Supplementary material for: KoopMotion: Learning Almost Divergence Free Koopman Flow Fields for Motion Planning

S1 Additional Spectral Analyses—KoopMotion Eigenfunctions

In addition to performing stability analysis on the learnt system by studying the learnt Koopman operator, $\hat{\mathcal{K}}$, as shown in the main text, we can additionally visualize the system eigenfunctions. Let $\Phi(\mathbf{y})$ denote the eigenfunctions of $\hat{\mathcal{K}}$ such that the following is true, for a given state of interest $\mathbf{y}$:

$$\mathcal{K}\Phi(\mathbf{y}) = \lambda\Phi(\mathbf{y}) \tag{11}$$

For a eigenvector v_i of $\hat{\mathcal{K}}$, which provides a weighting of each of the learnt basis functions $\hat{\Psi}(\mathbf{y})$, the associated eigenfunction ϕ_i, is defined by:

$$\phi_i = v_i^T \Psi(\mathbf{y}) \tag{12}$$

Since we are interested in studying how the Koopman mode, as defined in [16], affects or contributes to the vector field, we define $\mathbf{y} = \hat{\mathbf{x}}_{k+1} - \hat{\mathbf{x}}_k$.

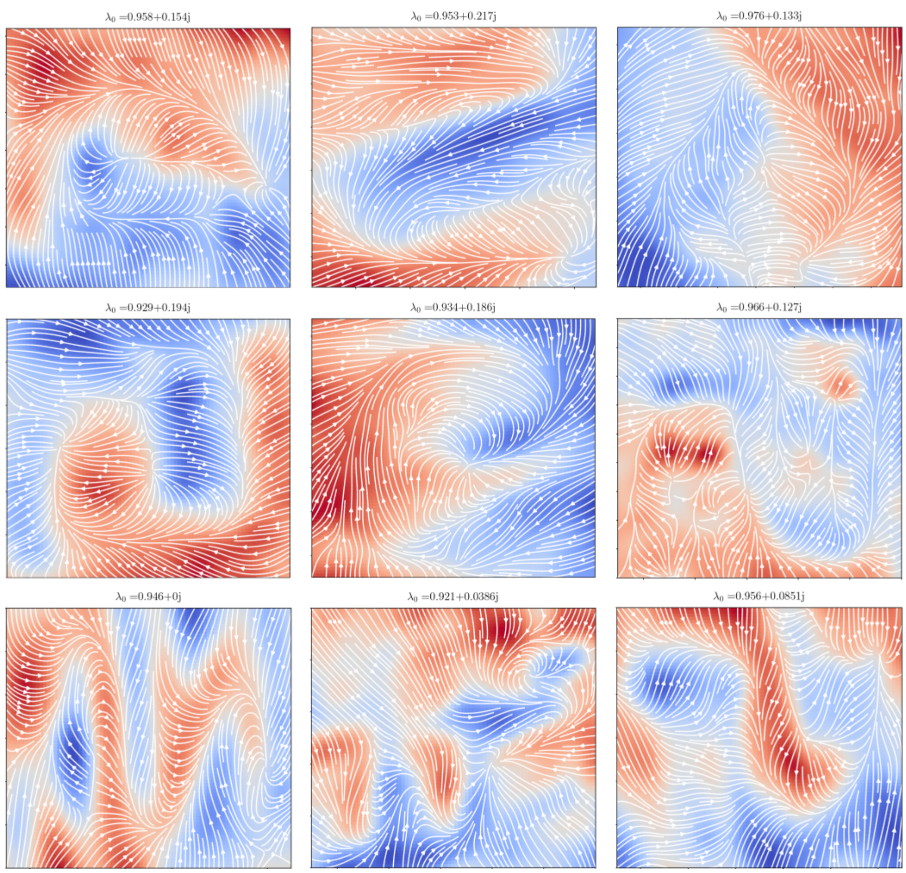


Figure 6: Eigenfunctions of KoopMotion models for shapes of the LASA dataset, based on the largest eigenvector of $\hat{\mathcal{K}}$. Eigenfunctions of the learnt Koopman operator represent groupings within the domain, where the space is partitioned into regions that exihibit similar dynamics. This is illustrated by partitions with the same color (eigenfunction value). The shapes shown here from left to right, top to bottom are BENDEDLINE, ZSHAPE, LEAF_1, JSHAPE_2, PSHAPE, MULTI_MODELS_4, SINE, MULTI_MODELS_3, SPOON

S2 KoopMotion Hyperparameters

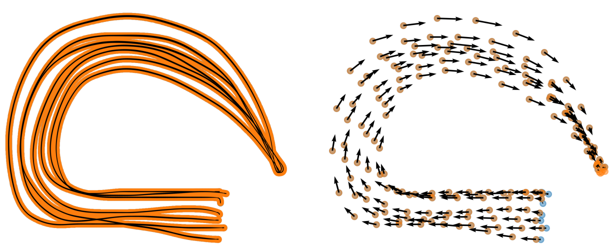
We train all models on the same hyperparameters to demonstrate the lack of fine-tuning required for each modeling vector fields for each individual nonlinear shape, and therefore lack of reliance on defining a a unique hyperparameter for good models (*e.g.,* in prior work, requiring a reliance on defining the number of Gaussian models that make up a GMM).

For this work, we use Adam, a stochastic gradient based optimizer, for optimization over the learnable parameters for the Fourier features $\hat{\omega}, \hat{b}$ and for the Koopman operator $\hat{\mathcal{K}}$, defined in Equation 6. Based on the same Equation 6, we select $\beta_k = 1$, $\beta_d = 0.01$, and $\beta_g = 0.01$ for all examples shown. We lift the system to a higher dimensional space of dimension $\nu = 1024$. We train every KoopMotion model for 2200 total iterations, with 200 epochs and a batch size of 16, with 168 trajectory points. We use a learning rate of $8e^{-4}$) In practice, we use a low-rank form of the Koopman operator to minimize the number of parameters to be learnt. Meaning, we use $\hat{\mathcal{K}} = AB^T$, where $A \in \mathbf{R}^{(\nu+d)\times r}$ and $B \in \mathbf{R}^{(\nu+d)\times r}$ are matrices, and $\hat{\mathcal{K}} \in \mathbf{R}^{(\nu+d)\times(\nu+d)}$, where ν is the number of Fourier features, and d represents the dimension of the original system, which for planar data such as the LASA dataset $d = 2$, otherwise for 3d trajectories, $d = 3$. We use $r = 32$. For a model with $\nu = 1024$, the model is defined by approximately 70k parameters.

S3 Sparse Training Data

As detailed in the main text, we train on 3% of the original LASA dataset. To demonstrate how little data is used, we include examples below. Every demonstration has 25 time-steps of the trajectory. We train on the 7 demonstrations in the dataset. This gives rise to 168 time-steps worth of trajectories in the training data (24 pairs of time-shifted data).

During evaluation using the metrics defined in the main text (DTWD and SEA), we compare time integrated data against the entire dataset, which has not been sub-sampled in time.



(a) All training data. (b) Training data used

Figure 7: Sparsity of the dataset.

S4 Ablation Studies—Number of Fourier Features

Across all of the shapes, we perform an ablation study on the effects of the number of Fourier features, ν on the two evaluation metrics defined in the main text, DTWD and SEA.

Table 1: Metric Average ($\pm$ Standard Deviation)

Metric/ν	500	750	1000	1250	1500
DTWD	1432.40 ($\pm$ 413.97)	1548.00 ($\pm$ 574.38)	1595.49 ($\pm$ 537.12)	1576.91 ($\pm$ 504.71)	1963.84($\pm$ 1380.01)
SEA	148.35 ($\pm$ 69.55)	162.76 ($\pm$ 78.46)	169.39 ($\pm$ 73.95)	174.94 ($\pm$68.15)	154.06($\pm$ 75.70)

S5 Additional Robot Experiments

We include additional real-robot experiments that further validate the effectiveness of KoopMotion. For these experiments, we limit the vehicle's motion to half of the maximum velocity, for smoother trajectory following using the KoopMotion vector fields, with a PID controller tuned for aggressive steering, as shown for following the SINE shape well.

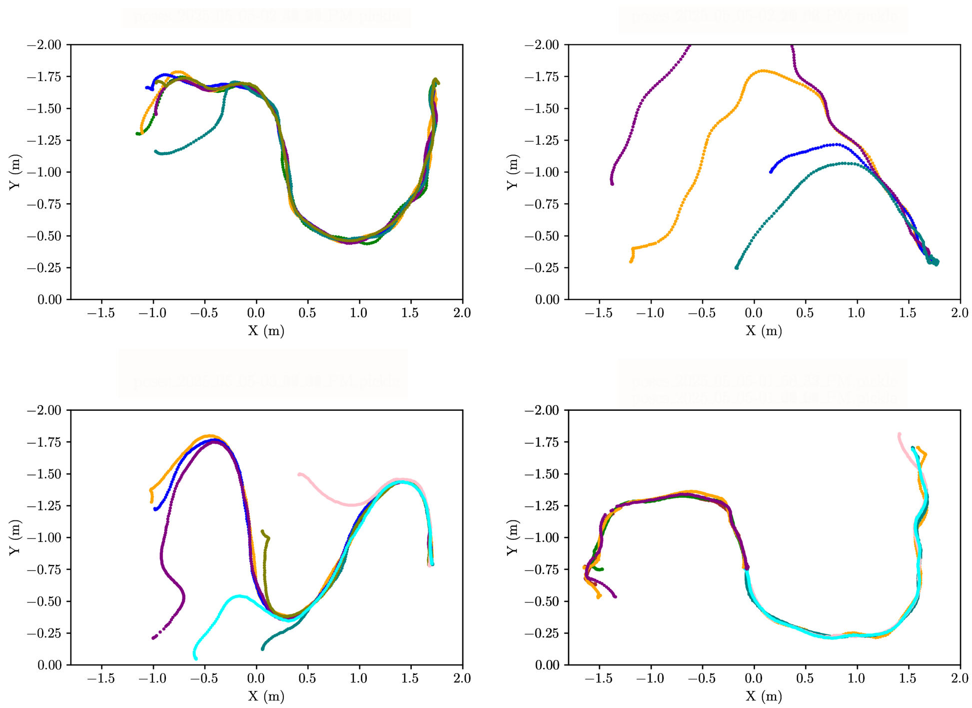


Figure 8: Additional results of the miniature autonomous surface vehicle using the vector fields for motion planning, where vector field is rescaled to half of the max vehicle velocity. The shapes shown here from left to right, top to bottom are SPOON, ANGLE, SINE, BENDEDLINE, ZSHAPE, MULTI_MODELS_4

S6 Modeling Trajectories in Higher Dimensional Spaces

We also perform evaluations on an additional datasets of trajectories in 3D, generated from the Robocasa simulator [38]. Datasets are acquired by teleoperating a 7-DOF robot moving in the simulator, and collecting the pose of the end-effector.

In our results, we show that we are successfully able to model the nonlinearities of the 3D trajectories in the training data. The eigenfunctions are able to additionally partition the dynamics of the three different modes of trajectories. And finally, the learnt modelisstable, as the magnitude of the largest eigenvalue is within the unit circle.

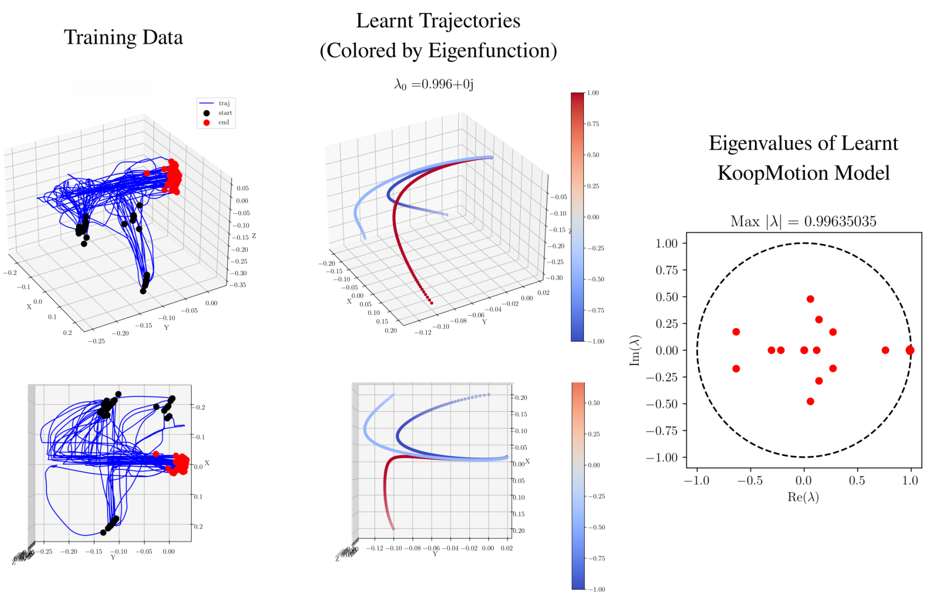


Figure 9: Learnt KoopMotion R^3 dynamics for multi-modal trajectory data (3 separate groups of starting locations, converging to one goal position). LEFT is the training data of an end effector operating in 3-dimensional space. MIDDLE is the learnt trajectories of points propagated by KoopMotion from initial conditions matching those in training data. These trajectories are colored by the eigenfunction value of these initial conditions. Given the three distinguishing colors, (dark blue, light blue, and red), similar to plots in Fig. 6, we demonstrate that eigenfunctions which are obtained from the learnt Koopman operator, partition the space into regions that exhibit similar dynamics. This partitioning is not readily available to the user with existing works, (e.g., GMMs, NODEs etc.).