PACT: Perception-Action Causal Transformer for Autoregressive Robotics Pretraining

Rogerio Bonatti, Sai Vemprala, Shuang Ma, Felipe Frujeri, Shuhang Chen, Ashish Kapoor
Microsoft

Abstract

Robotics has long been a field riddled with complex systems architectures whose modules and connections, whether traditional or learning-based, require significant human expertise and prior knowledge. Inspired by large pre-trained language models, this work introduces a paradigm for pre-training a general purpose representation that can serve as a starting point for multiple tasks on a given robot. We present the Perception-Action Causal Transformer (PACT), a generative transformer-based architecture that aims to build representations directly from robot data in a self-supervised fashion. Through autoregressive prediction of states and actions over time, our model implicitly encodes dynamics and behaviors for a particular robot. Our experimental evaluation focuses on the domain of mobile agents, where we show that this robot-specific representation can function as a single starting point to achieve distinct tasks such as safe navigation, localization and mapping. We evaluate two form factors: a wheeled robot that uses a LiDAR sensor as perception input (MuSHR), and a simulated agent that uses first-person RGB images (Habitat). We show that finetuning small task-specific networks on top of the larger pretrained model results in significantly better performance compared to training a single model from scratch for all tasks simultaneously, and comparable performance to training a separate large model for each task independently. By sharing a common good-quality representation across tasks we can lower overall model capacity and speed up the real-time deployment of such systems.

1 Introduction

Recent advances in machine learning architectures have started a paradigm shift from task-specific models towards large general purpose models. Such a shift has most commonly been observed in the domain of natural language, as evidenced by large language models such as BERT [1], GPT-3 [2] and Megatron-Turing [3], as well as in computer vision [4, 5, 6].

Some of these large models already combine multiple data modalities such as text, images, video, audio, as well as the relationship between datapoints over time [7, 8]. The use of foundational models is appealing because they are trained on broad datasets over a wide variety of downstream tasks, and therefore provide general skills which can be used directly or with minimal fine-tuning to new applications. More recently, large pretrained models have been applied to multi-task learning spanning multiple domains [9].

While machine learning models are finding widespread use in robotics, most of them have been task or hardware-specific, which necessitate redesign and retraining if there are minor changes in robot dynamics, environment, or operational objectives [10]. A contrast can be drawn between such approaches in robotics and research in the domain of natural language. For instance, hand-crafted language models that encoded grammatical rules and syntax have been replaced by large models that can learn directly from data. Such large models often encode general purpose information about language, grammar, and can be finetuned for specific tasks with relative ease. Similarly, we envisage

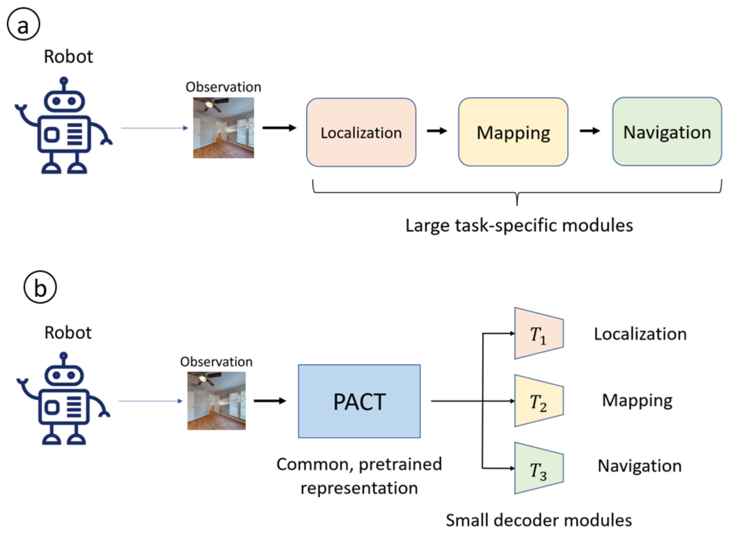


Figure 1: Comparison between a) traditional robotics architecture with hand-crafted modules, and b) foundation model architecture with common pre-trained representations and lightweight downstream modules.

a general architecture for robotics that requires less priors and domain expertise, and can serve as a starting point for several tasks.

The representation learning methods presented in this paper are built to be agnostic to the specific robotics domain as long as we use states and actions to represent our system. However, this paper focuses on domains related to mobile agents, where the typical robot autonomy pipeline involves objectives such as localization, mapping, and planning.

Recent works have just started to explore the use of pretrained models towards robotic decision-making. The main challenges in robotics that differ from traditional language, vision and vision-language models are:

Multi-modal data: unlike large language models that are limited to a single domain of data, robotics models often need to process disparate input modalities (images, point clouds, velocities, arbitrary features), and output low and high-level decisions;

Sequential decision-making: actions have consequences, and a robotics model must be able to not only summarize the current information, but also reason about the multitude of possible futures scenarios and the causal relationships between the problem's variables;

Expensive and scarce data: unlike language and vision data, real-world robotics data is prohibitively expensive to collect in large quantities, and therefore requires the use of simulators which often involves a sim-to-real gap. In addition, data collected for a particular robot form does not necessarily generalize across new platforms or tasks, and sufficient data diversity is problematic.

In this paper, we present a pre-training paradigm for robotics, with special experimental focus on the domain of mobile agents. We identify that at their core, most robotic agents process a perception-action loop between their states/observations, and associated actions - and that an understanding of state-action transitions is beneficial for several tasks in the pipeline of robot autonomy. We hence present the Perception-Action Causal Transformer (PACT), a transformer-based generative model that is trained on sequences of states and actions coming from robot trajectories. By learning to autoregressively predict such sequences, PACT implicitly encodes general purpose information such as the notion of which next state would be reached from a current state and action (robot dynamics), as well as the notion of which action to take given a certain state. We train PACT on trajectories obtained from two different navigation domains (MuSHR wheeled car and virtual Habitat agent), and we show that representations learnt by PACT can act as a base to efficiently solve several robotics tasks such as localization, mapping and navigation.

Our main contributions are listed below:

- We propose a pretraining architecture for robotics tasks based on a Perception-Action Causal Transformer (PACT), which uses an autoregressive objective to encode state-action transitions;
- Within the domain of mobile agents, we show that this pretrained model can provide a single reasonable starting point for the tasks of safe navigation, localization, and mapping. We experimentally verify that our frozen common representation can achieve performance levels similar to training separate networks of equal capacity individually for each downstream task, and significantly higher performance than training a single network for all tasks from scratch.
- We provide experiments in navigation tasks that span different data modalities (LiDAR and RGB) and robot dynamics models (wheeled, omni-directional), and real-world robot validation.

2 Related Work

Learning robotics representations: Approaches for task learning in robotics can be categorized into three main segments. The first, and the most common, is the task-specific training approach where modules are designed specific to each task. Such methods have been proposed for several of the most common robotic tasks such as visual/LiDAR based localization, mapping, path planning and control [11, 12]. Another category involves multi-task learning approaches, where models are trained jointly to be able to solve several tasks [13, 14]. Finally, there exists a class of techniques that perform task-agnostic pre-training, whose representations can later be finetuned for a task of choice [15, 16, 17].

Multi-modal robotics representations: Representation learning is a rapidly growing field. The existing visual-language representation approaches primarily rely on BERT-style [1] training objectives to model the cross-modal alignments. Common downstream tasks consist of visual question-answering, grounding, retrieval and captioning etc. [5, 6, 18, 19]. Learning representations for robotics tasks poses additional challenges, as perception data is conditioned on the motion policy and model dynamics [20]. Visual-language navigation of embodied agents is well-established field with clear benchmarks and simulators [21, 22], and multiple works explore the alignment of vision and language data by combining pre-trained models with fine-tuning [23, 24, 25] To better model the visual-language alignment, [26] also proposed a co-grounding attention mechanism. In the manipulation domain we also find the work of [27], which uses CLIP [28] embeddings to combine semantic and spatial information.

Transformers in robotics: Transformers were originally introduced in the language processing domain [29], but quickly proved to be useful in modeling long-range data dependencies other domains. Within robotics we see the first transformers architectures being used for trajectory forecasting [30], motion planning [31, 32], and reinforcement learning [33, 34]. Our main difference between the related works in [33, 34] (Decision Transformer and Trajectory Transformer) is that they are focused on training a model for a single task, while we propose learning representations amenable to multiple downstream tasks for a robot.

3 Pretraining Approach

We aim to create a general purpose pre-training architecture that can be trained to produce an effective state-action representation. Our model is called the Perception-Action Causal Transformer, (PACT) which works as a causal transformer that ingests perception and action data. Note that, as is typical in robotics, the states that are being learned by our model are not the ground truth states, but rather sensor observations. Using this data, PACT is expected to learn an effective joint representation of states and actions, resulting in an implicit understanding of dynamics in an end-to-end manner.

3.1 Tokenization

In a large variety of applications, raw observations can be of distinct modalities, (e.g. RGB images, LiDAR scans, depth maps). Similarly, robot actions can be of several types as well such as steering angles, motor commands, or a discrete choice from a predefined library of actions. In order to convert

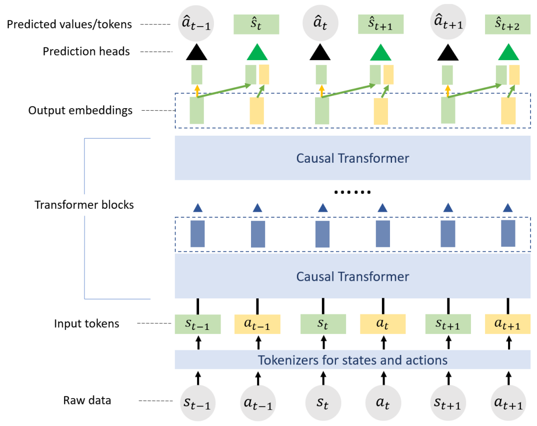

Predicted values/tokens
Prediction heads
Output embeddings
Causal Transformer
Transformer blocks
Causal Transformer
Input tokens
Tokenizers for states and actions
Raw data
$\hat{a}_{t-1}$ $\hat{s}_t$ $\hat{a}_t$ $\hat{s}_{t+1}$ $\hat{a}_{t+1}$ $\hat{s}_{t+2}$
s_{t-1} a_{t-1} s_t a_t s_{t+1} a_{t+1}

Figure 2: Perception-Action Causal Transformer (PACT) architecture. $\hat{a}$ and $\hat{s}$ are autoregressively predicted actions and states. The tokenizer does not share information across data, and applies operations individually on raw data inputs. The black and green arrows represent predictions heads for actions and future state tokens respectively.

such a wide variety of data into a format that is easily accessible by the Transformer, a tokenization procedure is required. We create state and action tokenizers such that our PACT model can ingest different observation modalities, and can also handle discrete and continuous action spaces. We descrive specific architectures next:
RGB images: We use a ResNet-18 backbone [35] trained from scratch to compute features for RGB images, which are then converted into a token of length 128.
Raw LiDAR scans: We use a 2D LiDAR that returns a sequence of range values. We convert these values into XY locations relative to the vehicle, and then use PointNet [36] to compute a feature vector. We remove PointNet's transform blocks so that the resulting token is not agnostic to the point cloud's orientation relative to the vehicle.
BEV LiDAR scans: An alternate approach to tokenize LiDAR scans was to convert the returns into a bird's eye view (BEV) image of size 200×200 corresponding to 15×15 meters around the robot and process it with a ResNet-18 backbone to a token of size 128. While this technique presented equivalent performance to PointNet in simulation, we found that sim-to-real transfer was more robust using BEV projection.
Discrete actions: For Habitat we have a 4D discrete action space with 'left', 'right', 'forward', and 'stop' actions. We use a simple linear embedding to map the 4D actions to a token.
Continuous actions: We use a 2-layer MLP to map continuous actions into tokens.

3.2 Model design

The Transformer architecture [29] has found significant use in sequence modeling problems. Most Transformer models consist of stacked self-attention layers, and operate on a sequence of embeddings corresponding to input tokens, and transform them into another sequence of embeddings of equal length. Let us assume s_t and a_t to represent the state and action at time t, and consider a trajectory τ as a sequence of pairs of the form $\tau = \{(s_0, a_0), (s_1, a_1), ..., (s_T, a_T)\}$. In our work we attempt to autoregressively model this sequential data of states and actions using the causal transformer architecture, similar to the GPT architecture [2]. The transformer's causal self-attention mask ensures that any particular output token within the sequence is a result of operations over only tokens in past time steps.

As shown in Fig. 2, the main blocks in PACT are the state and action tokenizers, causal transformer blocks and prediction heads. Our model computes predictions within a limited time horizon of length K which contains a total of $2K$ tokens of alternating states and actions. Each input is first tokenized into the embedding dimension, and added with a learned positional embedding for each time step. Similar to Decision Transformer [33], we use a global time embedding to indicate the token's position within the full sequence ($1 \rightarrow 2K$), and a local time embedding which indicates the current time step ($1 \rightarrow K$). This combined sequence of inputs is then fed through multiple layers of causal Transformer blocks to result in a set of output embeddings.

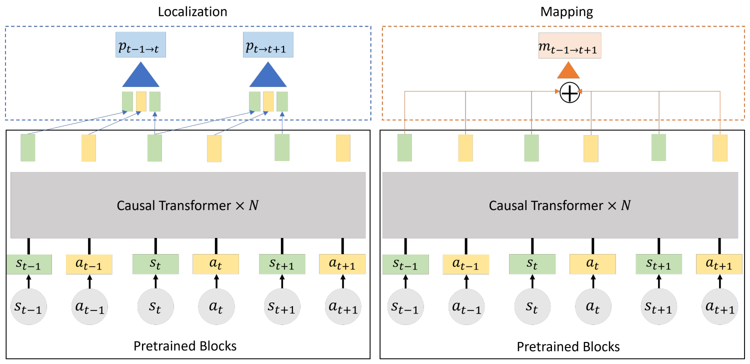


Figure 3: PACT representations can be used for downstream tasks such as localization (left), which needs frame-to-frame computations, and mapping (right) which accumulates data over a window of time.

3.3 Pre-training Objectives

Let $q(\cdot)$ represent the tokenizer operation, and $X(\cdot)$ represent the transformer embedding operation for a given token. To pretrain PACT, we use self-supervised learning with separate action and state prediction heads. The action prediction head h_a is expected to predict the appropriate next action given the current state embedding $h_a(X(q(s_t))) \to a_t$, acting as a policy. The state prediction head h_s predicts the next state token given the previous state and action embeddings $h_s(\ X(q(s_t)), X(q(a_t))\ ) \to q(s_{t+1})$, acting as a dynamics model.

Details on the hyperparameters used during pretraining can be found in Appendix A.

4 Downstream Tasks and Experimental Setup

Our goal is to use the pre-trained PACT representation as a basis for different robotics downstream tasks. We focus our evaluation on the domain of mobile agents, and show that our robot-specific representations can function as a single starting point towards safe navigation, localization and mapping.

4.1 Finetuning pipeline

Localization: Here define robot localization as the task of predicting the robot's pose over time relative to the initial pose of the trajectory. Our network runs a deep odometry algorithm, calculating the pose difference between consecutive time steps: $p_{t-1\to t}$. We us a lightweight 3-layer MLP h_{loc} ($[64, 32, 3]$) as the localization task decoder (Fig. 3, left), whose input is a tuple containing the previous state and action embeddings plus and current state embedding: $p_{t-1\to t} = h_{\text{loc}}(X(q(s_{t-1})), X(q(a_{t-1})), X(q(s_t)))$. We train the head using MSE loss with respect to the ground-truth pose difference $\{\Delta X, \Delta Y, \Delta\theta\}$. The pose differences are integrated over time to estimate the robot's global pose. Note that we can use the transformer in a rolling-window fashion and accumulate poses over the entire trajectory duration, much longer than the original transformer length K.

Mapping: We define mapping as the task of predicting a local 2D occupancy image I_o around the robot's current location (as a top-down view), given the local history (length K) of the state and action embeddings. We use a 2-layer deconvolutional decoder h_{map} to output an image of size 64×64 that represents the local occupancy of an area of 15×15 meters surrounding the robot: $I_o = h_{\text{map}}(X(q(\tau_{j:j+K}))$. We train the mapping decoder (Fig. 3, right) using an MSE loss over the binary ground-truth pixel values.

4.2 Experimental setup

MuSHR car: We apply PACT to a wheeled robot platform, MuSHR [37], which is equipped with a 2D LiDAR sensor with angular resolution of 0.5 degree, and takes a steering angle in the range of $[-21.75, 21.75]$ deg. as control input. The data for pre-training and downstream tasks is obtained by running a MuSHR simulator on a real-world map collected from an office environment

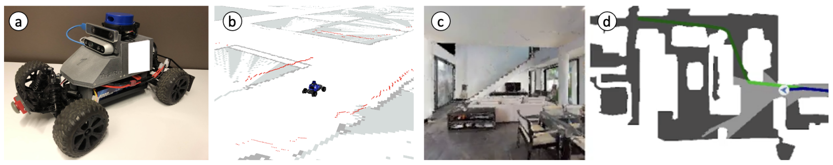
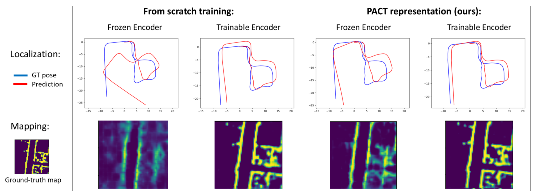
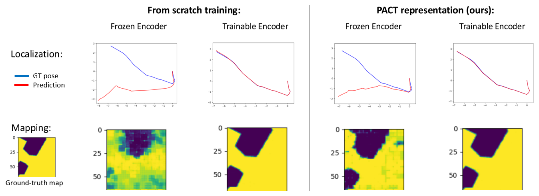


Figure 4: Deployment environments. a) Real-life MuSHR vehicle, b) MuSHR simulator with dynamics model and LiDAR, c) Habitat FPV image, and d) Top-down view of Habitat environment map.



(a) Mushr



(b) Habitat

Figure 5: Visualization of mapping and localization for Mushr (a) and Habitat (b). We compare PACT pre-training against the same model backbone that trained from scratch. Under each setting, we compare frozen and trainable feature encoders.

of approximately $75m \times 30m$. The simulator contains an expert MPC planner which outputs safe trajectories given the occupancy map. Using this simulator, we record LiDAR scans and corresponding actions for a total of 5.5K trajectories, or 3M perception-action tokens.

For all simulated experiments the observations, or LiDAR scans, are first converted into a feature vector of size 1024 using learned PointNet [36], and then reduced into a token of dimension 128. For real-world experiments we verified experimentally that processing a top-down LiDAR projection image resulted in better policy transfer, as explained in Section 3.1. Each action is converted from a continuous scalar into a token in a dimension of 128 using a two-layer MLP.

Habitat: We also test our method in the Habitat simulator [38]. In this case, we train our models on data obtained from expert trajectories on the PointNav task in Habitat. We record the first-person camera images and action information across 10, collecting a total of 10K episodes, or 840K tokens. Actions belong to a discrete set ('left', 'right', 'forward', and 'stop'). We train a ResNet18 [35] encoder to tokenize the images of size 224×224, and use single linear embedding to tokenize the discrete actions from a one-hot vector. All embeddings are mapped to a size of 128.

5 Results

5.1 Main Experimental Hypotheses

We investigate a set of hypotheses regarding pre-training and finetuning representations for navigation scenarios. The main metrics used here are MSE (mean squared error), MAE (mean absolute error), and ATE (absolute trajectory error). We detail these hypotheses and results below:

H1 - PACT representations can achieve similar or better performance than models trained from scratch: As discussed in [39], there are no theoretical guarantees that finetuning a model starting from a pre-trained representation will necessarily achieve better results than training from scratch, given enough data and time. In this hypothesis we want to validate that the PACT representation starting point can achieve at least a similar, and ideally better, performance in downstream tasks when compared to training a network from scratch for localization and mapping.

Tables 1 and 2 show the comparison results. For the columns under 'PACT' we first pre-train a base model for each environment, and as a second step we finetune this representations for the downstream tasks. 'F' indicates a frozen representation, where only the small task-specific head is trained, and 'T' indicates a fully trainable network. The 'Scratch' column shows networks trained starting from random weights. Figure 5 displays visualizations of the output from the four different models for MuSHR and Habitat.

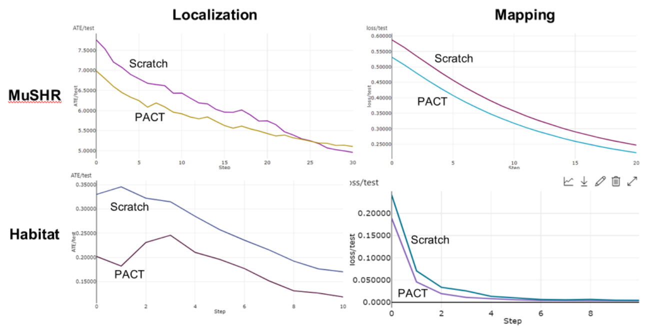


Figure 6: Speed of convergence over training epochs for localization and mapping metrics. We see that pre-training offers a better starting point, even if training from scratch eventually surpasses it.

The main conclusions from tables 1 and 2 is that in general our hypothesis holds true, and networks derived from PACT pre-training achieve similar and in several cases better results than randomly initialized weights. We also find, not surprisingly, that trainable representations (T) generally achieve better results than freezing the transformer encoder layers (F). The gap in performance between F and T is especially large for the Habitat environment, likely due to the high dimensionality of the image perception modality.

In Figure 6 we also investigate the training time required to achieve good performance for pre-trained versus random weights. We can see that PACT offers a better initialization, even if training from scratch eventually surpasses its performance in later epochs.

Table 1: Localization ATE averaged over 30 trajectories (meters, lower error is better)

(a) MuSHR

# FT eps.	Scratch		PACT	
	F	T	F	T
100	7.13	6.30	5.51	**4.60**
1000	5.59	4.59	**4.21**	4.90
5500	5.39	**3.08**	4.09	3.62

(b) Habitat

# FT eps.	Scratch		PACT	
	F	T	F	T
800	1.94	0.22	2.64	**0.16**
2400	1.30	0.13	1.72	**0.03**
4000	1.27	0.08	1.47	**0.03**
8000	1.14	0.04	1.01	**0.02**

Table 2: Local map reconstruction performance MSE

(a) MuSHR

# FT eps.	Scratch		PACT	
	F	T	F	T
100	0.566	0.425	**0.365**	0.503
1000	0.47	0.182	0.326	**0.167**
5500	0.43	0.143	0.300	**0.134**

(b) Habitat

# FT eps.	Scratch		PACT	
	F	T	F	T
800	0.368	6.4e-4	0.166	**5.2e-4**
2400	0.326	**2.61e-4**	0.1	2.78e-4
4000	0.297	**1.72e-4**	0.075	1.98e-4
8000	0.26	**1.56e-4**	0.053	1.78e-4

H2 - The PACT representation is a good starting point for multiple tasks: One of our major objectives with PACT is to show that it can serve as a general common representation, useful for a diverse set of downstream tasks. Leveraging a common representation is advantageous for a neural robotics architecture with real-time compute constraints because we can use a single expensive feature extraction module followed by lightweight decoders for each task as opposed to multiple individual large networks. In addition, the training process for downstream tasks should require less data and compute if initialized with a good representation.

In Table 3 we make a direct comparison between multi-head training for localization, mapping and navigation simultaneously from scratch versus initialization with PACT features. The table shows that PACT features can achieve superior performance for our mobile agent tasks. The performance gap holds especially in low-data regimes, showing that our transformer can effectively use the pre-training procedure to serve as a good-quality initialization.

H3 - Both perception and action are important for representation learning: As seen in section 2, previous methods for robotics representation learning work almost exclusively only with perception

Table 3: Joint multi-head training from scratch versus training from frozen PACT representation

(a) MuSHR

# of training episodes	Multi-head training from scratch		Finetuned PACT representation	
	Loc ATE	Map MSE	Loc ATE	Map MSE
100	18.5	0.59	**10.3**	**0.40**
1000	8.12	0.31	**5.46**	**0.25**
5500	4.87	0.22	**3.84**	**0.21**

(b) Habitat

# of training episodes	Multi-head training from scratch		Finetuned PACT representation	
	Loc ATE	Map MSE	Loc ATE	Map MSE
800	1.233	0.173	**0.802**	**0.055**
2400	0.263	0.023	**0.185**	**0.010**
4000	0.239	0.016	**0.0.126**	**0.008**
8000	0.119	0.007	**0.105**	**0.006**

features [15, 40, 41], treating model dynamics as a separate entity. Our hypothesis is that a feature representation method that combines both perception and action jointly can encode more useful information for downstream tasks.

Similar to some of the analysis in [42], we conduct an ablation study to understand the effect of different auxiliary pre-training losses. Table 4 displays metrics for different downstream task decoders trained on different frozen PACT representations: exclusively using state or action prediction losses, or using both simultaneously during pre-training. We see that representations that include a combination of both state and action prediction losses yields the richest representations for localization and mapping.

Table 4: Pre-training comparison

# of episodes	Task	State only	Action only	State and Action
MuSHR	Localization ATE	52.0	4.34	**4.21**
	Mapping MSE	0.609	0.331	**0.326**
Habitat	Localization ATE	2.92	3.218	**1.512**
	Mapping MSE	0.596	0.110	**0.053**

5.2 Additional Ablations

Model size and transformer sequence length: We analyze the pre-training model performance for MuSHR as a function of the number of tokens used for training and as a function of model capacity, expressed as the number of layers of the transformer architecture. We evaluated 4 model sizes (3, 6, 12, 24 layers), as shown in Fig 7a. Performance is measured in terms of average number of meters traversed over 150 model deployments in a realistic floor plan.

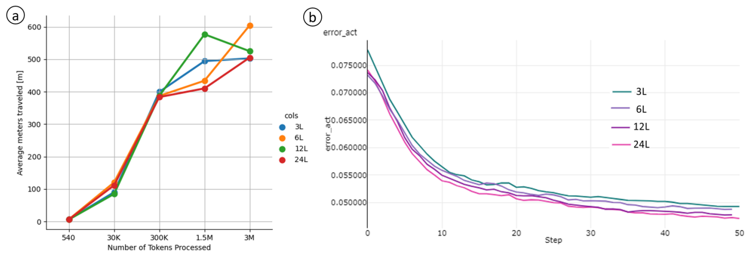


Figure 7: a) Effect of model and dataset sizes on pre-training performance measured as the average number of meters traversed until a crash; b) Effect of transformer layer depth on action prediction MAE.

In general we see an improvement in model performance as we increase the number of training tokens. Interestingly, larger models did not necessarily result in better performance for robot navigation.

Even though larger models consistently presented better loss values for action prediction on a static dataset, (Fig. 7 b), when it comes to real-time deployment the larger network capacity introduces inference delays that become a disadvantage and lead to earlier crashes. For example, while LiDAR perception measurements arrive to the vehicle every 0.077s (13Hz), the largest model of 24 layers takes on average 0.023s for inference with a RTX3090 GPU, roughly 40% longer the 3 layer model (0.016s). These time differences can amount to even larger performance gaps in small embedded systems, and further emphasize the importance of multiple downstream task architectures sharing a common representation branch for real-time robotics applications.

Generative properties of the pre-trained model: Analogous to how a model like GPT-3 operates, we can bias the future distribution of states and actions produced by our pre-trained model by prompting the transformer sequence with specific initial values. Fig. 8 displays heatmaps with state distributions for multiple runs where the car was initialized in the same position and orientation, and the only difference being the prompting of the very first 15 action tokens. The figure highlights that prompting with straight trajectories results in future actions that tend to keep the vehicle on a straighter course when compared to the actions that are generated from prompts including turns. It demonstrates the effectiveness of PACT to generate sequences of actions that mimic a desired behavioral prompt.

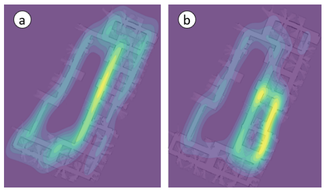


Figure 8: Visualization of state distribution heatmaps for different prompting sequences.

Real-world experiments: Even thought the entirety of the pre-training data was generated in a robotics simulator, we tested how transferable the transformer features and action prediction decoder were when deployed *in the wild*. We deploy the MuSHR car in real life and show results in Figure 9. Longest runs we observed were in the order of 80 meters, despite the sim-to-real gap, demonstrating the robustness of our model.

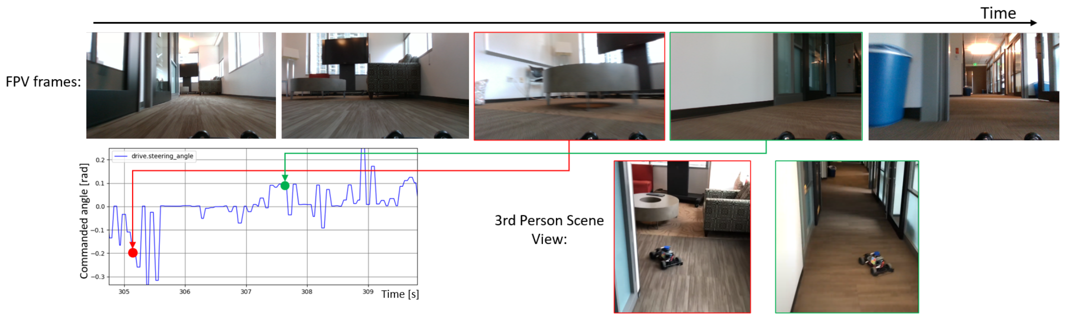


Figure 9: Real-world deployment of pre-trained model

6 Conclusion and Discussion

We presented PACT, a Perception-Action Causal Transformer architecture aimed at building a general purpose representation from robot data in a self-supervised fashion. Through autoregressive prediction of states and actions over time, we showed that our model can implicitly encode robot states, dynamics and behaviors. Such a representation, built in a robot-specific way, can function as a single starting point to achieve distinct tasks such as safe navigation, localization and mapping.

We demonstrated our approach in two navigation scenarios using a wheeled robot and a simulated agent, and showed that fine-tuning task-specific networks for localization and mapping on top of the pre-trained models results in better performance compared to models that are trained from scratch. In addition, sharing a single common representation across tasks is advantageous to speed up real-time deployment, opening a promising avenue towards foundational models for robotics and control tasks.

When considering future work, we highlight the fact that PACT does not necessarily require *optimal* demonstrations to learn statistical patterns between state-action tokens, which lowers the burden of developing expert trajectories towards collecting *reasonable* demonstrations. We are interested in a more formal analysis of how demonstration quality affects the representation performance.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[3] Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*, 2022.

[4] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021.

[5] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*, 2019.

[6] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

[7] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021.

[8] Hassan Akbari, Liangzhe Yuan, Rui Qian, Wei-Hong Chuang, Shih-Fu Chang, Yin Cui, and Boqing Gong. Vatt: Transformers for multimodal self-supervised learning from raw video, audio and text. *Advances in Neural Information Processing Systems*, 34:24206–24221, 2021.

[9] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent, 2022. URL https://arxiv.org/abs/2205.06175.

[10] Elia Kaufmann, Antonio Loquercio, Rene Ranftl, Alexey Dosovitskiy, Vladlen Koltun, and Davide Scaramuzza. Deep drone racing: Learning agile flight in dynamic environments. In *Conference on Robot Learning*, pages 133–145. PMLR, 2018.

[11] Qing Li, Shaoyang Chen, Cheng Wang, Xin Li, Chenglu Wen, Ming Cheng, and Jonathan Li. Lo-net: Deep real-time lidar odometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8473–8482, 2019.

[12] Ioan Andrei Barsan, Shenlong Wang, Andrei Pokrovsky, and Raquel Urtasun. Learning to localize using a lidar intensity map. *arXiv preprint arXiv:2012.10902*, 2020.

[13] Dmitry Kalashnikov, Jacob Varley, Yevgen Chebotar, Benjamin Swanson, Rico Jonschkowski, Chelsea Finn, Sergey Levine, and Karol Hausman. Mt-opt: Continuous multi-task robotic reinforcement learning at scale. *arXiv preprint arXiv:2104.08212*, 2021.

[14] Wenlong Huang, Igor Mordatch, Pieter Abbeel, and Deepak Pathak. Generalization in dexterous manipulation via geometry-aware multi-task learning. *arXiv preprint arXiv:2111.03062*, 2021.

[15] Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The unsurprising effectiveness of pre-trained vision models for control. *arXiv preprint arXiv:2203.03580*, 2022.

[16] Shuang Ma, Sai Vemprala, Wenshan Wang, Jayesh K Gupta, Yale Song, Daniel McDuff, and Ashish Kapoor. Compass: Contrastive multimodal pretraining for autonomous systems. *arXiv preprint arXiv:2203.15788*, 2022.

[17] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[18] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7464–7473, 2019.

[19] Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason Corso, and Jianfeng Gao. Unified vision-language pre-training for image captioning and vqa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13041–13049, 2020.

[20] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[21] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in Neural Information Processing Systems*, 34, 2021.

[22] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3674–3683, 2018. doi: 10.1109/CVPR.2018.00387.

[23] Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13134–13143, 2020. doi: 10.1109/CVPR42600.2020.01315.

[24] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *Conference on Robot Learning*, pages 394–406. PMLR, 2020.

[25] Khanh Nguyen and Hal Daumé III. Help, anna! visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning. *arXiv preprint arXiv:1909.01871*, 2019.

[26] Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. The regretful agent: Heuristic-aided navigation through progress estimation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6725–6733, 2019. doi: 10.1109/CVPR.2019.00689.

[27] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022.

[28] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[30] Francesco Giuliari, Irtiza Hasan, Marco Cristani, and Fabio Galasso. Transformer networks for trajectory forecasting. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 10335–10342. IEEE, 2021.

[31] Arthur Bucker, Luis Figueredo, Sami Haddadin, Ashish Kapoor, Shuang Ma, and Rogerio Bonatti. Reshaping robot trajectories using natural language commands: A study of multi-modal data alignment using transformers. *arXiv preprint arXiv:2203.13411*, 2022.

[32] Devendra Singh Chaplot, Deepak Pathak, and Jitendra Malik. Differentiable spatial planning using transformers. In *International Conference on Machine Learning*, pages 1484–1495. PMLR, 2021.

[33] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34, 2021.

[34] Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34, 2021.

[35] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[36] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[37] Siddhartha S. Srinivasa, Patrick Lancaster, Johan Michalove, Matt Schmittle, Colin Summers, Matthew Rockett, Joshua R. Smith, Sanjiban Chouhury, Christoforos Mavrogiannis, and Fereshteh Sadeghi. MuSHR: A low-cost, open-source robotic racecar for education and research. *CoRR*, abs/1908.08031, 2019.

[38] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[39] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4918–4927, 2019.

[40] Markus Wulfmeier, Arunkumar Byravan, Tim Hertweck, Irina Higgins, Ankush Gupta, Tejas Kulkarni, Malcolm Reynolds, Denis Teplyashin, Roland Hafner, Thomas Lampe, et al. Representation matters: improving perception and exploration for robotics. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6512–6519. IEEE, 2021.

[41] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Ken Goldberg, and Pieter Abbeel. Daydreamer: World models for physical robot learning. *arXiv preprint arXiv:2206.14176*, 2022.

[42] Mengjiao Yang and Ofir Nachum. Representation matters: Offline pretraining for sequential decision making. In *International Conference on Machine Learning*, pages 11784–11794. PMLR, 2021.

A Training Parameters

The training and network parameters used for the main paper experiments are described in the table below, unless where noted differently.

Hyperparameter	Value
# of layers	12
# of attention heads	8
Embedding length	128
Sequence length	16
Batch size	32
Pre-training Learning rate	6e-4
Finetuning Learning rate	6e-5
Weight decay for transformer weights	1e-1
Weight decay for all other layers	1e-4
LR schedule	Ramp-up (5% of total tokens) followed by decay
Dropout	0.1