$ from a set of discrete actions (move fwd 0.25m, turn left and right by 30°). The stop action indicates the agent’s confidence in having reached the goal. Because the relative goal position $g_t$ is defined at the beginning of each episode, it has to be updated throughout the episode as the actions change the agent’s position and orientation. Following [12, 64], we update $g_t$ through an estimate of the agent’s coordinate transformation. With access to GPS+Compass, computing this transformation is trivial. However, since those sensors are unavailable, we estimate the transformation from the agent’s subsequent observations $o_t, o_{t+1}$ and update the estimated relative goal position $\hat{g}_t$. When taking an action $a_t$, the agent’s coordinate system $C_t$ transforms into $C_{t+1}$. Because the agent can only navigate planarly in the indoor scene, we discard the 3rd dimension for simplicity. We define the estimated transformation as $\hat{H} \in SE(2)$, with $SE(2)$ being the group of rigid transformations in a 2D plane and parameterize it by the estimated rotation angle $\hat{\beta} \in \mathbb{R}$ and estimated translation vector $\hat{\xi} \in \mathbb{R}^2$:

$$\hat{H} = \begin{bmatrix} \hat{R} & \hat{\xi} \\ 0 & 1 \end{bmatrix}, \quad \hat{R} = \begin{bmatrix} \cos(\hat{\beta}) & -\sin(\hat{\beta}) \\ \sin(\hat{\beta}) & \cos(\hat{\beta}) \end{bmatrix} \in SO(2). \quad (1)$$

We then learn a VO model $f_\phi$ with parameters $\phi$ predicting $\hat{\beta}, \hat{\xi}$ from observations $o_t, o_{t+1}$: $\hat{\beta}, \hat{\xi} = f_\phi(o_t, o_{t+1})$. Finally, we transform $\hat{g}_t$ in coordinate system $C_t$ to the new agent coordinate system $C_{t+1}$ by $\hat{g}_{t+1} = \hat{H} \cdot \hat{g}_t$.

3.2. Visual Odometry Transformer

Model Architecture: When facing “optional” modalities, it is not yet clear how systems should react. Options range from constructing an alternative input, e.g., noise [29], to falling back on a model trained without the missing modalities, to training the network with placeholder inputs [31]. Besides these, recent approaches depend on a fixed set of modalities during train- and test-time due to their ConvNet-based backbone. Transformer-based architectures can process a variable number of input tokens and can be explicitly trained to accept fewer modalities during test-time while observing multiple modalities throughout training [4, 51]. Furthermore, the Transformer’s global receptive field could be beneficial for VO, which often gets solved with correspondence or feature matching techniques [41]. We, therefore, propose the Visual Odometry Transformer (VOT), a multi-modal Transformer-based architecture for VO.

Visual Odometry Estimation: To estimate the VO parameters, we pass the encoded Action Token ($\text{[ACT]}$) token to a prediction head. We use a two-layer Multi-layer Perceptron (MLP) with learnable parameters $\psi$ composed into $W_0 \in \mathbb{R}^{d_\psi \times d_h}, b_0 \in \mathbb{R}^{d_h}$, and $W_1 \in \mathbb{R}^{d_h \times 3}, b_1 \in \mathbb{R}^3$. figure 2. The Visual Odometry Transformer architecture for RGB-D input. Image patches are turned into tokens through modality-specific linear projections before a fixed positional embedding is added to them. We pass an action token that embeds the action taken by the agent. An MLP-head then estimates the VO parameters $\hat{\beta}, \hat{\xi}$ from the output token corresponding to the input action token. By randomly dropping either RGB or Depth during training, the Transformer backbone becomes modality-agnostic, allowing it to deal with a subset of these input modalities during test-time without losing performance. When more modalities are available during training, other modality-specific linear projections can be added to process the additional information.
with token dimensions $d = 768$, and hidden dimensions $d_h = d/2$. A Gaussian Error Linear Unit (GELU) [21] acts as the non-linearity between the two layers. The VO model can then be defined as a function $f(x, y, a_t, o_{t+1}, o_t)$ taking as input the action $a_t$ and the observations $o_{t}, o_{t+1}$ corresponding to either RGB, Depth, or RGB-D and predicting the VO parameters $\beta, \xi$. Simplifying the backbone as $b_o(a_t, o_{t+1}, a_t)$ that returns extracted visual features $v_{t+1} \in \mathbb{R}^{1 \times d}$, and governed by parameters $\phi$, the resulting model is:

$$b_o(a_t, o_{t+1}, a_t) = v_{t+1}$$

$$MLP_{\psi}(v) = \text{GELU}(vW_0 + b_0)W_1 + b_1$$

$$f(x, y, a_t, o_{t+1}, o_t) = MLP_{\psi}(b_o(a_t, o_{t+1}, a_t)) = \beta, \xi$$

**Action Prior:** The action $a_t$ taken by the agent to get from $o_t$ to $o_{t+1}$ is a powerful prior on the VO parameters. To provide this information to the model, we embed the action using an embedding layer [36]. This layer acts as a learnable lookup for each action, mapping it to a fixed-size embedding. With the embedding size equal to the token dimensions, we can create an $[ACT]$ and pass the information directly to the model (cf. Figure 2). In contrast to [35, 64], we pass the token directly to the encoder instead of concatenating it to the extracted features. This practice conditions the visual feature extraction on the action and helps ignore irrelevant parts of the image. Note that this approach is not limited to discrete actions but tokens could represent continuous sensor readings like accelerometers, gyroscopes, and compasses, allowing for flexible deployment, e.g., in smartphones or autonomous vehicles [43].

**Explicit Modality-invariance Training:** Explicitly training the model to be invariant to its input modalities is one way of dealing with missing sensory information during test-time. To enforce this property, we drop modalities during training to simulate missing modalities during test-time. Furthermore, this procedure can improve training on less informative modalities by bootstrapping model performance with more informative ones. For example, RGB is more prone to overfitting than Depth because the model can latch onto spurious image statistics, e.g., textures. Training on RGB-only would likely cause the model to latch onto those and converge to local minima, not generalizing well to unseen scenes. By increasing the amount of Depth observations seen during training, the model learns to relate both modalities, acting as regularization. We model this notion as a multinomial distribution over modality combinations (here: RGB, Depth, RGB-D) with equal probability. For each batch, we draw a sample from the distribution to determine on which combination to train.

### 4. Experimental Evaluation

#### 4.1. Setup

**Simulation:** We use the AI Habitat (Habitat) simulator for data collection and model evaluation, following the Habitat PointNav Challenge 2020 [1] specifications. The guidelines define an action space of $fwd$ (move forward 0.25m), left (turn left by 30°), right (turn right by 30°), and stop (indicate the agent reached its goal), and include a sensor suite of RGB-D camera, and GPS+Compass (not used in the realistic PointGoal navigation task). The RGB observations get returned into a $[0, 255]$ range while the Depth map is scaled to $[0, 10]$. Both sensors are subject to noise, i.e., noisy actuations [33] and observations [10]. Furthermore, collision dynamics prevent sliding, a behavior that allows the agent to slide along walls on collision. Cosmetic changes bring the simulation closer to the LoCoBot [18], a low-cost robotic platform with an agent radius of 0.18m and height of 0.88m. An optical sensor resolution of 341 × 192 (width × height) emulates an Azure Kinect camera. An episode is successful if the agent calls stop in a radius two times its own, i.e., 0.36m, around the point goal and does so in $T = 500$ total number of time steps. By specification, the 3D scenes loaded into Habitat are from the Gibson [57] dataset, more precisely Gibson-4+ [40], a subset of 72 scenes with the highest quality. The validation set contains 14 scenes, which are not part of the training set.

**Dataset:** For training VOT, we collect a training- and a validation dataset. Each set consists of samples containing the ground truth translation $\xi$ and rotation parameters $\beta$ retrieved from a perfect GPS+Compass sensor, observations $o_t, o_{t+1}$, and taken action $a_t$. We keep samples where the agent collides with its environment as the transformations strongly differ from standard behavior [64]. The collection procedure follows Zhao et al. [64] and is performed as: 1) initialize the Habitat simulator and load a scene from the dataset, 2) place the agent at a random location within the environment with a random orientation, 3) sample a navigable PointGoal the agent should navigate to, 4) compute the shortest path and let the agent follow it, and 5) randomly sample data points along the trajectory. We collect 250 k observation- transformation pairs from the training and 25 k from the validation scenes of Gibson-4+, which is significantly less than comparable methods (1 M [64], 5 M [35]). Furthermore, we apply data augmentation during training to the left and right actions by horizontally flipping the observations and computing the inverse transformation.

**Loss Function:** Our loss function is the $L2$-norm between the ground truth VO parameters and their estimated counterparts. We further add the geometric invariance losses $L_{inv}$ proposed by Zhao et al. [64] and use the Adam [26] optimizer ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1e^{-8}$) to minimize the resulting loss function $L = ||\xi - \xi||^2 + ||\beta - \beta||^2 + L_{inv}$. 
the trained VO models as it has shown minimal navigation performance gains in \[64\] and was abandoned in \[35\].

We train all models, including baselines, for 100 epochs with 10 warm-up epochs that increase the learning rate linearly from 0.0 to \(2e^{-4}\), and evaluate the checkpoints with the lowest validation error. We further find gradient norm clipping \[62\] (max gradient norm of 1.0) to stabilize the training of VOT but to hurt the performance of the ConvNet baselines. The training was done with a batch size of 128 on an NVIDIA V100-8GB GPU with automatic mixed-precision enabled in PyTorch \[36\] to reduce memory footprint and speed up training. Our backbone is a ViT-B \[14\] with a patch size of \(16 \times 16\) and 12 encoder blocks with 12 Multi-head Attention (MHA) heads each, and token dimensions 768. To encode the input into tokens, we use a 2D sine-cosine positional embedding and separate linear projection layers for each modality. Note that if additional modalities are available, our model can be extended by adding additional linear input projections or fine-tuning existing ones \[4\]. Finally, we pass all available tokens to the model and resize each observation to \(160 \times 80 \times c\) (width \(\times\) height \(\times\) channels \(c\)) and concatenate modalities along their height to \(160 \times 160 \times c\) to reduce computation. We keep a running mean and variance to normalize RGB and Depth to zero mean and unit variance.

**Evaluation Metrics:** Anderson et al. \[2\] propose the Success weighted by (normalized inverse) Path Length (SPL) to evaluate agents in a PointGoal or ObjectGoal navigation setting. A crucial component of this metric is the success of an episode (success \(S = 1\), failure \(S = 0\)). With \(l\) the shortest path distance from the starting position and \(p\) the length of the path taken by the agent, the SPL over \(N\) episodes is defined as \[\text{SPL} = \frac{1}{N} \sum_{i=0}^{N-1} \frac{S(i)}{\max(p(i),1(i))},\]
While SPL depends on the success of an episode, [12] propose the Soft Success Path Length (SSPL) that provides a more holistic view of the agent’s navigation performance. The authors replace the binary success $S$ of an episode with a soft value consisting of the ratio between the (geodesic) distances to target upon start $d_{\text{init}}$ and termination of an episode $d_f$. The resulting metric is then $SSPL = \frac{1}{N} \sum_{i=0}^{N-1} \left(1 - \frac{d_{\text{init}}^i}{d_f^i}\right) \frac{d_f^i}{\max(p^i, r^i)}$

The closer the agent gets to the goal, the higher the SSPL, even if the episode is unsuccessful. This softening allows distinguishing agents that fail to complete a single or multiple episodes but move significantly close to the goal from ones that move away from it. Without access to GPS+Compass, SSPL becomes significantly more important as an agent might call stop prematurely due to inaccurate localization. We report the SPL, SSPL, success $S$, and (geodesic) distance to goal on termination $d_f$ on the validation scenes of Gibson-4+ with decimals truncated.

**Navigation Policy:** Similar to prior work [12, 35, 64], we replace the GPS+Compass with our VO model to estimate the relative goal position, which serves as the input to a pre-trained navigation policy. We use the same pre-trained policy as Zhao et al. [64] for our experiments, which was trained using a goal position updated by ground truth localization. The policy architecture consists of a Long Short-Term Memory (LSTM) [22] with two recurrent layers that process 1) a 512-dimensional encoding of the agent’s observations $o_t$ (here: Depth), 2) a 32-dimensional embedding of the previous action, and 3) a 32-dimensional embedding of the updated relative goal position. The observation encoding gets obtained by passing the observations $o_t$ through a ResNet-18 [20] backbone, flattening the resulting feature map to dimensionality 2052, and projecting it to dimensionality 512 with a fully-connected layer. Finally, the output of the LSTM is fed through another fully-connected layer to produce a distribution over the action space and a value function estimate. The policy was trained using DDPO [55], a distributed version of Proximal Policy Optimization (PPO) [42].

### 4.2. Dealing With Optional Modalities

We evaluate the models’ robustness to missing modalities by randomly dropping access to one of the training modalities. This setup probes VOT for dependencies on the input modalities, which directly influence the downstream performance under limited access. In case of sensor malfunctioning, e.g., only a single modality might be available, a ConvNet’s failure is predetermined as it requires a fixed-size input. If not given, the system converges to a Blind behavior, exemplified in Table 1. Limiting access to modalities reveals VOT’s dependency on Depth. Dropping RGB barely decreases performance, while dropping Depth causes the localization to fail more drastically. Comparing the true agent localization and its “imaginary”, i.e., VO estimate, it becomes clear why. Figure 3 shows how the errors accumulate, causing the true location to drift away from the estimate. While the effect is less drastic when dropping RGB, the agent still fails to reach the goal.

Training VOT with the proposed invariance training (w/ inv.), i.e., sampling RGB for 20%, Depth for 30%, and RGB-D for 50% of the training batches, eliminates this shortcoming. Removing RGB now only decreases the success rate by 1.6%, while removing Depth also leads to a stronger performance. This observation suggests that RGB is less informative for the VO task than Depth. Especially when navigating narrow passages, RGB might consist of uniform observations, e.g., textureless surfaces like walls, making it hard to infer the displacement, unlike Depth which would still provide sufficient geometric information (cf. Figure 3). However, this information asymmetry only leads to a decline in the metrics that are sensitive to subtle inconsistencies in the localization, i.e., $S$, and SPL. Inspecting the SSPL, the drop of $-3.5$ is less drastic. Explicit modality-invariance training keeps VOT-B (RGB-D)
and increases to set (ble 216). When training the baseline on our smaller data (improves performance by S baseline [64]. Using only 25% of the training data, VOT for both approaches and the published VO models of the agent. We use the same publicly available navigation policy GPS+Compass as of downstream navigation performance, i.e. 4.3. Quantitative Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Observations</th>
<th>Pre-train</th>
<th>[ACT]</th>
<th>S ↑</th>
<th>SPL ↑</th>
<th>SSPL ↑</th>
<th>d_y ↓</th>
<th>L_{train} ↓</th>
<th>L_vat ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[64] (separate)</td>
<td>RGB-D</td>
<td></td>
<td>22.4</td>
<td>13.8</td>
<td>31.5</td>
<td>305.3</td>
<td>0.125</td>
<td>0.186</td>
</tr>
<tr>
<td>2</td>
<td>[64] (unified)</td>
<td>RGB-D</td>
<td>✔</td>
<td>64.5</td>
<td>48.9</td>
<td>65.4</td>
<td>85.3</td>
<td>0.264</td>
<td>0.420</td>
</tr>
<tr>
<td>3</td>
<td>Blind</td>
<td>–</td>
<td>✔</td>
<td>0.0</td>
<td>0.0</td>
<td>5.4</td>
<td>398.7</td>
<td>48.770</td>
<td>47.258</td>
</tr>
<tr>
<td>4</td>
<td>VOT-B</td>
<td>RGB</td>
<td>✔</td>
<td>27.1</td>
<td>21.2</td>
<td>57.7</td>
<td>177.0</td>
<td>0.735</td>
<td>1.075</td>
</tr>
<tr>
<td>5</td>
<td>VOT-B</td>
<td>Depth</td>
<td>✔</td>
<td>43.2</td>
<td>32.0</td>
<td>59.3</td>
<td>122.5</td>
<td><strong>0.441</strong></td>
<td><strong>0.644</strong></td>
</tr>
<tr>
<td>6</td>
<td>VOT-B</td>
<td>RGB-D</td>
<td>✔</td>
<td><strong>47.3</strong></td>
<td><strong>36.3</strong></td>
<td><strong>61.2</strong></td>
<td>119.7</td>
<td>1.256</td>
<td>1.698</td>
</tr>
<tr>
<td>7</td>
<td>Blind</td>
<td>–</td>
<td>✔</td>
<td>13.3</td>
<td>10.0</td>
<td>46.3</td>
<td>251.8</td>
<td>1.637</td>
<td>1.641</td>
</tr>
<tr>
<td>8</td>
<td>VOT-B</td>
<td>RGB</td>
<td>✔</td>
<td>42.0</td>
<td>32.3</td>
<td>62.7</td>
<td>107.0</td>
<td>0.043</td>
<td>0.571</td>
</tr>
<tr>
<td>9</td>
<td>VOT-B</td>
<td>Depth</td>
<td>✔</td>
<td><strong>76.1</strong></td>
<td><strong>58.8</strong></td>
<td><strong>69.2</strong></td>
<td><strong>60.7</strong></td>
<td><strong>0.017</strong></td>
<td><strong>0.113</strong></td>
</tr>
<tr>
<td>10</td>
<td>VOT-B</td>
<td>RGB-D</td>
<td>✔</td>
<td>72.1</td>
<td>55.6</td>
<td>68.5</td>
<td>64.4</td>
<td>0.019</td>
<td>0.129</td>
</tr>
<tr>
<td>11</td>
<td>VOT-B</td>
<td>RGB</td>
<td>✔</td>
<td>54.5</td>
<td>41.3</td>
<td>65.2</td>
<td>69.9</td>
<td>0.056</td>
<td>0.347</td>
</tr>
<tr>
<td>12</td>
<td>VOT-B</td>
<td>Depth</td>
<td>✔</td>
<td>83.2</td>
<td>63.4</td>
<td>69.1</td>
<td>49.9</td>
<td>0.079</td>
<td>0.205</td>
</tr>
<tr>
<td>13</td>
<td>VOT-B</td>
<td>RGB-D</td>
<td>✔</td>
<td><strong>85.7</strong></td>
<td><strong>65.7</strong></td>
<td><strong>69.7</strong></td>
<td>56.1</td>
<td><strong>0.021</strong></td>
<td><strong>0.060</strong></td>
</tr>
<tr>
<td>14</td>
<td>VOT-B</td>
<td>RGB</td>
<td>✔</td>
<td>59.3</td>
<td>45.4</td>
<td>66.7</td>
<td>66.2</td>
<td>0.003</td>
<td>0.280</td>
</tr>
<tr>
<td>15</td>
<td>VOT-B</td>
<td>Depth</td>
<td>✔</td>
<td><strong>93.3</strong></td>
<td><strong>71.7</strong></td>
<td><strong>72.0</strong></td>
<td><strong>38.0</strong></td>
<td><strong>0.004</strong></td>
<td><strong>0.044</strong></td>
</tr>
<tr>
<td>16</td>
<td>VOT-B</td>
<td>RGB-D</td>
<td>✔</td>
<td>88.2</td>
<td>67.9</td>
<td>71.3</td>
<td>42.1</td>
<td>0.004</td>
<td>0.051</td>
</tr>
<tr>
<td>17</td>
<td>VOT-B w/ inv.</td>
<td>RGB-D</td>
<td>✔</td>
<td>✔</td>
<td><strong>92.6</strong></td>
<td><strong>70.6</strong></td>
<td><strong>71.3</strong></td>
<td><strong>40.7</strong></td>
<td><strong>0.008</strong></td>
</tr>
</tbody>
</table>

**oracle** GPS+Compass – – 97.8 74.8 73.1 29.9 – –

Table 2. Ablation study of architecture design and input modalities. We further investigate pre-training with MultiMAE [4] in models 11-14. Losses L, Success S, SPL, SSPL, and d_y reported as e−2. Bold indicates best results.

Table 3. Habitat Challenge 2021. Results for the Point Nav Test-Standard Phase (test-val split) retrieved on 05-Nov-2022.

from exploiting this asymmetry and matches the performance of VOT-B (RGB) when Depth is dropped during test-time Tab. 1.

4.3. Quantitative Results

We compare our approach to Zhao et al. [64] in terms of downstream navigation performance, i.e., the VO model as GPS+Compass replacement for a learned navigation agent. We use the same publicly available navigation policy for both approaches and the published VO models of the baseline [64]. Using only 25% of the training data, VOT improves performance by S + 12.3, SPL+9.7, SSPL+2.0 (cf. Table 2 15) and S + 7.2, SPL+5.7, SSPL+1.3 (cf. Table 2 16). When training the baseline on our smaller data set (cf. Table 2 2, unified, ResNet-50), this improvement increases to S + 29.8, SPL+22.8, SSPL+6.6 (cf. Table 2 15) and S + 23.7, SPL+19.0, SSPL+5.9 (cf. Table 2 16).

To capture the raw VO performance detached from the indoor navigation task, we inspect the absolute prediction error in Figure 4. We differentiate between translation ξ in x- and y- direction (ξ_x, ξ_y), and taken action. VOT is accurate up to 0.36 cm (fwd), 1.04 cm (right), 1.05 cm (left) in x-direction and 0.20 cm (fwd), 0.41 cm (right), 0.38 cm (left) in z-direction. Note how the baseline struggles to capture ξ, corresponding to the forward-moving direction z when taking the fwd action.

Given the results in Table 2, we advise using VOT trained on Depth-only when access is assumed, as the difference to using GPS+Compass is a mere S − 4.5, SPL−3.1, SSPL−1.1. When "optional" modalities are needed, e.g., they are expected to change during test-time, invariance training should be used. Trained on RGB-D, this setup also reaches GPS+Compass like performance with differences of only S − 5.2, SPL−4.2, SSPL−1.8.

4.4. Ablation Study

We identify the impact of different input modalities and model design choices in our ablation study (cf. Table 2). Without observations, the Blind VO model cannot update the goal position. This means the agent can only act without goal-related feedback, resulting in a 0% success rate.

Extending the model with our proposed [ACT] token allows it to surpass the Blind performance. Able to up-
We conclude that the RGB modality is a more informative modality than Depth. While the RGB and Depth observations correlate with the \textit{ACT} token, they also contain information about the noisy actuation. Vice versa, \textit{ACT} disambiguates corner cases where the visual observations do not provide explicit information about the underlying action. For instance, a \textit{fwd} action colliding with a wall might be hard to distinguish from a noisy \textit{left} turning less than 30° \cite{64}.

Our results show that MultiMAE pre-training provides useful multi-modal features for VO that fine-tuned outperform the ConvNet baselines. In addition, these features are complementary to the \textit{ACT} prior, together achieving state-of-the-art results. We conclude that the \textit{ACT} prior biases the model towards the mean of the corresponding transformation, while the pre-training supports the learning of the additive actuation noise.

Training separate models for each modality reveals that Depth is a more informative modality than RGB for VO. We assume this to be a direct result of its geometric properties, \textit{i.e.}, the 3D structure of the scene. We find that training VOT on noisy RGB even hurs the localization. The model overfits the visual appearance of the scenes and is unable to generalize to unseen ones. In turn, Depth does not suffer from this issue as it only contains geometric information.

**4.5. Action-conditioned Feature Extraction**

We show what image regions the model attends to by visualizing the attention maps of the last MHA-layer (\textit{cf.} Table 2 \textit{id}) corresponding to the \textit{ACT} token in Figure 5. To reduce the dimensionality of the visualization, we fuse the heads’ weights via the \textit{max} operator and align the attention maps with the input images. We normalize the maps to show the full range of the color scheme.

We find that passing different actions to VOT primes it to attend to meaningful regions in the image. When passed turning actions \textit{left} or \textit{right}, VOT focuses on regions present at both time steps. This makes intuitive sense, as a turning action of 30° strongly displaces visual features or even pushes them out of the agent’s field of view. A similar behavior emerges for a \textit{fwd} action which leads to more attention on the center regions, \textit{e.g.}, the walls and the end of a hallway (\textit{cf.} Figure 5b). These results are particularly interesting as the model has no prior knowledge about the VO task but learns something about its underlying structure.

**4.6. Habitat Challenge 2021 PointNav**

We compare our approach (\textit{cf.} Table 2 \textit{id}) to several baselines submitted to the Habitat Challenge 2021 benchmark in Table 3. Using the same navigation policy as Partsey \textit{et al.} \cite{35}, VOT achieves the highest SSPL and on par with SPL and $d_g$ training on only 5% of the data. These results clearly show that reusability doesn’t come with a price of lower performance and that scaling data requirements doesn’t seem to be the answer to solving deep VO.

**4.7. Limitations**

In our work, we separate the VO model from the navigation policy and only focus on the modality-invariance of the former, neglecting that the navigation policy expects Depth as input \cite{12,35,64}. Designing policies to be modality-invariant is subject to future research. Assuming an accurate sensor failure detection when dropping modalities, additionally, is an idealized setup. Furthermore, our experiments in the Habitat’s simulator limit the available modalities to RGB–D. Even though SemSeg has shown to be beneficial for some VO applications \cite{37,50}, there is no specific sensor for it. However, SemSeg could be estimated from RGB. While our experiments focus on discrete actions and RGB–D, our architecture could be adapted to continuous actions and other sensor types. However, training might become more difficult due to a lack of pre-trained weights.

**5. Conclusions**

We present Visual Odometry Transformers for learned Visual Odometry. Through multi-modal pre-training and action-conditioned feature extraction, our method is sample efficient and outperforms current methods trained on an order of magnitude more data. With its modality-agnostic design and modality-invariance training, a single model can deal with different sensor suites during training and can trade-off subsets of those during test-time.
References


[12] Samyak Datta, Oleksandr Maksymets, Judy Hoffman, Stefan Lee, Dhruv Batra, and Devi Parikh. Integrating egocentric localization for more realistic point-goal navigation agents. In Conference on Robot Learning, 2021. 1, 2, 3, 5, 6, 8


[19] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In Computer Vision and Pattern Recognition, 2022. 3


Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In European Conference on Computer Vision, 2012. 2

Ruslan Partsey, Erik Wijmans, Naoki Yokoyama, Oles Dobosyvych, Dhruv Batra, and Oleksandr Maksymets. Is mapping necessary for realistic pointgoal navigation? In Conference on Computer Vision and Pattern Recognition, 2022. 2, 3, 4, 5, 6, 7, 8


Xiangwei Wang, Daniel Maturana, Shichao Yang, Wenshan Wang, Qijun Chen, and Sebastian Scherer. Improving


