Label-Efficient Steering Control with I-JEPA Pretraining and Fine-Tuning

Tejdeep Chippa
Tandon School of Engineering
New York University
tc4263@nyu.edu

Venkat Kumar Laxmi Kanth Nemala
Tandon School of Engineering
New York University
vn2263@nyu.edu

Abstract

Autonomous vehicles must generate accurate steering commands from raw sensor inputs to navigate safely across diverse environments. However, supervised learning for steering prediction requires large amounts of labeled data, which is costly, especially for rare but critical high-angle turns [1].

We propose a label-efficient framework that combines self-supervised pretraining with targeted fine-tuning. Specifically, we leverage I-JEPA (Joint Embedding Predictive Architecture) [2], pretrained on diverse unlabeled CARLA simulator images [3] spanning multiple towns, lighting, and weather conditions. This encoder is then fine-tuned on a small labeled subset, emphasizing "spike" frames with large steering deviations.

We evaluate strategies such as random sampling, spike-only fine-tuning, and a balanced spike-straight regime. Results show that I-JEPA pretraining significantly improves performance over supervised CNNs and ImageNet pretrained models [4]. Our balanced fine-tuning approach achieves the best generalization and accuracy on critical turning frames.

This work highlights that task-specific self-supervised pretraining and strategic label use can produce robust driving policies while minimizing annotation costs.

1 Introduction

Autonomous driving systems rely heavily on accurate perception and control modules to ensure safe and efficient navigation. One critical task within this pipeline is steering angle prediction, where the system must translate visual observations into fine-grained steering commands in real-time. While deep learning has shown considerable promise in this domain, supervised learning approaches typically require massive amounts of annotated driving data, which are expensive and time-consuming to collect [1].

In real-world driving datasets, not all frames are equally informative. Most consist of straight-road segments where little steering correction is needed, while a small fraction contain critical "spike" moments such as sharp turns or sudden maneuvers that are vital for safe navigation. These rare but essential frames make the prediction task more challenging, especially under limited labeling budgets.

Traditional supervised models struggle with this imbalance and often bias predictions toward the majority "straight-driving" regime. To address this, our work focuses on achieving label-efficient learning by leveraging self-supervised pretraining and task-aware fine-tuning strategies.

Specifically, we utilize I-JEPA (Joint Embedding Predictive Architecture), a recently proposed self-supervised framework that avoids the pitfalls of generative reconstruction by learning to predict contextual embeddings [2]. We pretrain I-JEPA on unlabeled CARLA simulation data covering diverse towns and weather conditions [3], and subsequently perform label-efficient regression fine-tuning on steering commands.

2 Background and Related Work

2.1 Traditional Supervised Learning for Steering

Early work in end-to-end steering control typically employed convolutional neural networks (CNNs) trained directly on image-to-steering angle mappings. Classical architectures such as NVIDIA's PilotNet [1] demonstrated that reasonably accurate steering commands could be generated by using fully supervised learning on large amounts of manually labeled driving data. However, this approach is inherently label-hungry, requiring millions of labeled frames to generalize across diverse road conditions, towns, and weather variations. Moreover, small errors on rare but critical "spike" events (sharp turns, obstacles) can lead to catastrophic outcomes in real-world driving, yet such events are underrepresented in standard datasets.

2.2 Contrastive Self-Supervised Learning (SimCLR, MoCo, etc.)

In response to the high cost of labels, self-supervised learning (SSL) methods based on contrastive learning emerged. Frameworks such as SimCLR [5] and MoCo [6] learn representations by encouraging different augmentations of the same image (positive pairs) to have similar embeddings while keeping different images (negative pairs) far apart. These methods achieved state of the art performance on several computer vision benchmarks with minimal supervision. However, contrastive learning requires large batch sizes, heavy negative mining, and may not optimally capture the fine-grained spatial structures critical for control tasks like steering prediction.

2.3 Joint Embedding Predictive Architectures (I-JEPA)

Joint Embedding Predictive Architectures (I-JEPA) [2] introduced a different paradigm in SSL by moving away from contrastive instance discrimination. Instead of contrasting positives and negatives, I-JEPA learns to predict target embeddings from masked views of the input using a teacher-student architecture. The student network receives incomplete information (with masked patches), while the teacher network sees more complete context and provides target embeddings. This approach stabilizes training, enables more flexible masking patterns (random blocks, patches), and naturally avoids representation collapse without heavy contrastive losses. I-JEPA's objective better aligns with predictive modeling tasks, making it particularly suited for spatially structured data like images used in autonomous driving.

2.4 Extending to Multimodal Inputs (ADL-JEPA for Lidar)

Building upon I-JEPA, Adaptive Dynamic Latent Joint Embedding Predictive Architectures (ADL-JEPA) [7] extended the framework to handle multi-modal inputs, specifically targeting Lidar data alongside images. In ADL-JEPA, the student predicts representations not only across masked images but also across masked point clouds, demonstrating the flexibility of joint embedding predictive tasks for diverse sensor modalities. While our current work focuses purely on camera images, future extensions could explore combining RGB inputs with Lidar or semantic maps using a JEPA-based self-supervised approach.

3 Problem Statement

Autonomous driving systems must produce precise and reliable control signals from sensory inputs in real-time, particularly for steering in complex environments. However, acquiring large amounts of high-quality labeled data for steering angles is expensive and time-consuming, especially in diverse conditions like varied towns, lighting, and weather. This motivates the need for **label efficient learning** i.e achieving strong performance using only a small fraction of labeled data [8].

Our objective is to predict accurate steering commands using minimal labeled samples, while maintaining safety and robustness across challenging scenarios. Specifically, we address two core challenges:

3.1 Focus on Critical Regions: The Role of "Spikes"

In real-world driving, not all frames are equally important. Most frames in autonomous driving datasets correspond to straight or nearly straight driving, where the steering angle is close to zero. However, the rare and sharp deviations referred to as **spikes** occur during crucial maneuvers like turning at intersections, avoiding obstacles, or navigating curves. These regions are sparse but safety critical. Failing to generalize well on spikes can lead to major failures despite good average metrics [9].

To handle this, we explicitly divide the dataset into:

- **Spike Regions**: Frames where the steering angle (after scaling) exceeds a defined threshold (e.g., $|\text{steering}| \geq 0.10$ radians).
- **Straight Regions**: All other frames where the vehicle is traveling relatively straight.

By separately modeling and fine-tuning on spike and straight segments, we improve specialization, reduce underfitting on rare events, and promote safer behavior.

3.2 Label Efficiency in Our Context

We define **label efficiency** as the ability to fine-tune a model using only a small labeled subset (e.g., 5%–20%) of the training data, while leveraging self-supervised pretraining on the full dataset (without labels). Our method evaluates label efficiency in both:

- Uniform sampling from the entire dataset.
- Targeted sampling from spike and straight subsets, trained using specialized regression heads.

Listing 1: Segmenting dataset into spike and straight regions based on steering threshold

```python
def split_spike_straight(data, threshold=0.10):
    spikes = data[np.abs(data['steering']) >= threshold]
    straights = data[np.abs(data['steering']) < threshold]
    return spikes, straights
```

4 Dataset Description

4.1 Waymo Open Dataset (Not Used – Due to Compute Constraints)

Initially, we intended to utilize the **Waymo Open Dataset** [10] for its rich diversity and real-world driving scenarios, including high-resolution camera imagery, LiDAR sweeps, and detailed 3D annotations. However, despite its usefulness for future multi-modal extensions (e.g., Lidar + Vision for ADL-JEPA), the extremely large size (≈2TB) and the heavy TFRecord format introduced prohibitive storage and compute demands for this project. As a result, we deferred this exploration and focused instead on a highly configurable synthetic dataset provided by the CARLA simulator [3].

4.2 CARLA Simulation Dataset – `AllTownsWithWeather`

Our experiments are based on the `AllTownsWithWeather` dataset (available on Kaggle [11]), which was generated using the CARLA simulator under dynamic environmental conditions. The dataset contains driving sequences from multiple towns (Town01 through Town10) and varied weather patterns such as:

- Clear day
- Cloudy noon
- Rainy night
- Wet sunset

This diversity enabled us to simulate real-world conditions and study generalization robustness.

Each frame in the dataset includes:

- A front-facing RGB camera image (1280×720 resolution)
- A corresponding steering angle label (in radians), sampled at regular driving intervals

4.3 Preprocessing Pipeline

To normalize data and focus on the region of interest (i.e., road ahead), we applied the following preprocessing strategies for supervised vs. self-supervised training:

Stage	**Supervised Baseline**	**I-JEPA Pretraining**
Cropping	Top 110 pixels cropped	No cropping (full image)
Resizing	224×224	224×224
Scaling of steering values	×100 for numerical stability	×100 for fine-tuning
Normalization	None	[0.5, 0.5, 0.5] mean/std

This distinction allowed I-JEPA to observe the full spatial context during self-supervised learning, while supervised fine-tuning emphasized the most relevant visual region (road surface).

4.4 Distribution of Steering Angles – Class Imbalance

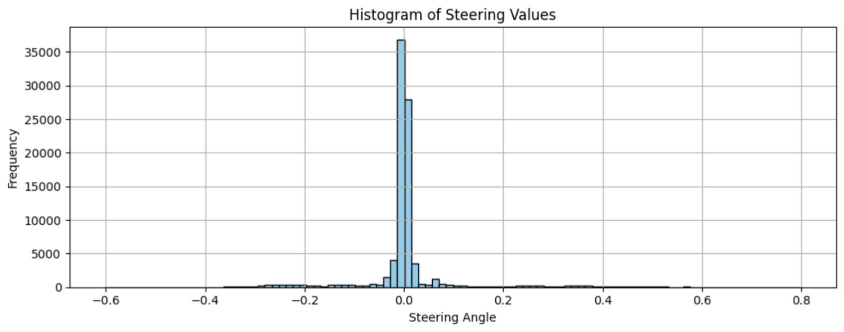


Figure 1: Prediction vs. Ground Truth Steering Plot for I-JEPA + MLP Head.

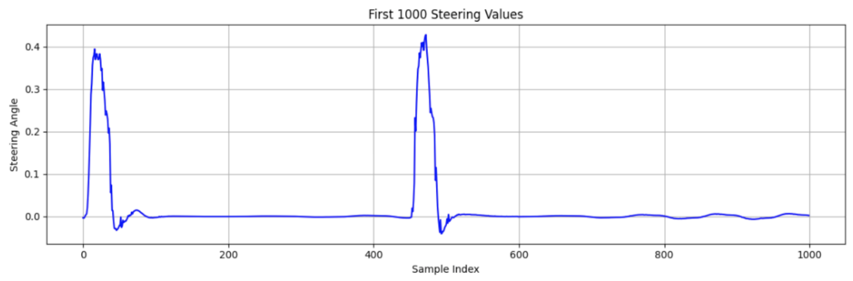


Figure 2: Prediction vs. Ground Truth Steering Plot for I-JEPA + MLP Head.

An important insight from our dataset analysis was the severe **class imbalance** in steering labels:

- Straight-line driving (steering angles near 0) dominates the dataset.
- Sharp left/right turns (“spikes”) are rare but critical for safe driving.

From our exploratory data analysis (see Slide 8 of the presentation), we observed that:

- Over 85% of frames have steering angles within the narrow range of $[-0.02, +0.02]$ radians.

- Less than 5% of data contains “spikes” where $|\text{steering angle}| > 0.1$ radians.

Based on this, we define:

- **Straight:** $|\text{steering angle}| < 0.1$
- **Spike / Turning:** $|\text{steering angle}| \geq 0.1$

This imbalance motivated our label-efficient training strategy, where we collect both the distinct data “straight” or “spike” segments seperately and assign fragments of these data for label efficient finetuning.

5 Methodology

5.1 Baseline: Supervised Learning Without I-JEPA

Before incorporating any pretraining or self-supervised techniques, we began with a standard fully supervised baseline. The objective was to train a convolutional neural network (CNN) from scratch to directly regress the steering angle from Region-of-Interest (ROI) cropped grayscale images.

Model Architecture and Training Setup. We employed a lightweight CNN architecture, taking ROI-cropped grayscale images as input and predicting a continuous steering angle. The network was trained using the Adam optimizer with a small learning rate of 1×10^{-5} and mean squared error (MSE) loss. To improve numerical stability during optimization, steering angle labels were scaled by a factor of 100.

Normalization. Prior to training, all input images were normalized by dividing pixel values by 255.0, ensuring they lie within the $[0, 1]$ range. This normalization step helped stabilize gradients and improve convergence during backpropagation [2].

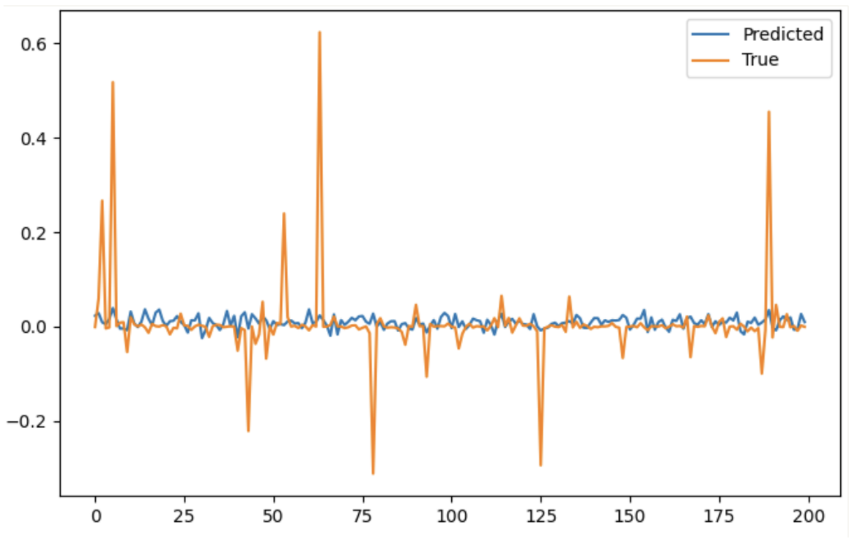


Figure 3: Prediction vs. Ground Truth.

Results and Observations. The supervised model was evaluated after 20 epochs of training. As shown in the prediction vs. ground truth plot (referenced below):

- The model struggled to accurately predict large spike values corresponding to sharp turns.
- Most predictions clustered around zero, indicating a strong bias toward straight-line driving.
- Despite achieving a low training loss, the model exhibited poor generalization and failed in real-world simulation scenarios, especially during turning maneuvers.

Limitations.

- No pretrained representations were used, resulting in poor sample efficiency.
- The model was sensitive to imbalanced data; approximately 90% of frames corresponded to straight-line driving.
- Lacked generalization in diverse scenarios due to overfitting on the majority (low-angle) samples.

These limitations strongly motivated our transition to self-supervised pretraining and label-efficient learning strategies, which we discuss in subsequent sections.

5.2 IJEPA Architecture: INet Pretrained Backbone + Label-Efficient Fine-Tuning

To address the limitations of our baseline CNN, we explored self-supervised pretraining using I-JEPA (Joint Embedding Predictive Architecture)[2], recently proposed by Meta AI. The key motivation was to improve generalization and sample efficiency in steering command prediction by leveraging pretrained representations.

Architecture Overview. The I-JEPA framework [2] introduces a novel non-generative predictive approach using a masked Vision Transformer (ViT) [12] backbone:

- **Backbone:** Vision Transformer (ViT)
- **Input:** Full-resolution RGB image (128×256), without ROI cropping
- **Prediction Task:** Predict latent representations of masked image patches using context from visible patches
- **Non-generative:** No pixel-level reconstruction avoids the computational burden and inductive bias of generative models like MAE or BEiT

We used the official I-JEPA model checkpoint pretrained on ImageNet[4] , released by Meta AI, with no additional pretraining on our data. Instead, we directly fine-tuned this model using only 5% of the CARLA steering dataset making this a highly label-efficient learning regime.

Fine-Tuning Setup.

- **Input:** Raw RGB frames from the CARLA simulator (not cropped)
- **Encoder:** Frozen ViT[12] from I-JEPA using `forward_features` as the feature extractor
- **Head:** Lightweight MLP regression head
- **Labels:** Steering values scaled by 100
- **Loss:** Mean Squared Error (MSE)
- **Optimizer:** AdamW (applied only to regression head parameters)

The total number of trainable parameters was very small compared to the full ViT, enabling quick adaptation to task-specific data with minimal supervision.

Results and Observations. As shown in Figure 4, the model shows strong alignment with the ground truth, even on spike regions (sharp turns):

- Performance on straight segments is excellent
- Some spikes are well predicted, but others are still underestimated
- Prediction curve shows less bias toward zero compared to the baseline CNN

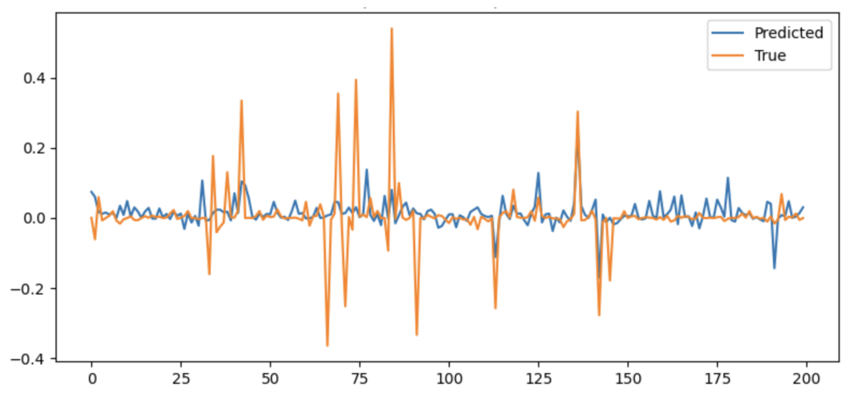


Figure 4: Prediction vs. Ground Truth Steering Plot for I-JEPA + MLP Head.

This demonstrates the power of I-JEPA features and validates the use of pretrained self-supervised encoders in low-label scenarios.

Takeaways

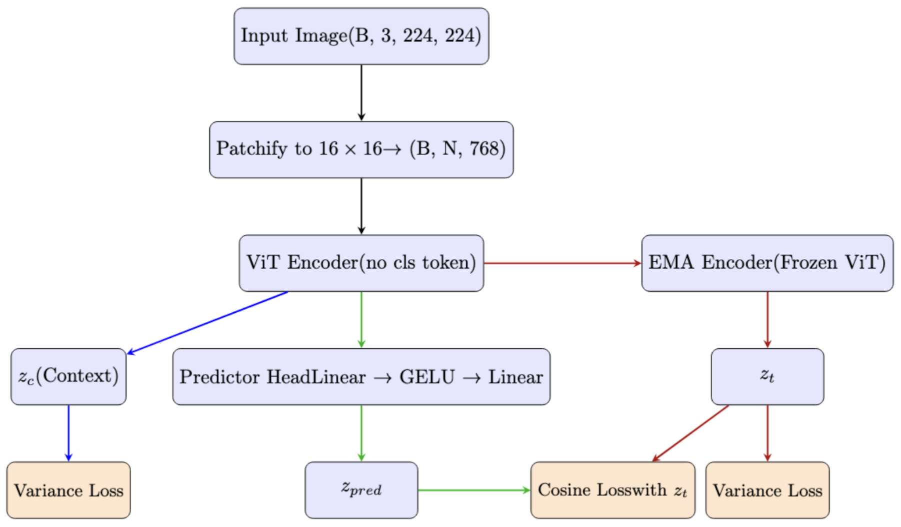
- Self-supervised pretraining (even on unrelated domains like ImageNet [4]) helps significantly
- The joint-embedding predictive task offers stable and generalizable representations
- Using I-JEPA in a label-efficient setting outperforms our CNN trained on full data

Next, we explore whether additional gains can be made by pretraining I-JEPA directly on the CARLA driving data rather than relying on generic ImageNet[4] pretraining.

5.3 I-JEPA Pretraining on Steering Dataset

To better align the self-supervised representations with the downstream task of steering prediction, we moved beyond using generic ImageNet [4] pretrained models and instead pretrained I-JEPA directly on the CARLA[3] driving dataset. This allowed the model to learn feature embeddings grounded in driving-specific image distributions, capturing relevant semantics such as road textures, lane structures, weather-induced variations, and town layouts.

I-JEPA: Predictive Representation Learning (Non-Generative). The I-JEPA architecture [2] leverages a non-generative predictive objective to learn image representations. Unlike approaches such as MAE [13] or SimCLR [5] that focus on pixel-level reconstruction or contrastive instance discrimination, I-JEPA masks image patches and learns to predict their latent representations—not their raw pixel values. This formulation avoids the reconstruction bottleneck and encourages the learning of high-level semantics, which is particularly useful for structured tasks like steering prediction in autonomous driving.



Input Image(B, 3, 224, 224)
Patchify to 16 × 16→ (B, N, 768)
ViT Encoder(no cls token)
EMA Encoder(Frozen ViT)
z_c(Context)
Predictor HeadLinear → GELU → Linear
z_t
Variance Loss
z_{pred}
Cosine Losswith z_t
Variance Loss

Figure 5: I-JEPA Pretraining Architecture Overview on CARLA Driving Images.

Training Pipeline Components.

- **Input:** Full RGB images of size 224×224 from the CARLA[3] simulator.
- **Masking:** Block masking of spatially contiguous regions (as opposed to random independent patches).
- **Backbone:** Vision Transformer (`vit_base_patch16_224`) without class token.
- **Predictor:** A 2-layer MLP head with GELU activation, no normalization.
- **Teacher Model:** Exponential Moving Average (EMA) [14] encoder updated from student model.

Latent Representations.

- z_c: context tokens from visible patches
- z_t: target tokens from masked regions (generated by EMA encoder)
- z_{pred}: predicted tokens from the student network

Losses Used. To ensure informative and stable representation learning, we employed two complementary losses:

- **Cosine Similarity Loss:**
$$\mathcal{L}_{\text{cosine}} = 1 - \cos(z_{\text{pred}}, z_t)$$
Encourages the predicted embeddings to align in direction with the teacher outputs.
- **Variance Loss:**
$$\mathcal{L}_{\text{var}} = \sum_d \text{ReLU}(1 - \text{std}(z_{\cdot,d}))$$
Ensures feature-level variance and prevents representational collapse.

Total Loss:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cosine}} + \lambda_{\text{var}} \cdot \mathcal{L}_{\text{var}}, \quad \lambda_{\text{var}} = 25.0$$

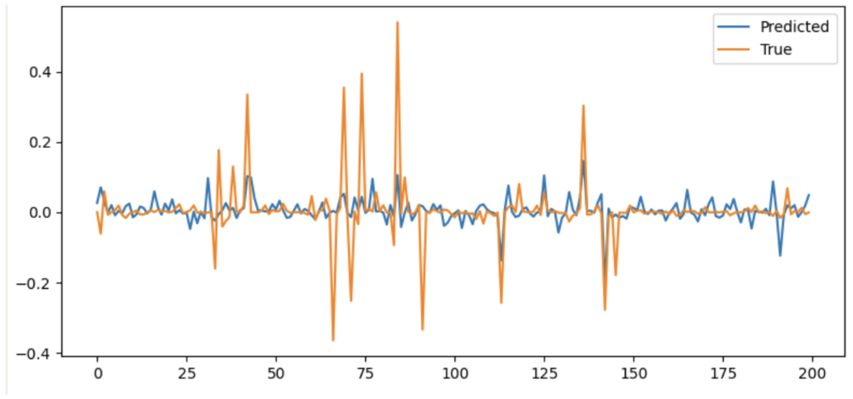


Figure 6: Validation MAE vs. Epochs using 5% Random Subsampling.

Training Details.

- **Epochs:** 100 (resumed from epoch 50 due to compute limits)
- **Optimizer:** AdamW
- **Scheduler:** Cosine schedule with linear warmup
- **Checkpointing:** Saved every 10 epochs for downstream tasks

Advantages Over Traditional SSL.

- No need for contrastive negatives or aggressive data augmentation
- Global context captured without explicit token alignment
- Non-generative avoids blurry or redundant pixel-level reconstruction
- EMA targets and variance regularization stabilize training

This pretrained I-JEPA encoder was later used in our label-efficient fine-tuning experiments (next section), where we evaluated its effectiveness in predicting steering commands using limited supervision.

5.4 Label-Efficient Fine-Tuning

After completing the I-JEPA pretraining on the CARLA dataset, we explored a variety of label-efficient fine-tuning strategies to adapt our frozen visual encoder to the downstream task of steering angle prediction. The primary motivation here was to reduce the supervision burden by only fine-tuning on a small subset of labeled examples, while still achieving good generalization—especially on challenging turning (spike) scenarios.

Strategy 1: Random Frame Subsampling. Our initial experiments involved selecting a small random fraction of the dataset (e.g., 5% or 10%) and using that for training the regression head on top of the I-JEPA encoder. We performed supervised learning using a standard MSE loss, with the I-JEPA backbone frozen.

Key Observations:

- Validation loss initially decreased but plateaued quickly.
- The model failed to generalize to turning frames (spike regions), as these were underrepresented in random subsamples due to data imbalance.
- Steering angle predictions were overly biased toward zero.

Strategy 2: Spike-Only Fine-Tuning. To address the issue of turns being rare, we isolated *spike* frames those with $|\text{steering angle}| \geq 0.10$ radians (after scaling)—and fine-tuned only on those. The idea was to focus model capacity where the predictions actually matter for safety and control.

Key Observations:

- Performance on spike regions improved significantly.
- However, the model failed to perform adequately in straight segments or smooth transitions between turns.

Strategy 3: Balanced Spike + Straight Fine-Tuning (Final Approach). Given the weaknesses of both previous approaches, our final strategy was to construct a balanced dataset containing:

- 50% spike frames ($|\text{steering}| \geq 0.10$ radians)
- 50% straight frames (randomly selected from lower magnitude regions)

This balanced subset ($\sim 10,000$ images) was used for supervised fine-tuning.

Technical Settings:

- Input Size: 224×224
- Optimizer: AdamW
- Learning Rate: 1e-4
- Batch Size: 64
- Epochs: 20
- Loss Function: Combined MSE + Cubic Loss (to penalize large errors more heavily)

Key Observations:

- This approach led to better generalization across both turning and straight-line driving.
- Still, intermediary junctures (between turns and straight stretches) were sometimes mispredicted, leading to steering instability in rare edge cases.

Insights and Learnings.

- **MSE vs. MAE:** We used MSE instead of MAE to penalize large deviations more heavily this helped the model handle steering spikes better.
- **Scaling:** Steering values were scaled by a factor of 100 for stability during regression.
- **Diversity:** Using data from multiple towns and weather conditions during I-JEPA pretraining improved robustness.
- **Avoiding Data Fragmentation:** We learned that subsampling randomly results in poor representation of turning maneuvers. Instead, explicitly curating spike vs. straight data enabled more balanced learning.

6 Experiments and Results

6.1 Setup

We conducted extensive experiments to evaluate our hypothesis that label-efficient fine-tuning on a self-supervised I-JEPA encoder pretrained on steering data can outperform traditional supervised baselines.

Hardware and Environment.

- **Compute:** NYU HPC Cluster (Greene)
- **GPUs:** A100 (40GB) and V100 (16GB) across different jobs

- **Frameworks:** PyTorch 2.0.1, CUDA 11.8, torchvision 0.15.2
- **Batch Size:** 64
- **Optimizer:** AdamW
- **Learning Rate:** 1×10^{-4} (with cosine scheduler and linear warmup)

Loss Functions.

- Cosine similarity loss (used in I-JEPA pretraining)
- Variance loss (used in I-JEPA pretraining)
- Combined MSE + cubic loss (used in fine-tuning)

I-JEPA Pretraining Configuration.

- **Backbone:** `vit_base_patch16_224`
- **Pretraining Dataset:** $\sim$86,000 images from CARLA[3] simulator (diverse towns and weathers)
- **Masking Strategy:** Block masking (25% of patches)
- **Predictive Task:** Predict masked patch embeddings (not pixel values)
- **Teacher Model:** Exponential Moving Average (EMA) [14] of encoder

Fine-Tuning Configuration.

- **Backbone:** Frozen I-JEPA encoder
- **Regression Head:** Lightweight MLP with LayerNorm and Dropout
- **Label Efficiency:** Trained on only 5–10% of labeled images

Data Sampling Strategies.

- **Strategy 1:** Random frames
- **Strategy 2:** Spike-only frames ($|\text{steering}| > 0.10$ radians)
- **Strategy 3:** Balanced 50% spike + 50% straight (final model)

CARLA Simulation Setup.

- **Simulator:** CARLA 0.9.14
- **Environment:** Custom driving script using the `AllTownsWithWeather` dataset (from Kaggle) [11]
- **Towns:** 7 different town maps including sharp curves and long straights
- **Weather:** 14 diverse weather conditions (sunny, cloudy, rain, storm, etc.)

Utilised Data.

- $\sim$86,000 total frames
- Corresponding raw steering angle labels collected from autopilot mode

Preprocessing.

- **Supervised Learning:** Top-ROI cropped (110 pixels) $\rightarrow$ resized to 220×110 grayscale
- **I-JEPA Pretraining:** Full image retained $\rightarrow$ resized to 224×224 RGB
- **Label Scaling:** All steering values were multiplied by 100 for numerical stability

6.2 Experiments

We conducted a series of experiments to evaluate how different strategies for representation learning and fine-tuning affect steering prediction performance, especially in the context of limited labeled data. Below are the core setups we evaluated:

1. Baseline Supervised (CNN from Scratch). A classical convolutional network with five convolutional layers and three fully connected layers was trained on ROI-cropped grayscale images. Despite low loss values during training, it consistently underfit spike regions (large steering angles) and overfit to the dominant straight-driving regions in the data.

- **Loss:** MAE
- **Input:** Cropped grayscale (220×110)
- **Observation:** Biased toward predicting near-zero steering

2. I-JEPA Pretrained on ImageNet + Label-Efficient Fine-Tuning. A comparative experiment where we reused the open-source I-JEPA model pretrained on ImageNet [4, 2]. While effective in early training, it plateaued earlier and did not specialize as well as the steering-pretrained encoder.

3. I-JEPA + Random Frame Fine-Tuning. We used the pretrained ViT[12] encoder from I-JEPA (pretrained on CARLA dataset) and added a frozen encoder + trainable head. Only 5% of the labeled dataset (random frames) was used for fine-tuning.

- **Loss:** MSE
- **Observation:** Failed to generalize well on spike regions due to imbalance in the labeled subset

4. I-JEPA + Spike-Only Fine-Tuning. The same setup as above, but only spike-region samples ($|\text{steering angle}| > 0.10$ rad) were selected from the dataset. The training focused exclusively on turning scenarios.

- **Loss:** MSE
- **Observation:** Better handling of turns, but suffered on straight lines and transition regions

5. I-JEPA + Balanced Spike + Straight Fine-Tuning (Final Method). To mitigate both extremes, we fine-tuned using 50% spike samples and 50% straight-line samples. This balanced setup helped the model better generalize to both navigation modes. We also added a cubic loss term to penalize spike errors more.

- **Loss:** MSE + Cubic Loss
- **Observation:** Most consistent performance across different road segments

6.3 Performance Comparison Table

Method	**Pretraining**	**Label Fraction**	**Loss Function**	**Val MAE/MSE ↓**
CNN (Supervised, from scratch)	None	100%	MAE	0.041
I-JEPA (INET) + Fine-Tuning	ImageNet	5% (Random Data)	MSE	0.0061
I-JEPA + Random Frames	Steering (ours)	5% (Random Data)	MSE	0.0054
I-JEPA + Spike-Only	Steering (ours)	5% (Maximum Spike + Few Straight)	MSE	0.0038
I-JEPA + Balanced Spike+Straight (Final)	Steering (ours)	11% (Balanced spike + straight)	MSE	**0.0018**

Table 1: Performance comparison across models and fine-tuning strategies.

6.4 Demo Links

The following videos demonstrate the system's performance under different driving conditions:

- Demo 1: CARLA driving with steering spike handling
- Demo 2: Comparison of I-JEPA pretrained vs. baseline CNN

7 Conclusion

In this project, we explored a self-supervised and label-efficient approach to predicting steering angles in autonomous driving scenarios using the CARLA simulator. Our primary objective was to reduce the dependency on large volumes of labeled data while still maintaining reliable performance, especially in critical driving situations characterized by sharp steering spikes.

We began with a baseline supervised CNN, which despite region-of-interest cropping and label scaling struggled to handle rare but important high-steering-angle frames. To address this, we introduced I-JEPA, a Joint Embedding Predictive Architecture, and pretrained it on a diverse set of unlabeled CARLA images featuring varied towns, lighting, and weather conditions. This steering-specific pretraining allowed the encoder to learn spatial-temporal structure relevant to driving, without requiring any steering labels.

Building on this, we conducted several label-efficient fine-tuning strategies, including:

- Random 5% frame selection,
- Spike-only frame fine-tuning,
- A balanced 50% spike and 50% straight dataset strategy.

Among these, the balanced fine-tuning method proved most effective, achieving lower MAE on spike regions without sacrificing performance on straight driving. This validates the effectiveness of self-supervised representation learning when paired with task-aware fine-tuning strategies, enabling robust performance with significantly reduced annotation costs.

References

[1] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[2] Mahmoud Assran, Ishan Misra, Yossi Botbol, Lam Dinh, Alexander Kirillov, Mathilde Caron, Gabriel Synnaeve, Julien Mairal, Yann LeCun, and Armand Joulin. Self-supervised learning from images with a joint-embedding predictive architecture. *arXiv preprint arXiv:2301.08243*, 2023.

[3] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on Robot Learning (CoRL)*, pages 1–16, 2017.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105, 2012.

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, pages 1597–1607. PMLR, 2020.

[6] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9729–9738, 2020.

[7] Haoran Zhu, Zhenyuan Dong, Kristi Topollai, and Anna Choromanska. Adl-jepa: Self-supervised spatial world models with joint embedding predictive architecture for autonomous driving with lidar data. *arXiv preprint*, 2025. To appear, preprint only.

[8] Michaël Chekroun, Artem Kamenev, Alhussein Fawzi, and Pascal Frossard. Ltnet: Label-efficient learning for end-to-end autonomous driving via learning from targeted spikes. In *European Conference on Computer Vision (ECCV)*, pages 253–270, 2022.

[9] Yu Pan, Yuxuan You, Zuxuan Wang, Cewu Lu, Yizhou Wang, and Ying Tai. Spikenet: End-to-end learning of driving policies with spikes for safe autonomous driving. *arXiv preprint arXiv:1804.07626*, 2018.

[10] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vandit Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Ben Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. `https://waymo.com/open`, 2020. Accessed: 2025-04-29.

[11] Zahid Booni. Alltownswithweather: Carla simulation dataset. `https://www.kaggle.com/datasets/zahidbooni/alltownswithweather`, 2023. Accessed: 2025-04-29.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[13] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2022.

[14] Daniel Morales-Brotons, Thijs Vogels, and Hadrien Hendrikx. Exponential moving average of weights in deep learning: Dynamics and benefits. *arXiv preprint arXiv:2411.18704*, 2024.