2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

LATTE: LAnguage Trajectory TransformEr

Arthur Bucker[1], Luis Figueredo[1], Sami Haddadin[1],
Ashish Kapoor[2], Shuang Ma[2], Sai Vemprala[2], Rogerio Bonatti[2]

[1]Technische Universität München, [2]Microsoft

***Abstract*— Natural language is one of the most intuitive ways to express human intent. However, translating instructions and commands towards robotic motion generation and deployment in the real world is far from being an easy task. The challenge of combining a robot's inherent low-level geometric and kinodynamic constraints with a human's high-level semantic instructions traditionally is solved using task-specific solutions with little generalizability between hardware platforms, often with the use of static sets of target actions and commands. This work instead proposes a flexible language-based framework that allows a user to modify generic robotic trajectories. Our method leverages pre-trained language models (BERT and CLIP) to encode the user's intent and target objects directly from a free-form text input and scene images, fuses geometrical features generated by a transformer encoder network, and finally outputs trajectories using a transformer decoder, without the need of priors related to the task or robot information. We significantly extend our own previous work presented in [1] by expanding the trajectory parametrization space to 3D and velocity as opposed to just XY movements. In addition, we now train the model to use actual images of the objects in the scene for context (as opposed to textual descriptions), and we evaluate the system in a diverse set of scenarios beyond manipulation, such as aerial and legged robots. Our simulated and real-life experiments demonstrate that our transformer model can successfully follow human intent, modifying the shape and speed of trajectories within multiple environments. Codebase available at: `https://github.com/arthurfenderbucker/LaTTe-Language-Trajectory-TransformEr.git`.**

I. Introduction

Robots are increasingly working in proximity to humans, sharing living and working spaces. Within this context, it is of high importance for the robotics community to research techniques that allow autonomous agents to seamlessly interact with human users. This work focuses on one important facet of human-robot interaction: given a user's objective and an obstacle environment, how can the robot best generate a trajectory that respects the human preferences while tending to safety and dynamics constraints in its surroundings?

Robots of today are still largely pre-programmed for specific tasks, and have very limited capability to operate and adapt to new contexts among unstructured human-centered environments. Ideally, in such scenarios the robot should have the ability to recognize and understand natural language commands in a given context and map them to the task-domain space – where tasks and constraints are largely influenced by context, intent and affordances with objects [2]. This paradigm shift deviates from traditional motion planning, and requires methodologies that are able to integrate multi-modal inputs coming from perception systems (for instance user-provided language commands and robot vision) together with geometrical information to shape robot trajectories towards the desired human intent. Figure 1 displays a typical application scenario for our trajectory adaptation method.

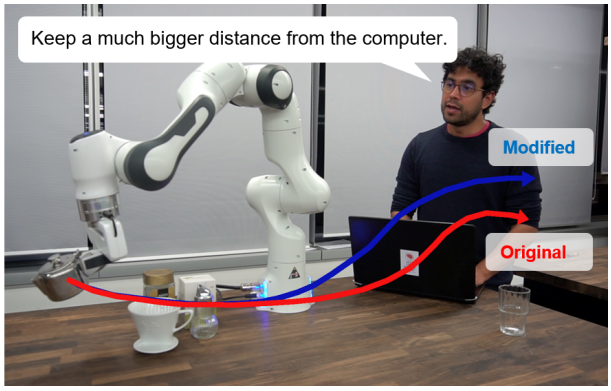

Fig. 1: Trajectory reshaping obeying user's constraints. Our method fuses natural language commands, images of the environment, and geometrical data to generate the modified robot's trajectory.

The core of our method lies within natural language understanding, which is the most intuitive way for a user to express their intent. While large pre-trained large language models (LLMs) such as BERT [3], GPT3 [4] and Megatron-Turing [5] have revolutionized our ability to perform linguistic tasks in recent years, we have just started to see pioneering works that incorporate large foundation language models with robotics tasks [1, 6]–[9]. The use of pre-trained LLMs is extremely beneficial within the robotics context because human-provided annotations are scarce and often costly to obtain. The challenge which we explore in this paper then becomes how we can exploit these rich semantic representations and align them with geometrical trajectory data when mapping commands towards trajectory waypoints.

In this work we propose a framework that allows a user to reshape a trajectory using language instructions. Our method uses a initialization from any geometrical planner (*e.g.* A*, RRT* [10], MPC [11]), which are concerned solely about obstacle avoidance and dynamics constraints, and augments it with semantic objectives. This paper serves as an extension of our previous work in this domain [1], but with significant improvements in the architecture and experimental evaluations:

- **Trajectory dimensionality:** we expand the dimension of each trajectory waypoint from planar (XY) to 3D and velocity in this work;
- **Environment images:** while the original paper used textual object labels (*e.g.* 'Hammer', 'Bottle') as input to the network, here we use images of objects when inferring targets for the user's commands, which is a

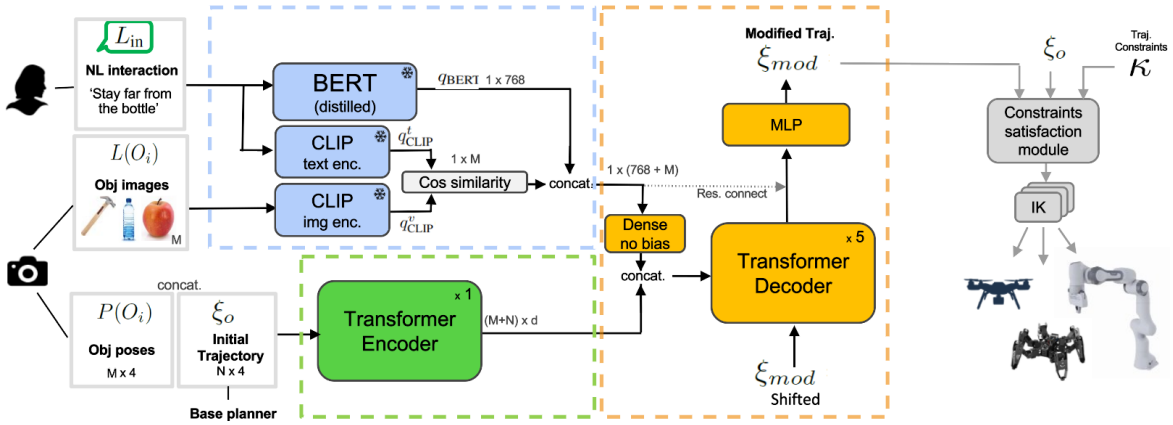

Fig. 2: Systems architecture: in blue, the language and contextual encoding module, compose mainly of frozen pre-trained models. In green the geometrical encoding . In orange the multimodal tranformer decoder.

more realistic setting;

- **Multi-platform evaluation:** We expand the experimental evaluation towards multiple robotics form factors beyond manipulators. We show that the model's outputs are amenable to different robot dynamics and motion controller in aerial and legged locomotion domains.

II. Related Work

Natural language and robotics: Equipping robots with natural language models provides an intuitive and straightforward interface to address these challenges through human interaction and decision-making. Classically, modeling human-robot interactions using language is challenging because it forces the user to operate within a rigid set of instructions [12], or requires mathematically complex algorithms to keep track of multiple probability distributions over actions and target objects [13, 14]. There has been an increase in recent works that explore the use of deep models to implicitly keep track of the complex mapping between language and actions, but the downside is that they often require vast amounts of data for training [15]–[18].

In the domain of navigation we find literature that investigates the use of multi-modal representations fusing natural language and perception along with planning modules through the use of cost functions or reinforcement learning [7, 16, 19]–[24]. In the manipulation domain we also find the work of [7], which uses CLIP [25] embeddings to combine semantic and spatial information. To this end, it can be often beneficial to use pre-trained multi-modal representations that align visual and language inputs representation such as [26]–[29], which often using BERT-style [3] training procedures. Representations are often fine-tuned [30]–[32] on the deployment scenario.

Transformers for robotics: Transformers, originally introduced in the language processing [33], quickly proved to be useful in modeling long-range data dependencies in other domains. Within the robotics motion planning context, transformers architectures have been directly used for trajectory forecasting [34] and reinforcement learning [35, 36]. A more common use of transformers in robotics has been as feature extraction modules for one or more modalities simultaneously that leverage large-scale pre-trained models [1, 6]–[9].

Particularly close to this paper is the work of [9]. It uses pre-trained LLMs to create a semantic cost map that guides a optimization-based motion planner to produce trajectories that satisfy motion constraints provided by a user in free-form text. Similarly, our method also uses LLMs for textual and visual feature extraction, however we use a transformer encoder-decoder pair to align semantic information with geometric cues to recast trajectories. Our first paper presented in [1] validated our approach for 2D scenarios, and showed its effectiveness compared to other interfaces for human-robot interaction. As described at the end of section I this paper extends these ideas to higher dimensions and more realistic experimental settings.

III. Approach

Our overall goal is to provide a flexible interface for human-robot interaction within the context of trajectory reshaping that is agnostic to robotic platforms. The user provides a natural language command, and the robot's body or end-effector behavior, which is expressed with a 3D trajectory over time, is expected to be modified accordingly. Our trajectory generation system uses a sequential waypoint prediction model that takes into account multiple data modalities from scene geometry, environment images and the language input, all of which are fed into a transformer encoder-decoder pair.

Beyond the user's semantic intent, we expect the final trajectory to also respect safety and dynamics space-state constraints, which can be achieved by post-processing the model's output into a continuous state space. This last stage allows our same model to be employed by different robot form factors by using the proper inverse kinematics modules.

A. Problem Definition

Let $\xi_o : [-1, 1] \rightarrow \mathbb{R}^4$ be the original normalized robot trajectory which is composed by a collection of N waypoints and associated velocities $\xi_o = \{(x_1, y_1, z_1, v_1), ..., (x_N, y_N, z_N, v_N))\}$, where x_i, y_i, z_i and

v_i are the waypoint coordinates and the velocity at time step i, respectively. We assume that the original trajectory obeys the system constraints and can be pre-calculated using any desired motion planning algorithm, but falls short of the full task specifications. Let L_{in} be the user's natural language input sent to correct the original trajectory, such as L_{in} = "Go slower when next to the fragile glasses".

Let $\mathcal{O} = \{O_1, ..., O_M\}$ be a collection of M objects in the environment, each with a corresponding position $P(O_i) \in \mathbb{R}^3$ and image $I(O_i)$. Our goal is to learn a function f that maps the original trajectory, user command and obstacles towards a modified trajectory ξ_{mod}, which obeys the user's semantic objectives and is contained in the system feasible domain K:

$$\xi_{mod} = f(\xi_o, L_{\text{in}}, \mathcal{O}) \tag{1}$$

B. Proposed Network Architecture

We approximate function f from (1) by a parametrized model f_θ, learned directly in a data-driven manner. This mapping is non-trivial since it combines data from multiple distinct modalities, and also contains ambiguities in solution space since there are multiple trajectories that satisfy the user's semantic objective.

Our model architecture is divided into 3 main modules and one constraint satisfaction step. Fig. 2 shows the connection between this modules. First, a language and image encoder makes use of distinct pre-trained feature encoders (BERT and CLIP) to generate a embedded representation of the natural language input and to identify the possible objects referred to in the text. Next, a geometry encoder uses object poses and trajectory waypoints as inputs and uses a transformer to learn geometric relations between the original trajectory, speed profiles and the objects in the scene. Finally, a multi-modal transformer decoder combines the embedded outputs of the two prior modules to generate the modified trajectory autoregressively. We discuss each module in detail below:

Language and image encoder: The use of a large language model creates more flexibility in the natural language interface, allowing the use of synonyms (shown in Section IV-B) and less training data, given that the encoder has already been trained with a massive corpus. We use a pre-trained BERT encoder [3], to produce semantic feature $q_{\text{BERT}}(z|L_{\text{in}})$ from the user's input. In addition, we use the pre-trained text and image encoders from CLIP [25] to extract latent embeddings from both the user's text $q^t_{\text{CLIP}}(z|L_{in})$ and the M object images $q^v_{\text{CLIP}}(z|I(O))$. We compute the cosine similarity vector s between the visual and textual embeddings in order to identify a possible target object for the user's command. In section IV-B we show that using the object's images for target identification brings equivalent results as our previous work [1] with object textual descriptions, since CLIP maps both modalities to a joint latent space. Finally, we concatenate the similarity vector s and the semantic features $q_{\text{BERT}}(z|L_{\text{in}})$ forming what we call semantic embedding q_{S}.

Geometry encoder: The original trajectory ξ_o is composed of points that are low-dimensional tuples $(x_i, y_i, z_i, v_i) \in \mathbb{R}^4$. In order to extract more meaningful information from each waypoint, we follow the example of [34] and apply a linear transform with learnable weights W_{geo} that projects each of these points into a higher dimensional feature space. The poses $P(O_i)$ of each object are also processed with the same linear transform, and padded with zeros for the velocity component.

We then concatenate the sequences of high-dimensional feature vectors from waypoints and objects and use a transformer-based feature encoder T_{enc} to extract geometrical features for each element. The use of a Transformer model is preferred for sequences over recurrent networks because its architecture can intrinsically attend to multiple time steps simultaneously. Conversely, recurrent networks suffer from vanishing gradient issues [34], which negatively affect feature extraction and training stability.

Multi-modal transformer decoder: Feature embeddings from both language and geometry are combined as input to a multi-modal transformer decoder block T_{dec}. This block generates the reshaped trajectory ξ_{mod} sequentially, feeding the last token prediction as input to the next waypoint prediction. This procedure is analogous to common transformer-based approaches for language translation [4, 33], but in this case we can image that our model *translates* trajectories from the original feature space towards a new space that obeys the user's semantic constraints. We use imitation learning to train the model, and employ the Huber loss [37] between the predicted and ground-truth waypoints.

C. Post-processing and execution

Once a trajectory is generated by our model it needs to be post-processed to allow for the robot's execution. The modules described here allow our method to be agnostic to specific robotics platforms.

Constraint satisfaction: Constraint satisfaction is a complex and open field of study in robotics. In this work we establish two simplifying assumptions regarding our deployment objectives. First, the base motion planner outputs a set of hard constraints K defined in the Cartesian space that define an admissible region for the trajectories. Second, we assume that the original trajectory is already within in the allowable constraint set. We post-process our model's output trajectory by taking steps starting at the original waypoint towards the direction of new one: $\xi(t) = \xi_o(t) + \alpha(\xi_{mod}(t) - \xi_o(t))$, where $0 < \alpha \leq 1$. If at any step we find that one waypoint reaches an inadmissible region then its position is not further updated. We note that more complex constraint satisfaction algorithms can be developed here, but the simple approached described worked well with our scenarios.

Inverse kinematics: Once the final trajectory is obtained, the user may plug in any inverse kinematics algorithm to obtain final trajectories for higher-dimensional degree of freedom robots. In this work we evaluate our system with manipulators, aerial and legged robots.

D. Synthetic Data Generation

Data collection in the robotics domain can be challenging and expensive, specially when we require alignment between

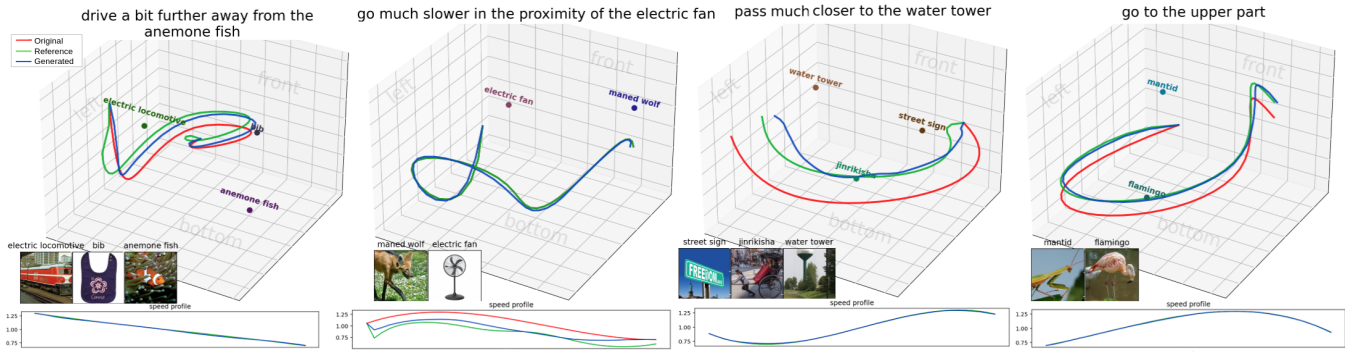

Fig. 3: Procedural dataset examples showing the original trajectory (red), ground-truth modifications, and model predictions (blue). Images representing objects are crawled from the web (bottom left), and the speed profile can also be modified (bottom right).

multiple modalities such as language, vision, and geometry. We find different strategies in the robotics literature to deal with these issues, ranging from costly large-scale online user studies for language labeling [38, 39] all the way to procedural data generation using heuristics [17]. Our work relies on purely procedural generation of trajectory-language pairs. We make a key hypothesis that the use of large-scale language models for feature encoding (q_{BERT}, q_{CLIP}) reduces the data requirements in terms of vocabulary diversity. We assume that if we are able to procedurally generate a small but meaningful set of examples with semantically-driven trajectory modifications we can train an effective transformer decoder, given that the BERT and CLIP encoders have already been trained with large corpuses and are able to handle vocabulary and sentence variations. These assumptions are validated experimentally in Section IV-D.

Each data sample is composed of a base trajectory ξ_0, a natural language input L_{in}, a modified trajectory ξ_{mod}, and a set of object $\mathcal{O} = \{O_1, ..., O_M\}$ represented as central poses $P(O)$ and images $I(O)$. ξ_0 is generated by fitting a spline in the Cartesian space through points generated in a random walk. Objects poses are then randomly generated in space, and we sample object names from the Imagenet dataset [40] as their labels, and obtain various images for each one using a crawler over Bing Images using the object name as the web query.

As for the language input L_{in}, we focus on three main trajectory modifications: i) changes in the absolute Cartesian trajectory space (*e.g.* "stay on the left", "go more to the right"), ii) changes in speed (e.g. "go faster", "go slower when next to x"), and iii) positional changes relative to objects (e.g. "walk closer to x", "drive further away from x"). We pick a sample from a vocabulary bank associate each modification type, and calculate a force vector field over the enviornment using a handcrafted function $F(L_{in}, P(O))$. The field strength may vary depending on additional intensifier words that can be added to the sentences such as "very", "a bit", etc. In the section IV-B we also explore augmenting these language inputs using BART [41], which is a pre-trained paraphrasing model. Finally, we generate the ground-truth trajectory modification by iteratively optimizing the original trajectory along the vector field.

We introduce one additional hyper-parameter in the dataset generation and model training which we name *locality factor*. For the same language prompt, some robotics contexts might require small localized trajectory changes while others might expect long-range modifications. After training, the locality factor allows the user to define their desired range of model influence.

IV. Experiments

We conducted several simulated and real-world experiments to validate our methods. Our main goals were to: i) measure the effectiveness of our trajectory modification algorithm in 3D and velocity space, ii) understand the influence of the different architectural components towards the model's success, and iii) validate the applicability of the model to multiple robotic platforms.

A. Model training details

We trained and evaluated the model described in Section III over a dataset containing 100k examples of procedurally generated trajectory modification. Among these, we used 70k samples for training, 10k for validation and 20k for testing. We kept both BERT and CLIP encoder weights frozen in other to avoid biasing the models towards our vocabulary, with $q_{\text{BERT}}(z|L_{\text{in}}) \in \mathbb{R}^{768}$ and $q^v_{\text{CLIP}}(z|I(O)) \in \mathbb{R}^{512}$. We upscale the dimensionality of each scene object pose from $4 \rightarrow 400$ (depth) using a learned linear matrix, and apply the same procedure to 40 waypoints from the original trajectory ξ_o. T_{enc} is a 1-block transformer encoder, and T_{dec} is a 5-block transformer. Each transformer has 3 hidden layers with 512 fully-connected neurons with Relu activations,one Layer Normalization, 8 attention heads. We use the AdamW [42] optimizer with an initial learning rate $\gamma = 1e-4$, a linear warm-up period of 15 epochs and a learning rate decay of 10% after a plateau of 10 epochs on the validation loss. We use a Nvidia Tesla V100 GPU with batch size of 16, and train the model for 500 epochs in approximately 2 hours.

B. Simulation Experiments

We apply our method to several simulated scenarios. First, we show the basic workings of our trajectory adaptation method through qualitative results which can be visualized in Figure 3. In this scenario, we use sample objects that were randomly chosen from crawling the web and their corresponding images. Assuming there is an initial trajectory that

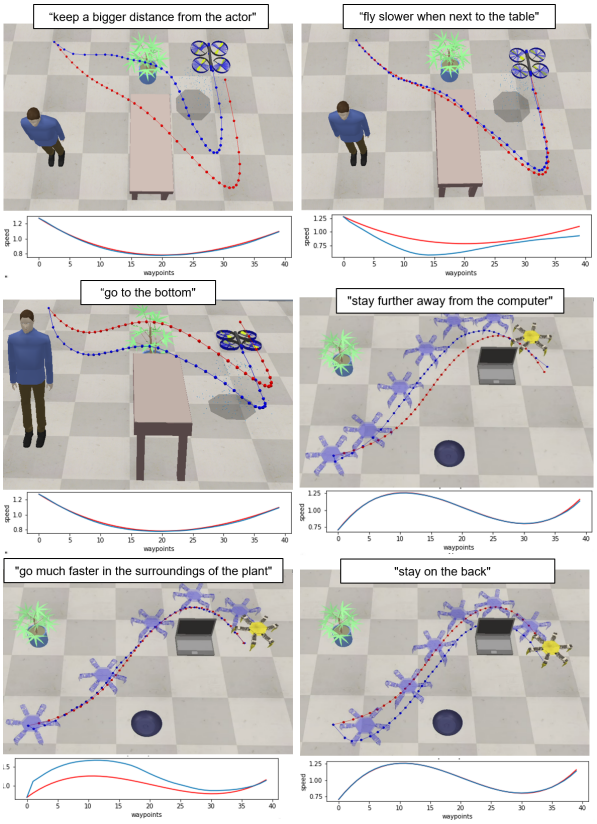

Fig. 4: Model deployed with different robot form factors (drone and legged hexapod) for obstacle avoidance, speed refinement and absolute cartesian changes. Original trajectory shown in red, modification in blue, and corresponding speed profiles below each scenario.

traverses around these objects, and given language commands indicating how to modify the trajectory (farther/closer to the object, faster/slower in the vicinity of an object), our model predicts trajectories that account for user intent. We show both spatial modifications as well as changes in speed profile in the trajectories output by our model.

Multi-platform evaluation: To validate our framework's ability to adapt to different robot dynamics and environments we designed simulated environments using the CoppeliaSim simulator with Bullet physical engine [43]. While our original training dataset presents itself in a format amenable to end-effector positions within a manipulation context, this new simulator allows us to test our system on distinct robotic platforms, dynamics and base motion controllers.

Specifically, we employ an aerial vehicle and a legged hexapod platform. The drone operates within a 3D global frame of reference and uses PID motion controller for trajectory tracking. In contrast, the hexapod is constrained to 2D movements and uses an open-loop motion controller. As figure 4 shows, our approach can successfully modify the base trajectories (red) for different types of natural language inputs. Additional experiments can be seen in the video attachment.

Baseline architectures: We compare our proposed multi-modal transformer against architecture variations. Table I shows the result of a grid search over the number of layers and encoding dimension (depth) of the transformer encoder and decoders. The model with one encode layer, 5 decoder layer and an depth of 400 was chosen to be the reference model for our architecture and further baseline comparisons. We measure performance in terms the similarity between our model's output and the ground-truth trajectory modification in the dataset. Our metrics are MSE (mean squared error), MAE (mean absolute error), DTW (dynamic time warping), and DFD (discrete Frechet distance).

n.enc	n.dec	n.depth	param.	MSE↓	MAE↓	DTW↓	DFD↓
2	3	256	4.95M	0.00306	0.0314	3.1085	0.1346
2	3	400	9.28M	0.00235	0.0273	2.6966	0.1198
2	5	256	6.53M	0.00280	0.0284	2.8455	0.1265
2	5	400	12.7M	0.00238	0.0231	2.4900	0.1152
1	3	256	4.42M	0.00274	0.0272	2.8122	0.1245
1	3	400	8.22M	**0.00224**	**0.0229**	**2.4445**	**0.1130**
1	5	256	6.00M	0.00277	0.0264	2.7527	0.1238
1	5	400	11.2M	0.00234	0.0227	2.4699	0.1138

TABLE I: Architecture variations

Table I provides valuable findings regarding the model architecture. For instance, increasing the number of encoder blocks caused no improvement on the model's performance. Furthermore the model with 3 decoder blocks presented slightly better results than the assumed baseline of 5 decoder block.

In addition to model size, in table II we compare different architecture structures. The *Naive* approach simply copies the original trajectory. The *No NL input* baseline represents a universal prior of the dataset, with an empty language command. *Ours light* is a more compact version of our model with 1 enc., 3 dec. and depth of 256.

Approach	Param.	MSE↓	MAE↓	DTW↓	DFD↓
Naive	-	0.00437	0.02709	3.568	0.1387
No NL input	11.2M	0.04193	0.1663	15.097	0.5674
Ours light	4,42M	0,00274	0,0272	2,8122	0,1245
Ours	11.2M	**0,00234**	**0,02273**	**2,4699**	**0,1138**

TABLE II: Baseline architecture comparisons

Locality factor: Fig. 5 shows the response of our model for different values of the locality factor (LF). This hyper-parameter provides useful information on the range of the desired change over the trajectory, which can serve as a finer user control besides the language input itself.

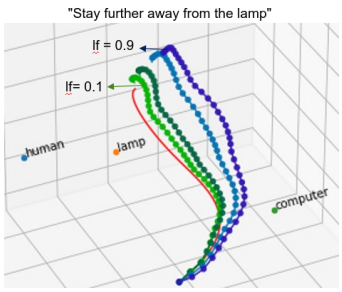

Fig. 5: Locality factor influence

Dataset size and augmentations: Table III shows the effect of increasing the training dataset size in model performance, as well as the effect of applying augmentations in the training data. An increase in the dataset size from 1k to 10k samples significantly improves the validation metrics with minimal challenges besides a longer training time, given that data can be generated procedurally without expensive human annotations. The geometrical augmentation (randomly shifting and scaling operations) shows a modest increase in

performance.

Dataset size	Without geometrical augmentation MSE↓	MAE↓	DTW↓	DFD↓
1k	0.02608	0.11063	8.20700	0.46488
10k	0.00243	0.02347	2.47016	0.11683
100k	0.00229	0.02201	2.39301	0.11175
Dataset size	With geometrical augmentation MSE↓	MAE↓	DTW↓	DFD↓
1k	0.01420	0.07590	5.35290	0.35737
10k	0.00248	0.02324	2.50841	0.11593
100k	0.00234	0.02273	2.46992	0.11383

TABLE III: Effect of dataset size and geometrical augmentation.

C. Real Robot Experiments with Manipulation

We deployed our model in real-world experiments using a 7-DOF PANDA Arm robot equipped with a claw gripper. An off-the-shelf CPU/GPU setup computes the arm's low-level controller and our model. A camera mounted on the workbench captures images of the obstacle setting, and a YOLOV3[44] object detector extracts bounding boxes of the five most likely objects to be sent to the CLIP encoder. Snapshots of the setup and results can be found in figures 1 and 6. Additional experiments shown in the video attachment.

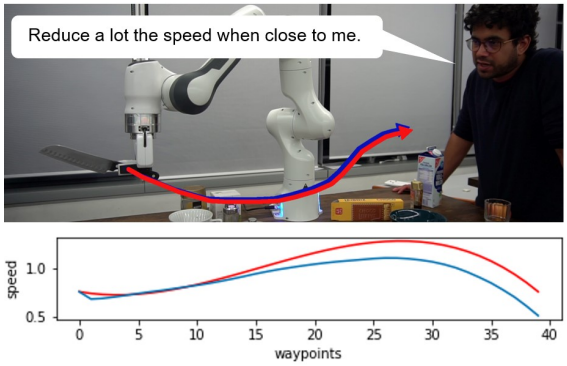

Fig. 6: Real life setup and sample interaction. It depicts the speed modification through online language instructions. An approximated representation of the original trajectory is shown in red, while the modified one in blue. Full videos in the supplementary material.

D. User study experiments

We evaluated the model's performance against baseline architectures in a user study, collecting in total 300 data-points from 10 participants. Each user was asked to evaluate within a 1-5 Likert scale the trajectory changes generated from 5 different approaches considering a given NL interaction. Figure 7 summarizes the distribution of answers for each baseline. *"Ground Truth"* represents the procedural dataset used for training. As the chart shows, most users considered that our trajectory modifications in the dataset correctly represented the language commands. A similar pattern emerged from our trained model (*"Ours"*), which yielded high-quality ratings. The *"Ground Fake"* approach shows samples of the dataset with intentionally wrong modifications, opposite to the ground truth, for the means of comparison. Non surprisingly it is rated with the lowest score. The *"No language"* baseline was also badly evaluated, showing that the model's performance is highly dependent on the language input, and that the model does not memorizes bias purely based on the scene context. Finally, the *"Projected 2D"* distribution shows a direct comparison with our previous work [1], which produces pure 2D trajectory modifications. Its bad performance motivates the importance of the additions of 3D and velocity space that we incorporate in this paper.

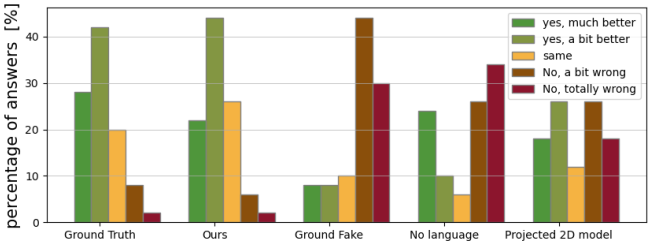

Fig. 7: Userstudy distributions of answer for each baseline.

After the initial evaluations, each user was asked to freely interact with 5 trajectories using a text box, and next judge the quality of the generated modifications. 48% of the user inputs presented words never seen by the model during the training process (out of distribution). Even under these challenging conditions the model only failed on 24% of the cases. Table IV compares our model's performance for in and out of distribution settings.

Textual interaction	Better [%]	Same [%]	Worse [%]
In-dataset vocabulary	66.0	26.0	8.0
Free user input	46.0	30.0	24.0

TABLE IV: Evaluation of out of distribution NL interactions

V. Conclusion and Discussion

This work develops a flexible language-based human-robot interface that allows a user to modify existing robotic trajectories. Our method leverages pre-trained large language and image models (BERT and CLIP) to encode the user's intent and target objects directly from a free- form text input and scene images, fuses geometrical features generated by a transformer encoder network, and outputs trajectories using a transformer decoder.

Our model can operate manipulate robot trajectories in 3D and velocity spaces. The output trajectory can be post-processed and applied towards diverse different platforms such as manipulation, aerial vehicles and legged robots. We provide a comprehensive set of simulated and real-world experiments demonstrating the effectiveness of our model and highlighting insights into what the model is learning.

In future iterations of this work we seek to explore additional modalities such as force inputs, as well as the ability of the model to interact with the user over longer time horizons and multiple instruction inputs. We hope that our framework can serve as a building block for a novel paradigms in human-robot collaboration that employ large language models.

REFERENCES

[1] A. Bucker, L. Figueredo, S. Haddadin, A. Kapoor, S. Ma, and R. Bonatti, "Reshaping robot trajectories using natural language commands: A study of multi-modal data alignment using transformers," *International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[2] A. Jain, S. Sharma, T. Joachims, and A. Saxena, "Learning preferences for manipulation tasks from online coactive feedback," *Int. J. Robotics Res.*, vol. 34, no. 10, pp. 1296–1313, 2015. [Online]. Available: https://doi.org/10.1177/0278364915581193

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[4] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[5] S. Smith, M. Patwary, B. Norick, P. LeGresley, S. Rajbhandari, J. Casper, Z. Liu, S. Prabhumoye, G. Zerveas, V. 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.

[6] S. Y. Gadre, M. Wortsman, G. Ilharco, L. Schmidt, and S. Song, "Clip on wheels: Zero-shot object navigation as object localization and exploration," *arXiv preprint arXiv:2203.10421*, 2022.

[7] M. Shridhar, L. Manuelli, and D. Fox, "Cliport: What and where pathways for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2022, pp. 894–906.

[8] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[9] P. Sharma, B. Sundaralingam, V. Blukis, C. Paxton, T. Hermans, A. Torralba, J. Andreas, and D. Fox, "Correcting robot plans with natural language feedback," *arXiv preprint arXiv:2204.05186*, 2022.

[10] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[11] C. E. Garcia, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[12] S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek, "Robots that use language," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 25–55, 2020.

[13] J. Arkin, D. Park, S. Roy, M. R. Walter, N. Roy, T. M. Howard, and R. Paul, "Multimodal estimation and communication of latent semantic knowledge for robust execution of robot instructions," *The International Journal of Robotics Research*, vol. 39, no. 10-11, pp. 1279–1304, 2020.

[14] M. R. Walter, S. Patki, A. F. Daniele, E. Fahnestock, F. Duvallet, S. Hemachandra, J. Oh, A. Stentz, N. Roy, and T. M. Howard, "Language understanding for field and service robots in a priori unknown environments," *arXiv preprint arXiv:2105.10396*, 2021.

[15] J. Fu, A. Korattikara, S. Levine, and S. Guadarrama, "From language to goals: Inverse reinforcement learning for vision-based instruction following," *arXiv preprint arXiv:1902.07742*, 2019.

[16] Y. Hong, Q. Wu, Y. Qi, C. Rodriguez-Opazo, and S. Gould, "A recurrent vision-and-language bert for navigation. arxiv 2021," *arXiv preprint arXiv:2011.13922*.

[17] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, and H. Ben Amor, "Language-conditioned imitation learning for robot manipulation tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 13 139–13 150, 2020.

[18] P. Goyal, R. J. Mooney, and S. Niekum, "Zero-shot task adaptation using natural language," *arXiv preprint arXiv:2106.02972*, 2021.

[19] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," *arXiv preprint arXiv:2201.07207*, 2022.

[20] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg, "Concept2robot: Learning manipulation concepts from instructions and human demonstrations," *The International Journal of Robotics Research*, vol. 40, no. 12-14, pp. 1419–1434, 2021.

[21] W. Goodwin, S. Vaze, I. Havoutis, and I. Posner, "Semantically grounded object matching for robust robotic scene rearrangement," *arXiv preprint arXiv:2111.07975*, 2021.

[22] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill *et al.*, "On the opportunities and risks of foundation models," *arXiv preprint arXiv:2108.07258*, 2021.

[23] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. S. Chaplot, O. Maksymets *et al.*, "Habitat 2.0: Training home assistants to rearrange their habitat," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[24] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. 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*, 2018, pp. 3674–3683.

[25] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8748–8763.

[26] C. Sun, A. Myers, C. Vondrick, K. Murphy, and C. Schmid, "Videobert: A joint model for video and language representation learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7464–7473.

[27] J. Lu, D. Batra, D. Parikh, and S. Lee, "Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," *Advances in neural information processing systems*, vol. 32, 2019.

[28] L. Zhou, H. Palangi, L. Zhang, H. Hu, J. Corso, and J. Gao, "Unified vision-language pre-training for image captioning and vqa," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 13 041–13 049.

[29] W. Su, X. Zhu, Y. Cao, B. Li, L. Lu, F. Wei, and J. Dai, "Vl-bert: Pre-training of generic visual-linguistic representations," *arXiv preprint arXiv:1908.08530*, 2019.

[30] W. Hao, C. Li, X. Li, L. Carin, and J. 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)*, 2020, pp. 13 134–13 143.

[31] J. Thomason, M. Murray, M. Cakmak, and L. Zettlemoyer, "Vision-and-dialog navigation," in *Conference on Robot Learning*. PMLR, 2020, pp. 394–406.

[32] K. Nguyen and I. Daumé, "Help, anna! visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning," 09 2019.

[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[34] F. Giuliari, I. Hasan, M. Cristani, and F. Galasso, "Transformer networks for trajectory forecasting," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 10 335–10 342.

[35] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch, "Decision transformer: Reinforcement learning via sequence modeling," *Advances in neural information processing systems*, vol. 34, 2021.

[36] M. Janner, Q. Li, and S. Levine, "Offline reinforcement learning as one big sequence modeling problem," *Advances in neural information processing systems*, vol. 34, 2021.

[37] P. J. Huber, "Robust estimation of a location parameter," in *Breakthroughs in statistics*. Springer, 1992, pp. 492–518.

[38] R. Bonatti, A. Bucker, S. Scherer, M. Mukadam, and J. Hodgins, "Batteries, camera, action! learning a semantic control space for expressive robot cinematography," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 7302–7308.

[39] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay *et al.*, "Roboturk: A crowdsourcing platform for robotic skill learning through imitation," in *Conference on Robot Learning*. PMLR, 2018, pp. 879–893.

[40] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[41] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.

[42] Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh, "Large batch optimization for deep learning: Training bert in 76 minutes," *arXiv preprint arXiv:1904.00962*, 2019.

[43] E. Rohmer, S. P. N. Singh, and M. Freese, "Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework," in *Proc.*

of The International Conference on Intelligent Robots and Systems (IROS), 2013, www.coppeliarobotics.com.
[44] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.