Supplementary Material

A. SVD vs. Data Augmentation for Trajectory Augmentation

We have compared the k-rank approximation used in our approach with popular data augmentation techniques used in trajectory prediction. Our findings show that data augmentation approaches produce suboptimal results compared to k-rank, which provides better representation learning, as Table 1 shows. The details regarding the data augmentation techniques used in these experiments are given below. **Scale**: In the scale trajectory augmentation, we generate a random scaling factor between [min, max]. This scaling factor is then applied to both observed and future trajectories. The augmentation returns the scaled observed trajectories and the scaled future trajectories. We use min = 0.8 and max = 1.2. **Flip**: In flip, we randomly select one of the four possible flip directions: horizontal flip, vertical flip, or both flips. It applies the selected flip direction to the observed trajectories and future trajectories. **Stretch**: In stretch augmentation, we randomly stretch both observed and future trajectories along horizontal and vertical dimensions independently within the specified range [min, max]. This augmentation generates two random stretch factors within the specified range [min, max], representing the stretch along the horizontal and vertical dimensions. We use min=0.9, max=1.1. **Noise**: In noise augmentation, we randomly add noise (Gaussian noise) to observed and future trajectories based on zero mean and a standard deviation of 0.01.

B. Additional Visualization

We have included an illustration of the Motion Cues generated by the LLM in Fig. 1

Table 1. Results for different data augmentation techniques used in trajectory prediction task. Results are reported in terms of ADE/FDE values on the SDD dataset. '+' indicates a gain (improvement) in ADE/FDE values, while '-' indicates a lose (degradation). *w/o* denotes without, *w/* denotes with, and *TA* denotes trajectory augmentation.

Augmentation	SDD		Difference
	ADE	FDE	w.r.t Baseline
LG-Traj w/o TA (Baseline)	8.26	13.59	-
LG-Traj w/ Flipping	8.30	13.65	-0.04/0.06
LG-Traj w/ Stretch	8.71	14.54	-0.45/0.95
LG-Traj w/ Scale	8.59	14.20	-0.33/0.61
LG-Traj w/ Noise	8.11	13.08	+0.15/0.51
LG-Traj w/ TA (Our)	**7.80**	**12.79**	+0.46/0.80

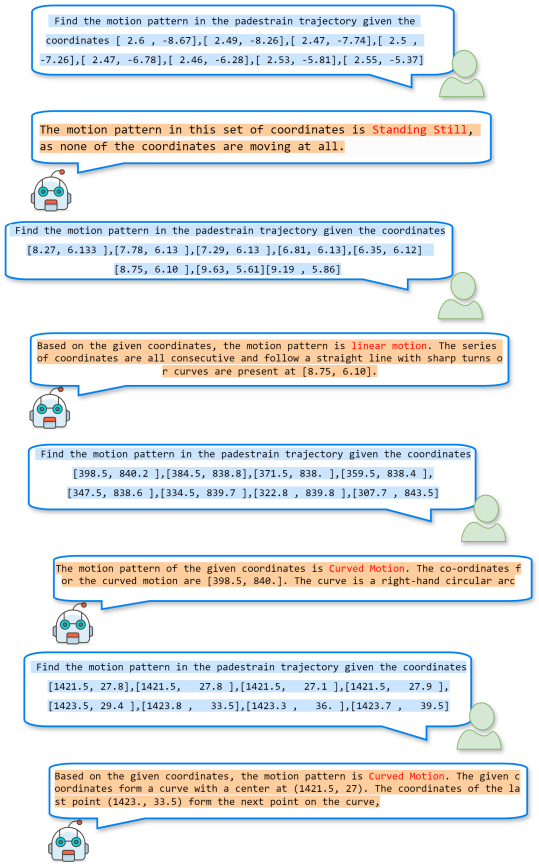

Figure 1. Illustration of input and responses of motion cues generation from the LLM. We present four different examples where the LLM identifies the underlying trajectory motion pattern, such as linear motion, curved motion, and standing still, based on the coordinates provided as input to the LLM.

C. Different Token Sizes for the Tokenizer

In this ablation study, we investigate the impact of token size on trajectory prediction performance through experiments with varied token sizes, as reported in Table 3. Our findings demonstrate that the optimal token size is 16 for the ETH and HOTEL datasets, while for the UNIV, ZARA1,

	k=1	*k=2*	*k=3*
ETH	13%	7%	3%
HOTEL	15%	9%	2%
UNIV	17%	10%	4%
ZARA 1	15%	8%	3%
ZARA 2	17%	8%	3%
AVG	15%	8%	3%
SDD	27%	15%	6%

Table 2. Information loss occurred (in %) for different k values in rank-k approximation for trajectory augmentation.

Table 3. Comparing the performance with different token sizes for the tokenizer, the optimal token size is 16 for ETH and HOTEL datasets, while for UNIV, ZARA1, ZARA2, and SDD datasets, the best result is achieved with a token size of 32.

Token Size	*16*	*32*	*64*
ETH	**0.38/0.56**	0.41/0.60	0.42/0.62
HOTEL	**0.11/0.17**	0.12/0.18	0.12/0.18
UNIV	0.24/0.43	**0.23/0.42**	0.24/0.45
ZARA1	0.18/0.33	**0.18/0.33**	0.19/0.45
ZARA2	0.14/0.25	**0.14/0.25**	0.14/0.26
SDD	7.85/12.90	**7.78/12.79**	7.80/12.86

ZARA2, and SDD datasets, the best results are achieved with 32 token size.

D. Information loss vs. *k* Value

We investigate the information loss incurred when augmenting the past trajectory using the k-rank approximation, and these losses are given in Table 2. Among the four singular values, the highest level of information loss is observed when $k = 1$, while for $k = 3$ there is only minor information loss. We use four singular values for trajectory augmentation; therefore, the size of the diagonal matrix is 4×4. We choose the k most significant singular values out of a total of four, where $k = 4$ represents the trajectory without any truncation or loss.