Multimodal Contrastive Learning for Zero-Shot Instruction-Following Robot with Synthetic Data

Washington Kamadi
Pan African University Institute for Basic Sciences, Technology and Innovation (PAUSTI), Kenya
washingtonkigan@gmail.com (corresponding author)

Jackson Githu Njiri
Department of Mechatronic Engineering, Jomo Kenyatta University of Agriculture and Technology (JKUAT), Kenya
jackgithu@gmail.com

Samuel Kangwagye
Robotics and Automation Group, Department of Materials and Production, Aalborg University, Denmark | Department of Mechanical and Production Engineering, Kyambogo University, Uganda
samkan@mp.aau.dk

Shohei Aoki
Department of Mechatronic Engineering, Jomo Kenyatta University of Agriculture and Technology (JKUAT), Kenya
shoaok@gmail.com

Received: 20 February 2026 | Revised: 22 March 2026 and 1 April 2026 | Accepted: 3 April 2026

 DOI: https://doi.org/10.48084/etasr.18291

ABSTRACT

Robot trajectory prediction is heavily dependent on large-scale real-world demonstrations, which limit scalability, increase data acquisition costs, and eventually prevent zero-shot generalization. To address this limitation, this paper introduces Zero-Shot Task Learning (ZSTL), a multimodal framework that uses structurally aligned synthetic data and contrastive learning to enable instruction-based trajectory generation without reliance on real-world demonstrations. ZSTL jointly encodes natural-language instructions, depth observations, LiDAR-derived spatial representations, and action trajectories within a joined embedding space, allowing cross-modal alignment and conditional behavior synthesis. The proposed architecture preserves modality structure prior to fusion by representing depth inputs as spatial tokens and LiDAR observations as temporal tokens. Together with a text token, these form a 101-token multimodal context attended over by a Transformer decoder to predict full 50-step trajectories with Gaussian uncertainty estimates. The system integrates a pretrained Bidirectional Encoder Representations from Transformers (BERT) language encoder, a ResNet-18 depth backbone, a one-dimensional convolutional LiDAR sequence encoder, and a two-layer Transformer decoder comprising approximately 125M parameters. Training was conducted entirely on a procedurally generated synthetic dataset of 5,000 samples for 50 epochs. The results demonstrate stable convergence, with the trajectory-negative log likelihood decreasing from 3.465 to -0.695 on validation data and the combined loss reaching -0.540 at epoch 18 under cosine annealed learning. The contrastive objective (InfoNCE, τ = 0.07) stabilized near 1.55, indicating consistent cross-modal alignment. The trajectory evaluation yielded an average final position error of 9.97 cm, a collision-free execution rate of 65.9%, and a task success rate of 59.2%, showing that structured synthetic supervision can support physically meaningful motion generation.

Keywords-multimodal learning; zero-shot task; synthetic data; sentence transformers

I. INTRODUCTION

Enabling robots to understand and execute human-like natural language instructions has been a challenge in robotics and Artificial Intelligence (AI). Despite steady progress in robotics and machine learning, existing systems still struggle with reliability and generalizability, i.e., traditional instruction following approaches normally require huge amounts of task-specific training data (e.g., large sets of human demonstrations or annotated trajectories) and carefully engineered vision action mappings [1]. These methods are highly dependent on their training environments and domains, which means that they struggle to generalize to novel instructions or changed environments without retraining [2, 3]. Additionally, acquiring the labeled data needed for multi-modal training, which includes aligning natural language with correct visual input, spatial contexts, and action sequences, is expensive and labor-intensive. Even in simulation, creating high-quality multimodal datasets demands careful design and considerable effort. Ensuring proper alignment across modalities (language, vision, spatial context, and actions) in these datasets is a major challenge, as annotation errors or mismatches can easily confuse learned models. Synthetic data can be effectively used to create valid datasets, eliminating the challenges associated with traditional data collection methods, which can be expensive and very labor-intensive [4]. Representation learning techniques in the vision-language field have emerged, aligning visual and textual information through contrastive or joint embedding training on huge datasets. Models like OpenAI's Contrastive Language-Image Pretraining (CLIP) [5] and subsequent frameworks such as Align Before Fuse (ALBEF) [6] and Bootstrapping Language-Image Pretraining (BLIP) [7] learn to pair images with natural language descriptions by training on hundreds of millions of image-caption pairs, resulting in representations that generalize in a zero-shot manner to concepts and tasks never seen during training. These models can recognize or retrieve visual concepts described in text without additional task-specific fine-tuning, showing the capabilities of cross-modal alignment learned through contrastive objectives. This suggests that a similar approach might benefit robotics: if a robot's perception, which includes vision and spatial understanding, and action space are embedded in the same semantic space as human language, the robot could potentially interpret new instructions or scenarios by analogy to what it learned, rather than relying purely on direct experience with every possible task.

Recent advances in Large Language Models (LLMs) and embodied multimodal systems further highlight the importance of unified perception, language, and action representations [8, 9]. Models such as PaLM-E [10] demonstrate that jointly encoding visual and textual inputs can support instruction-conditioned robot behavior within transformer architectures. However, these approaches typically depend on large-scale real-world data, complex simulation pipelines, or extensive robotic infrastructure, limiting accessibility and reproducibility. This has resulted in the evaluation of multimodal training approaches capable of achieving cross-modal grounding without relying on large datasets. However, existing multimodal and instruction-following approaches show several limitations that reduce their applicability to general-purpose robotic control [11, 12]. In addition, many robotics and embodied AI models use strong spatial and temporal pooling to combine sensory inputs into a single global vector. Although this works well for classification and skill selection [13, 14], it removes detailed spatial and temporal information that is important for trajectory-level reasoning and continuous control, especially in navigation and obstacle-aware tasks [15]. A unified and scalable framework that reduces reliance on large-scale real-world data while aligning language, vision, spatial context, and action within a shared representation space remains absent. Current approaches frequently depend on task-specific datasets, complex simulation platforms, or large robot deployments to support cross-task generalization. Many also require additional fine-tuning or supervision to operate in new environments, preventing true zero-shot capability.

This work presents how structurally aligned synthetic data, combined with multimodal contrastive learning, can be used in instruction-conditioned trajectory prediction without reliance on real-world robot demonstrations. To address these limitations, this paper introduces Zero Shot Task Learning (ZSTL), a multimodal framework for instruction-conditioned robot learning trained from generated synthetic supervision. This framework is designed around two complementary operational modes: an embedding mode and a trajectory prediction mode. Unlike pooled multimodal policies, ZSTL preserves structure before decoding by representing depth observations as spatial tokens and LiDAR observations as temporal tokens [16]. These tokens, in addition to a text token, form a 101-token context that the decoder attends to predict per-timestep Gaussian velocity commands. The main contributions of this work are summarized as:

- A multimodal learning framework that aligns language, depth, LiDAR, and action trajectories through contrastive training and conditional trajectory prediction.
- A structure-preserving token-level fusion strategy that retains spatial depth structure and temporal LiDAR structure, forming a 101-token context for Transformer-based trajectory decoding.
- A 2D synthetic data pipeline that enables scalable, perfectly aligned multimodal supervision without requiring real-world robot demonstrations.

II. METHODOLOGY

A. Problem Formulation

The navigation task used in this work is presented as a goal-directed motion planning problem under multimodal perception. The robot operates within an indoor environment filled with structural obstacles that constrain possible motion and require step-by-step decision making for successful motion planning and execution. Each scene defines a closed field containing walls, corridor-like paths, and random obstacles arranged to induce multiple directional changes before the robot can reach the goal. The robot is initialized at a random position in the field and must interpret natural language instructions while relying on visual observations and LiDAR measurements to safely navigate toward the required objective. Figure 1 illustrates a sample instance of the task. The

environment is structured to require several turns, hence preventing trivial paths and promoting meaningful planning behavior.

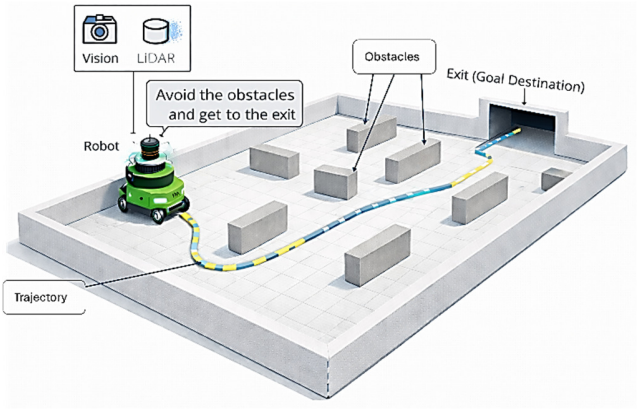

Fig. 1. Three-dimensional illustration of the navigation problem.

The system is designed to operate in two modes, described in detail as follows.

1) Embedding Mode

In embedding mode, the objective is to learn a shared representation space $\mathcal{E}$ that aligns heterogeneous robotic modalities. Let $\mathcal{T}$ denote the space of natural language instructions, $\mathcal{V}$ the visual observation space, $\mathcal{L}$ the LiDAR observation space, and $\mathcal{A}$ the action space. Modality-specific encoders $g_t: \mathcal{T} \to \mathcal{E}$, $g_v: \mathcal{V} \to \mathcal{E}$, $g_l: \mathcal{L} \to \mathcal{E}$, and $g_a: \mathcal{A} \to \mathcal{E}$ map each modality into the shared embedding space. The encoders are trained such that embeddings corresponding to the same instruction-scene-action tuple exhibit high cosine similarity. This mode supports cross-modal retrieval and semantic alignment through contrastive learning.

2) Trajectory Mode

In trajectory mode, the system performs conditional trajectory prediction. Given a visual observation $v \in \mathcal{V}$, a temporal sequence of 51 LiDAR scans $l_{1:51} \in \mathcal{L}^{51}$ providing short-term spatial context, and a natural language instruction $t \in \mathcal{T}$, the model predicts a continuous robot trajectory:

$$f: (\mathcal{V}, \mathcal{L}^{51}, \mathcal{T}) \to (\mu_{1:50}, \log \sigma^2_{1:50})$$

where each timestep i yields a Gaussian distribution over velocity commands $\mu_i, \log \sigma_i^2 \in \mathbb{R}^3$ corresponding to $[v, 0, \omega] \in [-1,1]^3$.

In addition to trajectory likelihood optimization, the model is regularized with a collision avoidance objective that penalizes predicted motion commands that would lead to obstacle intersection under the same kinematic integration model used during evaluation. The predicted trajectory consists of 50 action steps executed under a planar unicycle kinematic model with fixed time step $\Delta t = 0.1$ s, corresponding to a 5-second planning horizon. This horizon provides sufficient foresight for obstacle avoidance and goal-directed motion while limiting the uncertainty accumulation commonly observed in longer sequence prediction. Uncertainty-aware trajectory prediction enables probabilistic reasoning and mitigates overconfident motion estimates, particularly in geometrically ambiguous regions.

B. Synthetic Data Generation

1) LiDAR Simulation

LiDAR observations are simulated as temporal sequences rather than single snapshots. Each training sample contains a sequence of 51 LiDAR scans: one scan corresponding to the initial robot state, followed by 50 scans associated with each timestep along the ground truth trajectory.

The LiDAR scans are designed to contain 360 readings uniformly spaced at 1° intervals, providing full 360° coverage and mimicking the behavior of real LiDAR data with a 1° or lower resolution. Distances are computed using geometric ray casting within the 2D environment and normalized to the range $[-1, 1]$.

2) Instruction Generation

Natural language instructions are produced using a template-based generation procedure that introduces variation through randomized selection. This mechanism yields a diversity ratio of 34.2%, corresponding to 1,025 unique instructions generated from a set of 3,000 samples, achieved through systematic recombination of template structures. The instruction generation process employs a library of predefined templates associated with specific action types. For each scene, a templated phrasing is selected at random from the available variants, and simple linguistic modifications such as optional prefixes and suffixes are applied to introduce additional variability. Templates are matched to scene configurations to ensure semantic alignment between the instruction and the corresponding environment.

3) Action Trajectory Generation

Robot actions are represented as continuous trajectories rather than single control vectors. Each trajectory consists of up to 50 timesteps, where each timestep corresponds to a velocity command:

$$[v, 0, \omega] \in [-1,1]^3$$

Trajectories are generated using rule-based heuristics that are consistent with the instruction semantics and the geometric structure of the scene. To support variable-length trajectories, sequences shorter than the maximum length are padded and accompanied by a binary mask indicating valid time steps. Loss computation and evaluation are performed using this mask. Trajectory execution and evaluation assume a 2D unicycle kinematic model with a fixed timestep $\Delta t = 0.1$ s.

C. Multimodal Architecture

Figure 2 presents a high-level overview of the multimodal architecture. The diagram illustrates the four modality-specific encoders—text, vision, LiDAR, and action—that project their respective inputs into a unified 256-dimensional embedding space. The ZSTL architecture integrates specialized encoders for each modality into a unified 256-dimensional embedding space, enabling cross-modal alignment between language, vision, LiDAR, and action representations. The framework consists of four primary encoder modules.

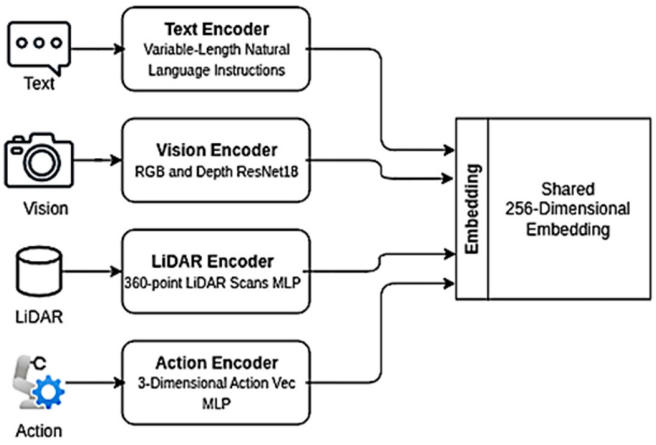

Fig. 2. Proposed multimodal architecture.

1) Text Encoder

In trajectory mode, natural language instructions are encoded using a pretrained BERT model (bert-base-uncased). The contextualized embedding corresponding to the [CLS] token is extracted and projected through a linear layer from 768 to 256 dimensions. The maximum instruction length is fixed to 32 tokens. Sentence-transformer models and OpenAI text embeddings are supported as alternative backends and are primarily used in embedding mode.

2) Vision Encoder

Visual observations are processed using a ResNet18 backbone. The encoder operates in a dual-output configuration. For embedding mode, global average pooling produces a single pooled feature vector. For trajectory mode, the final convolutional feature map of shape (512, 7, 7) is retained. The spatial feature map is reshaped into 49 spatial tokens and projected through a linear layer from 512 to 256 dimensions, yielding a tensor of shape (49, 256). These spatial tokens preserve spatial structure and serve as part of the multimodal context for trajectory decoding.

3) LiDAR Encoder

LiDAR observations are encoded using a temporal sequence encoder. The primary architecture consists of a one-dimensional convolutional network with two convolutional layers, followed by adaptive average pooling and layer normalization. This encoder supports both pooled outputs for contrastive alignment and sequential outputs of shape (51, 256) for trajectory decoding.

An alternative bidirectional LSTM encoder with two layers is supported through a configuration toggle.

4) Action Encoder

Action trajectories are encoded using a lightweight multilayer perceptron. Trajectory tensors of shape $(B,\ 50,\ 3)$ are flattened across the temporal dimension, encoded through a shared MLP, and temporally mean-pooled to produce a fixed-size representation. This encoder is used for contrastive regularization rather than direct trajectory prediction.

5) Multimodal Context Fusion

The multimodal context used for trajectory decoding is constructed by concatenating 49 spatial vision tokens, 51 temporal LiDAR tokens, and one text token. This results in a unified context tensor of shape (101, 256). This fusion strategy preserves spatial and temporal structure across modalities and avoids aggressive pooling prior to decoding.

6) Trajectory Decoder

Trajectory prediction is performed using a Transformer decoder with two layers, eight attention heads, and a model dimension of 256. The decoder operates on a set of 50 learnable query embeddings, with one query corresponding to each trajectory timestep. The multimodal context tensor serves as the decoder memory. Self-attention is applied among trajectory queries, while cross attention allows each query to attend to the multimodal context. Decoder outputs are passed through linear heads to predict the mean and log variance of Gaussian action distributions. Log variance values are clamped to the range $[-10, 2]$ to ensure numerical stability. An LSTM-based autoregressive decoder is supported as an alternative architecture.

D. Advanced Architectural Components

1) Attention Mechanisms

Attention is implemented through the trajectory Transformer decoder. Self-attention is applied among the 50 learnable trajectory queries, while cross-attention is applied from trajectory queries to the 101-token multimodal context memory. No standalone modality-level self-attention, cross-modal attention modules, or explicit temporal attention blocks are used outside the decoder.

2) Normalization and Regularization

Stable optimization and improved generalization are supported through a set of normalization and regularization techniques integrated throughout the architecture. Layer normalization is applied to each encoder output before projection into the shared embedding space to mitigate the internal covariate shift. Dropout with a probability of 0.1 is used during training to reduce overfitting by preventing co-adaptation of hidden units. Embeddings are L2 normalized to unit length, ensuring that cosine similarity provides a well-behaved metric for contrastive alignment. Weight decay is applied uniformly across parameters to encourage simpler, more stable representations. Collectively, these techniques contribute to smoother training dynamics and more robust multimodal embeddings.

3) Embedding Space Analysis

A comprehensive evaluation of the learned embedding space is conducted to assess the quality and structure of cross-modal representations. Pairwise cosine similarity distributions confirm that embeddings preserve meaningful relationships within modalities and across them. Nearest neighbor retrieval experiments demonstrate that the shared latent space supports semantically consistent matching between instructions, observations, and actions. Evaluation focuses on pairwise cosine similarity and recall@K cross-modal retrieval metrics.

E. Training Objective

Trajectory prediction is optimized using a Gaussian negative log likelihood loss:

$$\mathcal{L}_{\text{ta}} = \sum_{i=1}^{50} m_i \cdot \frac{1}{2}\left(\log \sigma_i^2 + \log 2\,\pi + \frac{\|y_i - \mu_i\|_2^2}{\sigma_i^2}\right)$$

where $y_i \in \mathbb{R}^3$ is the ground truth velocity command at timestep i, $(\mu_i, \log \sigma_i^2)$ are predicted Gaussian parameters, and $m_i \in \{0,1\}$ is a padding mask indicating valid timesteps.

1) Collision Loss

To encourage collision-aware motion, a collision avoidance loss is included, computed from the predicted trajectory after kinematic integration under the same unicycle model used for evaluation. Let $\hat{s}_i = (\hat{x}_i, \hat{y}_i, \hat{\theta}_i)$ denote the integrated pose at timestep i. Given an occupancy function $\mathcal{L}_{col\ell} = \frac{1}{50}\sum_{i=1}^{50} I_{occ}(\hat{x}_i, \hat{y}_i)$ indicating whether a point lies inside an obstacle region, the collision loss is defined as:

$$\mathcal{L}_{col\ell} = \frac{1}{50}\sum_{i=1}^{50} I_{occ}(\hat{x}_i, \hat{y}_i)$$

This objective penalizes trajectories that intersect obstacles and can be interpreted as the expected fraction of colliding timesteps.

2) Contrastive Regularization

A multimodal contrastive objective $\mathcal{L}_{\text{contrast}}$ is applied to align embeddings across modalities using a symmetric InfoNCE formulation with temperature τ.

3) Total Loss

The final training objective combines the three components:

$$\mathcal{L} = w_t \mathcal{L}_{\text{ta}} + w_{\text{coll}} \mathcal{L}_{\text{cl}} + w_c \mathcal{L}_{\text{cnrs}}$$

where $w_t = 1.0$, $w_c = 0.1$, and w_{coll} is set according to the training configuration.

F. Advanced Training Techniques

1) Optimization Stability

Training uses standard stabilization mechanisms, including optional gradient clipping (L2 norm 1.0, where enabled) and a cosine annealing learning rate schedule with optional linear warmup. A single learning rate is applied across all parameters.

2) Symmetric Contrastive Loss

The contrastive objective is implemented symmetrically for each selected modality pair (m_1, m_2) as

$$\mathcal{L}_{contrast}^{(m_1,m_2)} = \frac{1}{2}\left(\mathcal{L}_{InfoNCE}^{(m_1 \to m_2)} + \mathcal{L}_{InfoNCE}^{(m_2 \to m_1)}\right)$$

and the total contrastive loss is a weighted sum over selected modality pairs.

III. TESTING AND SIMULATION SETUP

A. Dataset Generation

Synthetic datasets of multiple scales were constructed to evaluate both multimodal embedding alignment and conditional trajectory prediction. The dataset generation pipeline produces samples composed of a rendered depth image, a temporal sequence of LiDAR scans, a natural language instruction, and a continuous action trajectory.

Each trajectory sample contains a depth image of resolution 224×224, a sequence of 51 LiDAR scans (one initial scan followed by 50 scans along the trajectory), a text instruction, and a 50-step action trajectory. Action trajectories are padded or truncated to a maximum length of 50 steps and accompanied by a binary mask indicating valid timesteps.

All datasets maintain a balanced distribution across eleven action categories, including forward motion, turning primitives, stopping behaviors, and simple navigation actions. Instruction lengths range from one to twelve words and follow the distribution induced by the template-based instruction generation process.

B. Training Configuration

Two distinct training configurations were employed for the embedding and trajectory modes, reflecting their differing objectives and computational requirements:

- Embedding mode training used a batch size of 32 and a learning rate of 1×10^{-3}. The embedding dimension was set to 512. Training was performed for 50 epochs using the AdamW optimizer with weight decay 1×10^{-4}. A cosine annealing learning rate schedule with a warmup period of 5 epochs was applied. Gradient clipping with an L2 norm threshold of 1.0 was enabled.
- Trajectory mode training used a batch size of 16 and a learning rate of 1×10^{-4}. The embedding dimension was set to 256. Training was performed for 50 epochs using the AdamW optimizer with weight decay 1×10^{-5}. A cosine annealing learning rate schedule was applied without warmup. No gradient clipping was used.

A collision avoidance loss was included with a weight w_{coll} to penalize predicted trajectories that intersect obstacles under kinematic rollout. The collision loss weight was set to $w_{coll} = 2.1$.

Trajectory prediction was optimized using a Gaussian negative log likelihood loss, combined with a contrastive regularization term. The trajectory loss weight was set to $w_t = 1.0$, and the contrastive loss weight was set to $w_c = 0.1$. The contrastive temperature parameter was fixed at $\tau = 0.07$.

Automatic Mixed Πrecision (AMP) was enabled during trajectory mode training on CUDA-enabled hardware.

C. Training Dynamics and Optimization Analysis

1) Optimization Behavior

Embedding mode training employed batch sizes of 32, while trajectory mode training used batch sizes of 16. Data loading was performed using the standard PyTorch DataLoader with four worker processes. No gradient accumulation was used. AMP was applied during trajectory mode training to reduce memory usage and improve throughput.

2) Convergence Analysis

The trajectory mode training exhibited rapid early convergence. The negative log likelihood decreased from an initial value of 3.47 to approximately 0.10 within the first six epochs, indicating that the model quickly captured coarse motion dynamics from the multimodal context.

Figure 3 illustrates the evolution of the trajectory prediction loss. A steep initial decline is observed, followed by a gradual plateau, suggesting that the decoder efficiently learned the dominant velocity patterns early in training, while later epochs focused on refining motion precision. The absence of oscillatory behavior further indicates stable gradient flow under the Gaussian likelihood objective.

The total training objective follows a similar trajectory. As shown in Figure 4, the loss decreases steadily before reaching its minimum near Epoch 18. Beyond this point, validation performance begins to degrade, providing evidence of overfitting. This behavior suggests that the model capacity is sufficient to fit the synthetic dataset and benefits from early stopping.

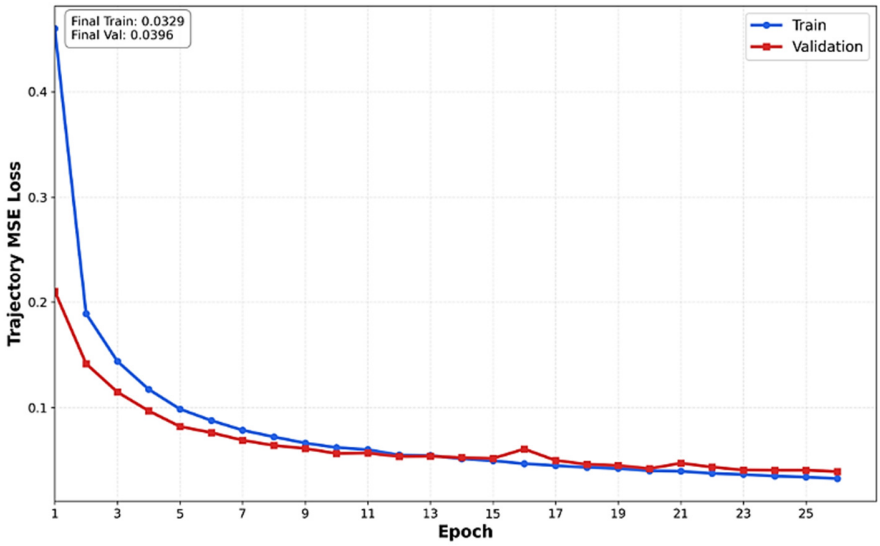

Fig. 3. Trajectory prediction loss (MSE) across training.

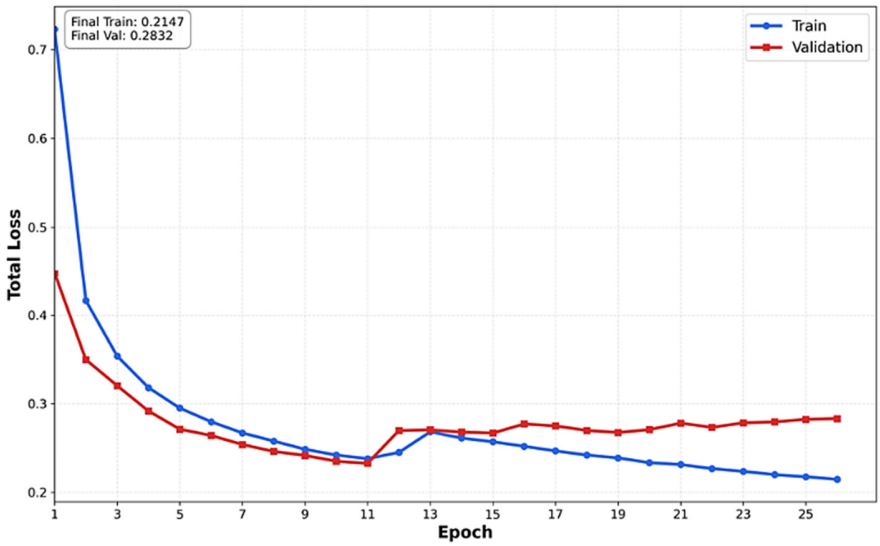

Fig. 4. Total training loss combining trajectory likelihood, collision loss, and contrastive regularization.

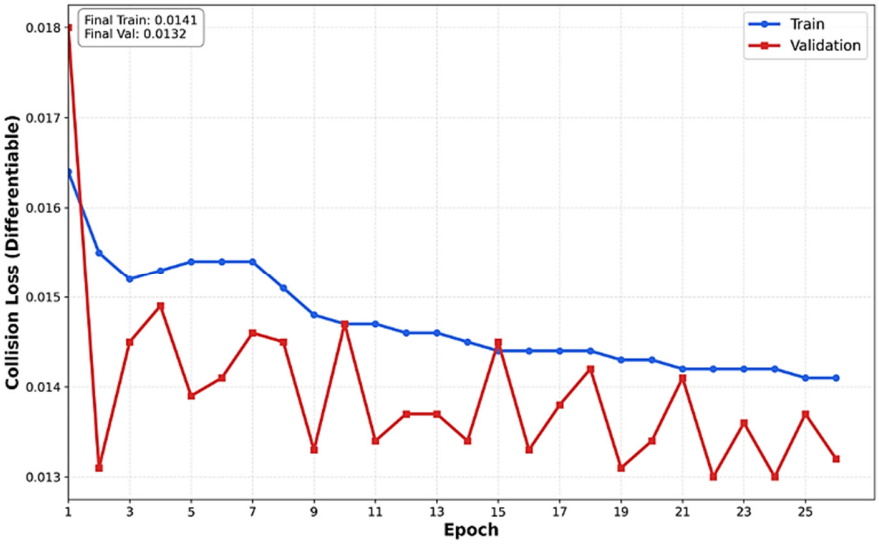

Fig. 5. Collision avoidance loss across epochs.

The collision avoidance loss exhibits a slower but consistently decreasing trend, as shown in Figure 5. Unlike the trajectory loss, which rapidly captures kinematic structure, collision-aware behavior appears to require longer optimization. This is expected, as obstacle avoidance depends on integrating spatial reasoning across modalities rather than learning purely local motion patterns. The collision loss, however, remains bounded throughout training, indicating that the auxiliary safety objective does not destabilize optimization despite being computed through kinematic rollout.

Trajectory prediction accuracy was evaluated using velocity Mean Squared Error (MSE), velocity Mean Absolute Error (MAE), cumulative position error, and final position error. Predicted velocity commands were integrated into planar position trajectories using a unicycle kinematic model with a timestep of

$$x_{t+1} = x_t + v_t \cos\theta_t \, \Delta t$$

$$y_{t+1} = y_t + v_t \sin\theta_t \, \Delta t$$

$$\theta_{t+1} = \theta_t + \omega_t \Delta t$$

where (x_t, y_t) denotes the robot position at time step t, θ_t is the heading orientation, v_t is the linear velocity, ω_t is the angular velocity, and Δt is the fixed integration time step.

3) Validation Protocol

All experiments used a fixed random 70/15/15 train/validation/test split generated through random shuffling.

IV. RESULTS AND DISCUSSION

A. Trajectory Model Performance

The trajectory prediction model was evaluated on a dataset comprising 500 samples. Performance metrics indicate that the model can generate collision-aware trajectories while maintaining spatial accuracy. The average final position error was measured at 9.97 cm, with a collision-free execution rate of 65.9% and an overall task success rate of 59.2%.

Figure 6 shows the distribution of final position errors across the evaluation set. A success threshold of 10 cm was used to compute task completion, explaining the observed success rate. The distribution exhibits a right-skewed tail, indicating that most failures arise from a small number of large terminal deviations rather than systematic drift. The relatively low positional error suggests that the learned policy captures key spatial relationships within the environment, while the collision-free rate indicates partial effectiveness of the collision-aware objective. However, the gap between collision avoidance and task success highlights the inherent difficulty of jointly optimizing safety and goal-directed behavior in trajectory prediction.

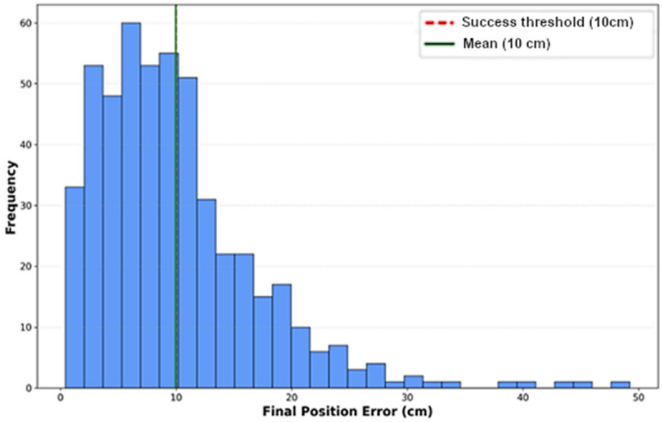

Fig. 6. Distribution of final position errors for the trajectory.

B. Training Behavior and Convergence

Training dynamics reveal stable optimization and effective loss minimization. The final trajectory loss converged to 0.0022, while the collision loss reached 0.0010, indicating that both motion accuracy and safety constraints were successfully incorporated into the learned policy. Convergence occurred at approximately epoch 26, after which additional training

produced diminishing returns. This behavior suggests that the model benefits from moderate training durations and may require stronger regularization to prevent unnecessary parameter updates beyond convergence. GPU-based training achieved a 5.6 × speedup relative to CPU execution, confirming the computational feasibility of the framework for large-scale experimentation.

C. Trajectory Analysis

Qualitative trajectory outputs were analyzed across multiple environments. These visualizations provided insight into the model's spatial reasoning, obstacle avoidance behavior, and goal-oriented motion under varying scene complexity. Figure 7 demonstrates near-perfect trajectory alignment in a narrow corridor scenario. The predicted path remains tightly coupled to the ground truth, yielding a mean positional error of 0.0075m and a final error of 0.0118m.

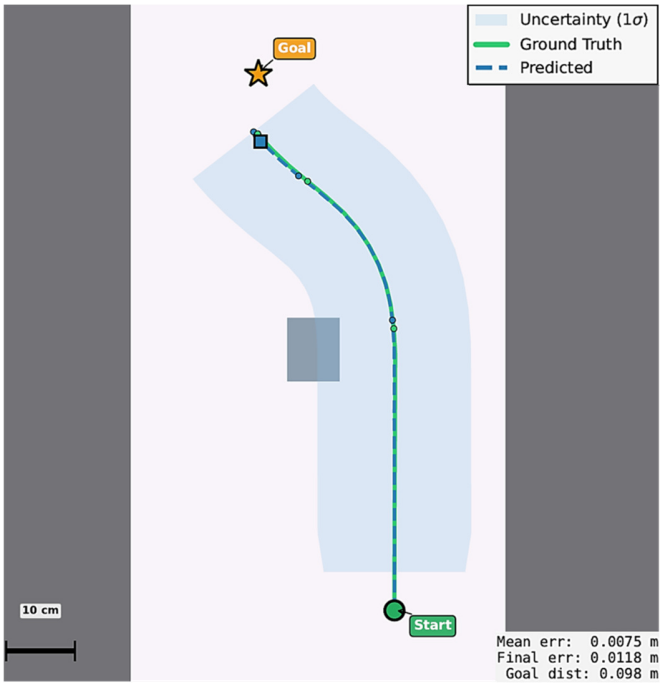

Fig. 7. Trajectory performance in a constrained corridor environment.

The model effectively utilizes spatial tokens when the navigation manifold is well constrained. The absence of unnecessary curvature further indicates stable velocity prediction and low accumulated integration error.

As shown in Figure 8, the model maintains accurate tracking over an extended planning horizon, achieving a mean error of 0.0095 m and a final error of 0.0084 m. The predicted trajectory exhibits smooth curvature and remains fully contained within the feasible corridor. This illustrates how the Transformer decoder successfully integrates multimodal context across time steps, enabling stable long-horizon motion without divergence.

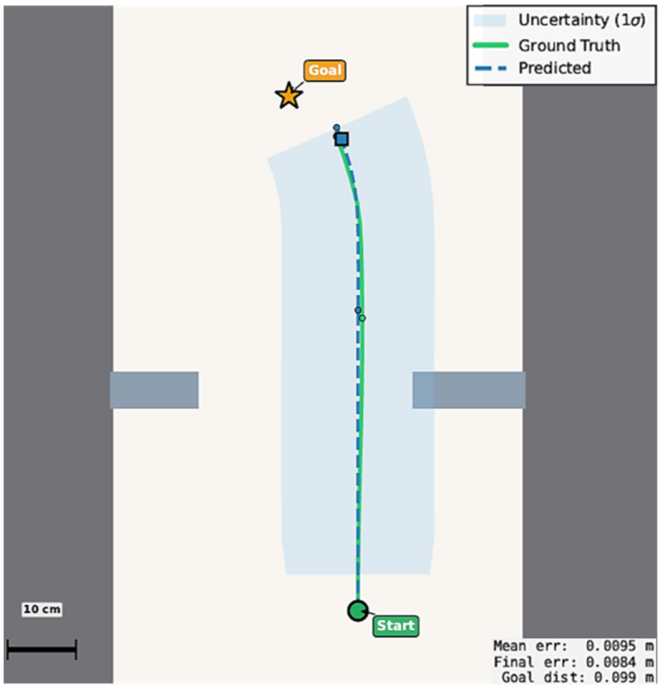

Fig. 8. Long-horizon trajectory prediction.

As illustrated in Figure 9, performance degrades in more geometrically complex environments. The predicted trajectory diverges from the ground truth, resulting in a mean error of 0.1077 m and a final error of 0.1238 m. This deviation likely arises from compounding uncertainty during obstacle negotiation, where small heading errors propagate over time. The example highlights the increased difficulty of multimodal trajectory prediction under dense spatial constraints and suggests that richer scene diversity may be necessary to improve robustness.

Figure 10 shows a trajectory that successfully avoids obstacles but terminates with residual goal displacement. Although the final positional error is relatively small (0.0275 m), the remaining goal distance of 0.307 m suggests incomplete goal convergence. This behavior indicates that while the model captures global navigation structure, fine-grained terminal control remains challenging, particularly when long-horizon planning interacts with obstacle constraints.

Despite the curved navigation corridor shown in Figure 11, the predicted trajectory remains dynamically smooth and reaches close proximity to the goal (0.026 m). The model demonstrates coordinated turning behavior, showing effective coupling between angular velocity prediction and spatial awareness.

These results demonstrate that the proposed architecture supports coherent motion generation across a range of navigation scenarios. Performance is strongest in structured environments and gradually degrades as geometric complexity increases, consistent with the quantitative trends observed earlier.

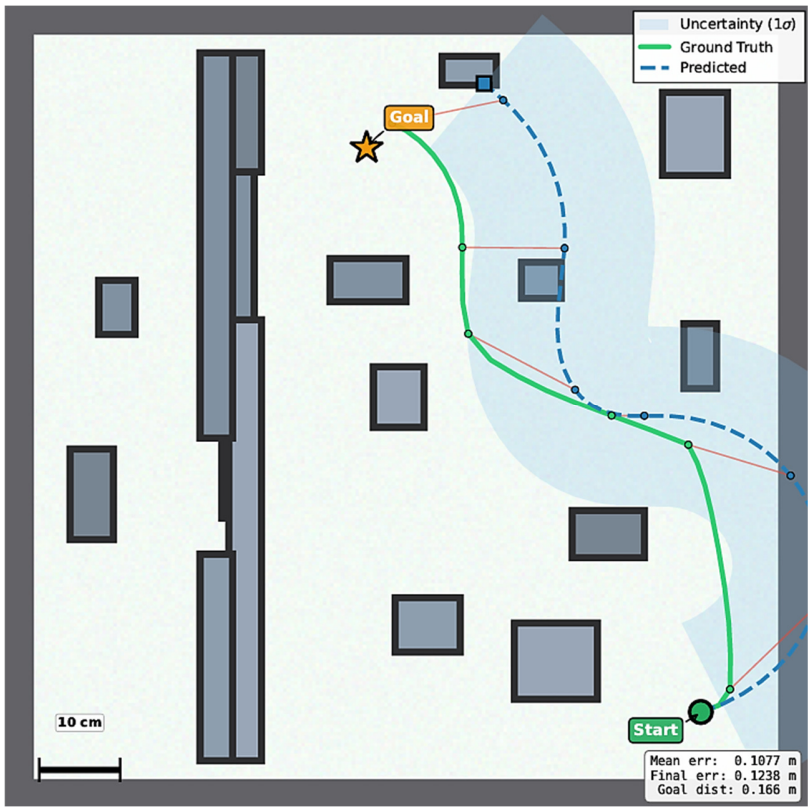

Fig. 9. Trajectory prediction in a cluttered environment.

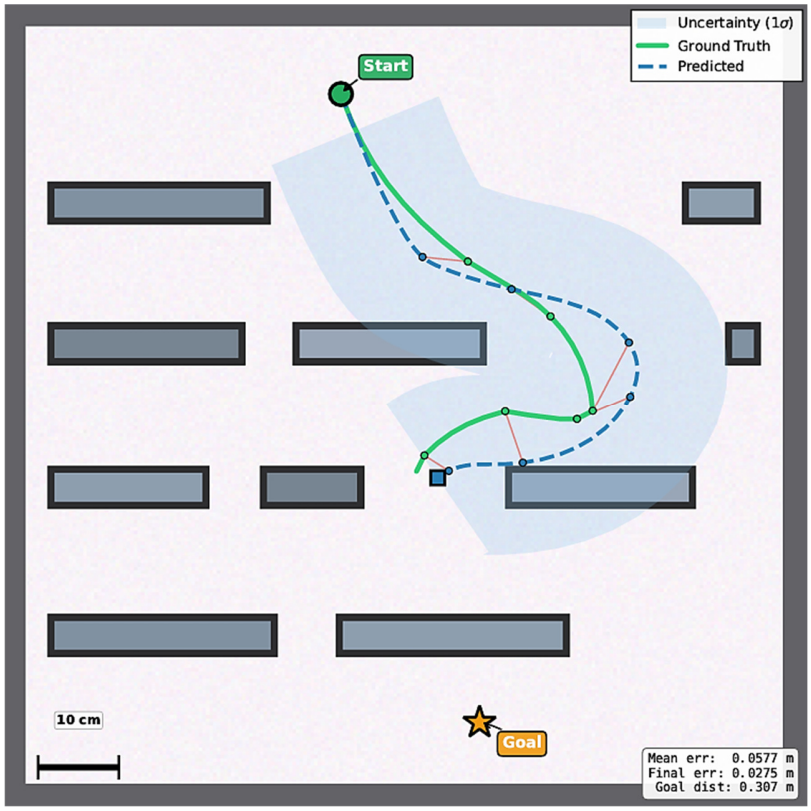

Fig. 10. Trajectory exhibiting slight goal misalignment.

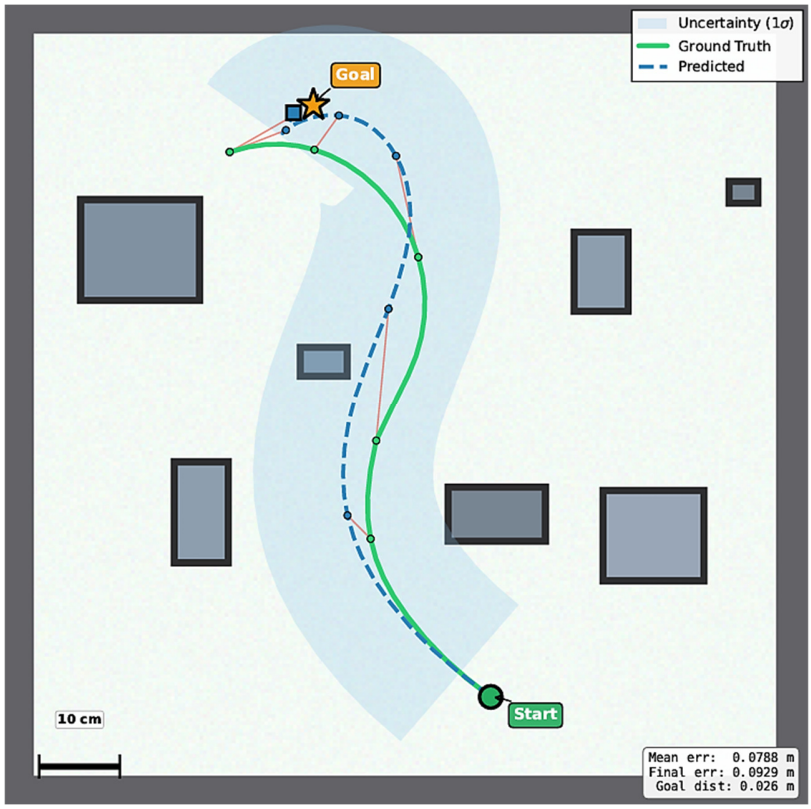

Fig. 11. Curved trajectory with coordinated obstacle avoidance

D. Architectural Implications

The trajectory model contains approximately 125 million parameters and operates on a 101-token multimodal context composed of 49 visual tokens, 51 LiDAR tokens, and a single text token. This rich contextual representation enables the Transformer decoder to integrate spatial, geometric, and linguistic signals when generating motion commands.

E. Multimodal Embedding Framework

The embedding model, comprising approximately 121 million parameters, enables cross-modal semantic alignment and supports retrieval-based reasoning between language, visual observations, and LiDAR signals. Contrastive learning facilitates the emergence of a shared representation space suitable for indexing and semantic search.

F. Key Insights and Implications

Several important observations emerge from the experimental results. First, synthetically generated environments are capable of supporting the training of multimodal robotic policies that produce coherent and executable trajectories. Despite the abstraction of the simulation pipeline, the model demonstrates measurable spatial reasoning behaviors that translate into successful navigation outcomes. Second, multimodal fusion appears to be critical for trajectory quality. Integration of vision, LiDAR, and language enables the system to ground instructions within geometric context, thus reducing ambiguity during action generation and improving trajectory feasibility. Third, uncertainty-aware prediction provides a principled mechanism for modeling motion variability. Although uncertainty estimates were not explicitly leveraged during execution, Gaussian parameterization establishes a foundation for future risk-aware planning strategies and adaptive control. The observed task success rate indicates that the framework captures meaningful relationships between perception and action while leaving room for further refinement. Performance trends suggest that improvements in terminal precision, scene complexity, and data diversity could yield substantial gains in reliability.

Overall, while this evaluation focused on synthetic environments, the architectural design and training paradigm provide a scalable foundation for future extensions toward higher fidelity simulations and real-world robotic platforms.

V. CONCLUSION

This work introduced ZSTL, a multimodal contrastive learning framework designed to illustrate that instruction-conditioned robot representations can be acquired entirely from synthetic data. The results demonstrate that even highly simplified 2D environments can provide sufficient structure for learning meaningful cross-modal associations across language, visual observations, and action representations. The observed text-to-depth similarity scores confirm that the model develops coherent semantic alignment, while the retrieval metrics establish an initial quantitative baseline for future refinement. Additionally, the results indicate that the preservation of spatial and temporal structure prior to decoding contributes positively to the quality of the trajectory. Rather than aggressively

pooling modality features, token-level fusion enables the decoder to selectively attend to relevant environmental cues, supporting more context-aware motion generation. These findings illustrate that synthetically generated datasets constructed from basic geometric primitives and simplified spatial encodings can support early-stage multimodal representation learning. The results further suggest that compact model architectures are capable of capturing the structural relationships required for cross-modal retrieval, positioning synthetic data pipelines as efficient platforms for rapid experimentation in embodied AI research.

Future work should investigate richer synthetic scene generation, physics-based simulation, enhanced linguistic diversity, and more expressive encoder architectures. Continued exploration of these directions will be essential for progressing from preliminary cross-modal alignment toward robust instruction following systems capable of generalization to real-world robotic settings.

DECLARATIONS OF COMPETING INTERESTS

The authors declare no competing interests that could have influenced the results of this study.

ACKNOWLEDGMENT

The authors thank the open-source community for providing essential tools and libraries that enabled this research.

DATA AVAILABILITY

The code developed for this study is available from the corresponding author upon request. The training data used in this work were synthetically generated using the provided codebase, and the corresponding data generation procedures are included therein. In addition, the architecture and structure of the proposed model are described in detail in the methodology section. Additional information on implementation and usage can be provided upon request.

REFERENCES

[1] M. Shridhar *et al.*, "ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020, pp. 10737–10746, https://doi.org/10.1109/CVPR42600.2020.01075.

[2] O. Kroemer, S. Niekum, and G. Konidaris, "A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms," *Journal of Machine Learning Research*, vol. 22, no. 30, pp. 1–82, 2021.

[3] M. Soori, B. Arezoo, and R. Dastres, "Artificial intelligence, machine learning and deep learning in advanced robotics, a review," *Cognitive Robotics*, vol. 3, pp. 54–70, Jan. 2023, https://doi.org/10.1016/j.cogr.2023.04.001.

[4] H. L. Nguyen, D. T. Le, and H. H. Hoang, "Application of Synthetic Data on Object Detection Tasks," *Engineering, Technology & Applied Science Research*, vol. 14, no. 4, pp. 15695–15699, Aug. 2024, https://doi.org/10.48084/etasr.7929.

[5] M. Shridhar, L. Manuelli, and D. Fox, "CLIPort: What and Where Pathways for Robotic Manipulation," in *Proceedings of the 5th Conference on Robot Learning*, Jan. 2022, pp. 894–906.

[6] J. Li, R. Selvaraju, A. Gotmare, S. Joty, C. Xiong, and S. C. H. Hoi, "Align before Fuse: Vision and Language Representation Learning with Momentum Distillation," in *Advances in Neural Information Processing Systems*, 2021, vol. 34, pp. 9694–9705.

[7] J. Li, D. Li, C. Xiong, and S. Hoi, "BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation," in *Proceedings of the 39th International Conference on Machine Learning*, June 2022, pp. 12888–12900.

[8] R. Garcia, S. Chen, and C. Schmid, "Towards Generalizable Vision-Language Robotic Manipulation: A Benchmark and LLM-Guided 3D Policy," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, May 2025, pp. 8996–9002, https://doi.org/10.1109/ICRA55743.2025.11127315.

[9] S. S. Kannan, V. L. N. Venkatesh, and B. C. Min, "SMART-LLM: Smart Multi-Agent Robot Task Planning using Large Language Models," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2024, pp. 12140–12147, https://doi.org/10.1109/IROS58592.2024.10802322.

[10] D. Driess *et al.*, "PaLM-E: An Embodied Multimodal Language Model." arXiv, Mar. 06, 2023, https://doi.org/10.48550/arXiv.2303.03378.

[11] X. Han *et al.*, "Multimodal fusion and vision–language models: A survey for robot vision," *Information Fusion*, vol. 126, Feb. 2026, Art. no. 103652, https://doi.org/10.1016/j.inffus.2025.103652.

[12] J. Urain *et al.*, "A Survey on Deep Generative Models for Robot Learning From Multimodal Demonstrations," *IEEE Transactions on Robotics*, vol. 42, pp. 60–79, 2026, https://doi.org/10.1109/TRO.2025.3631816.

[13] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent Advances in Robot Learning from Demonstration," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. Volume 3, 2020, pp. 297–330, May 2020, https://doi.org/10.1146/annurev-control-100819-063206.

[14] F. Yuan, E. Klavon, Z. Liu, R. P. Lopez, and X. Zhao, "A Systematic Review of Robotic Rehabilitation for Cognitive Training," *Frontiers in Robotics and AI*, vol. 8, May 2021, https://doi.org/10.3389/frobt.2021.605715.

[15] T. N. Huynh and K. D. Nguyen, "Integrative AI framework for robotics: LLM-enabled reinforcement learning in object manipulation and task planning," *Robotics and Autonomous Systems*, vol. 195, Jan. 2026, Art. no. 105197, https://doi.org/10.1016/j.robot.2025.105197.

[16] C. M. Huang and B. Mutlu, "Learning-based modeling of multimodal behaviors for humanlike robots," in *Proceedings of the 2014 ACM/IEEE International Conference on Human-robot Interaction*, Nov. 2014, pp. 57–64, https://doi.org/10.1145/2559636.2559668.