A APPENDIX

A.1 CONTINUOUS CONVOLUTION INVOLVING ρ_{reg}

This section is a more detailed version of Section 4.4.

Define the input $\mathbf{f}$ to be ρ_{reg}-field, that is, a distribution over $\mathbb{R}^2$ valued in ρ_{reg}. Define $K\colon \mathbb{R}^2 \to \rho_{\text{reg}} \otimes \rho_{\text{reg}}$. After identifying $\text{SO}(2)$ with its underlying manifold S^1, we can identify $K(\mathbf{x})$ as a map $S^1 \times S^1 \to \mathbb{R}$ and $f^{(\mathbf{x})}\colon S^1 \to \mathbb{R}$. Define the integral transform

$$K(\mathbf{x}) \circledcirc f^{(\mathbf{x})}(\phi_2) = \int_{\phi_1 \in S^1} K(\mathbf{x})(\phi_2, \phi_1) f^{(\mathbf{x})}(\phi_1) d\phi_1.$$

For $\mathbf{y} \in \mathbb{R}^2$, define the convolution $\mathbf{g} = K \star \mathbf{f}$ by

$$\mathbf{g}(y) = \int_{x \in \mathbb{R}^2} K(x) \circledcirc \mathbf{f}(x + y) dx.$$

The $\circledcirc$-operation parameterizes linear maps $\rho_{\text{reg}} \to \rho_{\text{reg}}$ and is thus analogous to matrix multiplication. If we chose to restrict our choice of κ to $\kappa(\phi_2, \phi_1) = \tilde{\kappa}(\phi_2 - \phi_1)$ for some function $\tilde{\kappa}\colon S^1 \to \mathbb{R}$ then this becomes the circular convolution operation.

The $SO(2)$-action on ρ_{reg} by $\text{Rot}_\theta(f)(\phi) = f(\phi - \theta)$ induces an action on $\kappa\colon S^1 \times S^1 \to \mathbb{R}$ by

$$\text{Rot}_\theta(\kappa)(\phi_2, \phi_1) = \kappa(\phi_2 - \theta, \phi_1 - \theta).$$

This, in turn, gives an action on the torus-field K by

$$\text{Rot}_\theta(K)(x)(\phi_2, \phi_1) = K(\text{Rot}_{-\theta}(x))(\phi_2 - \theta, \phi_1 - \theta).$$

Thus Equation 3, the convolutional kernel constraint, implies that K is equivariant if and only if

$$K(\text{Rot}_\theta(x))(\phi_2, \phi_1) = K(x)(\phi_2 - \theta, \phi_1 - \theta).$$

We use this to define a weight sharing scheme as described in Section 3.2. The cases of continuous convolution $\rho_1 \to \rho_{\text{reg}}$ and $\rho_{\text{reg}} \to \rho_1$ may be derived similarly.

A.2 COMPLEXITY OF CONVOLUTION WITH TORUS KERNEL

The complexity class of the convolution with torus kernel is $O(n \cdot k_{reg}^2 \cdot c_{out} \cdot c_{in})$, where n is the number of particles, the regular representation is discretized into k_{reg} pieces, and the input and output contain c_{in} and c_{out} copies of the regular representation respectively. We are not counting the complexity of the interpolation operation for looking up $K(\theta, r)$.

A.3 EQUIVARIANT PER-PARTICLE LINEAR LAYERS

Since this operation is pointwise, unlike positive radius continuous convolution, we cannot map between different irreducible representations of $\text{SO}(2)$. Consider as input a ρ_{in}-field I and output a ρ_{out}-field O where ρ_{in} and ρ_{out} are finite-dimensional representations of $\text{SO}(2)$. We define $O^{(i)} = WI^{(i)}$ using the same W, an equivariant linear map, for each particle $1 \leq i \leq N$. Denote the decomposition of ρ_{in} and ρ_{out} into irreducible representations of $SO(2)$ as $\rho_{\text{in}} \cong \rho_1^{i_1} \oplus \ldots \oplus \rho_n^{i_n}$ and $\rho_{\text{out}} \cong \rho_1^{j_1} \oplus \ldots \oplus \rho_n^{j_n}$ respectively. By Schur's lemma, the equivariant linear map $W\colon \rho_{\text{in}} \to \rho_{\text{out}}$ is defined by a block diagonal matrix with blocks $\{W_k\}_{k=1}^n$ where W_k is an $i_k \times j_k$ matrix. That is, maps between different irreducible representations are zero and each map $\rho_k \to \rho_k$ is given by a single scalar.

Per-particle linear mapping $\rho_1 \to \rho_{\text{reg}}$ and $\rho_1 \to \rho_{\text{reg}}$. Since the input and output features are ρ_1-fields, but the hidden features may be represented by ρ_{reg}, we need mappings between ρ_1 and ρ_{reg}. In all cases we pair continuous convolutions with dense per-particle mappings, this we must describe per-particle mappings between ρ_1 and ρ_{reg}.

By the Peter-Weyl theorem, $L^2(\mathrm{SO}(2)) \cong \bigoplus_{i=0}^{\infty} \rho_i$. In the case of $\mathrm{SO}(2)$, this decomposition is also called the Fourier decomposition or decomposition into circular harmonics. Most importantly, there is one copy of ρ_1 inside of $L^2(\mathrm{SO}(2))$. Hence, up to scalar, there is a unique linear map $i_1 \colon \rho_1 \to L^2(\mathrm{SO}(2))$ given by $(a, b) \mapsto a\cos(\theta) + b\sin(\theta)$.

The reverse mapping $\mathrm{pr}_1 \colon L^2(\mathrm{SO}(2)) \to \rho_1$ is projection onto the ρ_1 summand and is given by the Fourier transform $\mathrm{pr_i}(f) = (\int_{S^1} f(\theta)\cos(\theta)d\theta, \int_{S^1} f(\theta)\sin(\theta)d\theta)$.

Per-particle linear mapping $\rho_{\mathrm{reg}} \to \rho_{\mathrm{reg}}$. Though ρ_{reg} is not finite-dimensional, the fact that it decomposes into a direct sum of irreducible representations means that we may take $\rho_{\mathrm{in}} = \rho_{\mathrm{out}} = \rho_{\mathrm{reg}}$ above. Practically, however, it is easier to realize the linear equivariant map $\rho_{\mathrm{reg}}^i \to \rho_{\mathrm{reg}}^j$ as a convolution over S^1,

$$O(\theta) = \int_{\phi \in S^1} \kappa(\theta - \phi) I(\phi)$$

where $\kappa(\theta)$ is an $i \times j$ matrix of trainable weights, independent for each θ.

A.4 Encoding Individual Particle Past Behavior

We can encode these individual attributes using a per vehicle `LSTM` (Hochreiter & Schmidhuber, 1997). Let $X_t^{(i)}$ denote the position of car i at time t. Denote a fully connected `LSTM` cell by $h_t, c_t = \texttt{LSTM}(X_t^{(i)}, h_{t-1}, c_{t-1})$. Define $h_0 = c_0 = 0$. We then use the concatenation of the hidden states $[h_{t_{\mathrm{in}}}^{(1)} \ \ldots \ h_{t_{\mathrm{in}}}^{(n)}]$ of all particles as $Z \in \mathbb{R}^N \otimes \mathbb{R}^k$ as the encoded per-vehicle latent features.

A.5 Encoding Past Interactions

In addition, we also encode past interactions of particles by introducing a continuous convolution `LSTM`. Similar to `convLSTM` we replace the fully connected layers of the original `LSTM` above with another operation Xingjian et al. (2015). While `convLSTM` is well-suited for capturing spatially local interactions over time, it requires gridded information. Since the particle system we consider are distributed in continuous space, we replace the standard convolution with rotation-equivariant continuous convolutions.

We can now define $H_t, C_t = \texttt{CtsConvLSTM}(X_t, H_{t-1}, C_{t-1})$ which is an `LSTM` cell using equivariant continuous convolutions throughout. Note that in this case X_t, H_{t-1}, C_{t-1} are all particle feature fields, that is, functions $\{1, \ldots, n\} \to \mathbb{R}^k$.

Define `CtsConvLSTM` by

$$\begin{aligned}
i_t &= \sigma(W_{ix} \star_{cts} X_t^{(i)} + W_{ih} \star_{cts} h_{t-1} + W_{ic} \circ c_{t-1} + b_i) \\
f_t &= \sigma(W_{fx} \star_{cts} X_t^{(i)} + W_{fh} \star_{cts} h_{t-1} + W_{fc} \circ c_{t-1} + b_i) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \tanh(W_{cx} \star_{cts} X_t^{(i)} + W_{ch} \star_{cts} h_{t-1} + b_c) \\
o_t &= \sigma(W_{ox} \star_{cts} X_t^{(i)} + W_{oh} \star_{cts} h_{t-1} + W_{oc} \circ c_t + b_o) \\
h_t &= o_t \circ \tanh(c_t),
\end{aligned}$$

where $\star_{cts}$ denotes `CtsConv`. We then can use $H_{t_{\mathrm{in}}}$ as input feature for the prediction network.

A.6 Equivariance Error

We prove the proposition in Section 4.5.

Proposition. *Let $\alpha = 2\pi/k_\theta$. Let $\bar{\theta}$ be θ rounded to nearest value in $\mathbb{Z}\alpha$. Set $\hat{\theta} = |\theta - \bar{\theta}|$. Assume n particles samples uniformly in a ball of radius R with features $\mathbf{f} \in \rho_1^c$. Let $\mathbf{f}$ and K have entries sampled uniformly in $[-a, a]$. Let the bullseye have radius $0 < R_e < R$. Let $F = \texttt{CtsConv}_{K,R}$ and $T_\theta = \rho_1(\mathrm{Rot}_\theta)$. Then the expected* EE *is bounded*

$$\mathbb{E}_{K,\mathbf{f},\mathbf{x}}[T(F(\mathbf{f}, \mathbf{x})) - F(T(\mathbf{f}), T(\mathbf{x}))] \leq |\sin(\hat{\theta})| C \leq 2\pi C / k_\theta$$

where $C = 4cna^2(1 - R_e^2/R^2)$.

Proof. We may compute for a single particle $\mathbf{x} = (\psi, r)$ and multiply our result by n by linearity. We separate two cases: $\mathbf{x}$ in bullseye with probability R_e^2/R^2 and $\mathbf{x}$ in angular slice with probability $1 - R_e^2/R^2$. If $\mathbf{x}$ is in the bullseye, then there is no equivariance error since $K(\mathbf{x})$ is a scalar matrix. Assume $\mathbf{x}$ is an angular sector.

For nearest interpolation, the equivariance error is then

$$\|\rho_1(\bar{\theta})K(\mathbf{x})\rho_1(-\bar{\theta})\rho_1(\theta)\mathbf{f} - \rho_1(\theta)K(\mathbf{x})\mathbf{f}\|.$$

Since $\rho_1(\theta)$ is length preserving, this is

$$\begin{aligned} &\|\rho_1(-\theta)\rho_1(\bar{\theta})K(\mathbf{x})\rho_1(-\bar{\theta})\rho_1(\theta)\mathbf{f} - K(\mathbf{x})\mathbf{f}\| \\ =&\|\rho_1(\beta)K(\mathbf{x})\rho_1(-\beta)\mathbf{f} - K(\mathbf{x})\mathbf{f}\| \end{aligned} \tag{7}$$

where $\beta = \pm\hat{\theta}$. We consider only a single factor of ρ_1 in $\mathbf{f}$. The result will then be multiplied by c. Let

$$K(\mathbf{x}) = \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix}, \qquad \mathbf{f} = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix}.$$

We can factor out an a from $K(\mathbf{x})$ and an a from $\mathbf{f}$ and assume k_{ij}, f_i samples from $\mathrm{Uniform}([-1, 1])$. One may then directly compute that Equation 7 equals

$$\sqrt{((k_{21} + k_{12})^2 + (k_{11} - k_{22})^2)(f_1^2 + f_2^2)\sin^2(\beta)}$$

This is bounded above by $4|\sin(\beta)| = 4|\sin(\hat{\theta})|$. Collecting the above factors, this proves the bound $C|\sin(\beta)|$.

The further bound follows by the first order bound,

$$|\sin(\hat{\theta})| \leq |\hat{\theta}| \leq 2\pi/k_\theta.$$

□

The relationship $\mathrm{EE} \approx 2\pi C/k_\theta$ is visible in Figure 4. We can also see clearly the significance of the term $|\sin(\hat{\theta})|$ by plotting equivariance error against θ as in Figure 7.

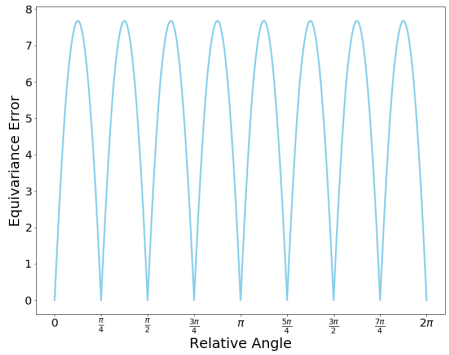

Figure 7: The above plot is generated from random input and kernels. We can clearly see the dependence of of EE on $|\sin(\hat{\theta})|$

A.7 Data Details

Argoverse dataset includes 324K samples, which are split into 206K training data, 39K validation and 78K test set. All the samples are real data extracted from Miami and Seattle, and the dataset provides HD maps of lanes in each city. Every sample contains data for 5 seconds long, and is sampled in 10Hz frequency.

TrajNet++ Real dataset contains 200K samples. All the tracking in this dataset is captured in both indoor and outdoor locations, for example, university, hotel, Zara, and train stations. Every sample in this dataset contains 21 timestamps, and the goal is to predict the 2D spatial positions for each pedestrain in the future 12 timestamps.

A.8 IMPLEMENTATION DETAILS

Argoverse dataset is not fully observed, so we only use cars with complete observation as our input. Since every sample doesn't include the same number of cars, we only choose those scenes with less than or equal to 60 cars and insert dummy cars into them to achieve consistent car numbers. TrajNet++ Real dataset is also not fully observed. And here we keep our pedestrain number consistent to 160.

Moreover, for each car, we use the average velocity in the past 0.1 second as an approximate to the current instant velocity, i.e. $v_t = (p_t - p_{t-1})/2$. As for map information, we only include center lanes with lane directions as features. Also, we introduce dummy lane node into each scene to make lane numbers consistently equal to 650.

In TrajNet++ task, no map information is included. And since pedestrians don't have a speedometers to tell them exactly how fast they are moving as drivers, instead they depends more on the relative velocities and relative positions to other pedestrians, we tried different combination of features in ablative study besides only using history velocities.

Our models are all trained by Adam optimizer with base learning rate 0.001, and the gamma rate for linear rate scheduler is set to be 0.95. All our models without map information are trained for 15K iterations with batch size 16 and learning rate is updated every 300 iterations; for models with map information, we train them for 30K iterations with batch size 16 and learning rate is updated every 600 iterations.

For CtsConv, we set the layer sizes to be 32, 64, 64, 64, and kernel size $4 \times 4 \times 4$; for ρ_1-ECCO, the layer sizes are 16, 32, 32, 32, k_θ is 16, k_r is 3; for ρ_{reg}-ECCO, we choose layer size 8, 16, 8, 8, k_θ 16, k_r 3, and regular feature dimension is set to be 8. For Argoverse task, we set the CtsConv radius to be 40, and for TrajNet++ task we set it to be 6.

A.9 ABLATIVE STUDY

We perform ablative study for ECCO to further diagnose different encoders, usage of HD maps and other model design choices.

Choice of encoders Unlike fluid simulations (Ummenhofer et al., 2019) where the dynamics are Markovian, human behavior exhibit long-term dependency. We experiment with three different encoders refered to as `Enc` to model such long-term dependency: (1) concatenating the velocities from the past m frames as input feature, (2) passing the past velocities of each particle to the same LSTM to encode individual behavior of each particle, and (3) implementing continuous convolution LSTM to encode past particle interactions. Our continuous convolution LSTM is similar to convLSTM (Xingjian et al., 2015) but uses continuous convolutions instead of discrete gridded convolutions.

We use different encoders to time-aggregate features and compare their performances (Table 3).

Use of HD Maps In Table 4, we compare performance with and without map input features.

Choice of features for pedestrian Unlike vehicles, people do not have a velocity meter to tell him how fast they actually walk. We realize that people actually tend to adjust their velocities based on others' relative velocity and relative position. We experiment different combination of features (Table 5), finding using relative velocities and relative positions as feature has the best performance.

A.10 QUALITATIVE RESULTS FOR TRAJNET++

Figure 8 show qualitative results for TrajNet++. Note that the non-equivariant baseline (2nd column) depends highly on the global orientation whereas the ground truth and equivariant models do not.

Encoder	Argoverse				TrajNet++	
	ADE	DE@1s	DE@2s	DE@3s	ADE	FDE
Markovian	4.67	-	-	9.84	0.969	1.952
LSTM	2.05	1.06	2.51	4.71	0.909	1.909
CtsConvLSTM	3.98	2.02	5.11	8.40	0.962	1.941
CtsConvDLSTM	2.02	1.03	2.46	4.58	0.910	1.916
D-Concat(20t feats)	**1.87**	**1.01**	**2.43**	**4.22**	**0.895**	**1.872**

Table 3: Ablation study on encoders for Argoverse and TrajNet++. Markovian: Use the velocity from the most recent time step as input feature. LSTM: Used LSTM to encode velocities of 20 timestamps. CtsConvLSTM: Instead of dense layer, the gate functions in LSTM are replaced by CtsConv. CtsConvDLSTM: Replaced gate functions by CtsConv + Dense. D-Concat (20t feats): Stacked velocities of 20 time steps as input.

Model	w/o Map				w/ Map			
	ADE	DE@1s	DE@2s	DE@3s	ADE	DE@1s	DE@2s	DE@3s
CtsConv	1.87	1.01	2.43	4.22	1.85	0.99	2.42	4.32
ρ_1-ECCO	**1.81**	1.02	2.42	4.14	1.70	0.93	2.22	3.89
ρ_{reg}-ECCO	**1.81**	**1.00**	**2.38**	**4.12**	**1.62**	**0.89**	**2.12**	**3.68**

Table 4: Ablative study on HD maps for Argoverse. Prediction accuracy comparison with and without HD Maps.

Velocity	Relative Position	Acceleration	ADE	FDE
Absolute	×	×	0.92	1.95
Absolute	×	✓	0.90	1.87
Relative	×	✓	0.89	1.86
Relative	✓	✓	**0.86**	**1.79**

Table 5: Ablative study on features for Traj++. Acceleration means whether we used acceleration to make numerically extrapolated position.

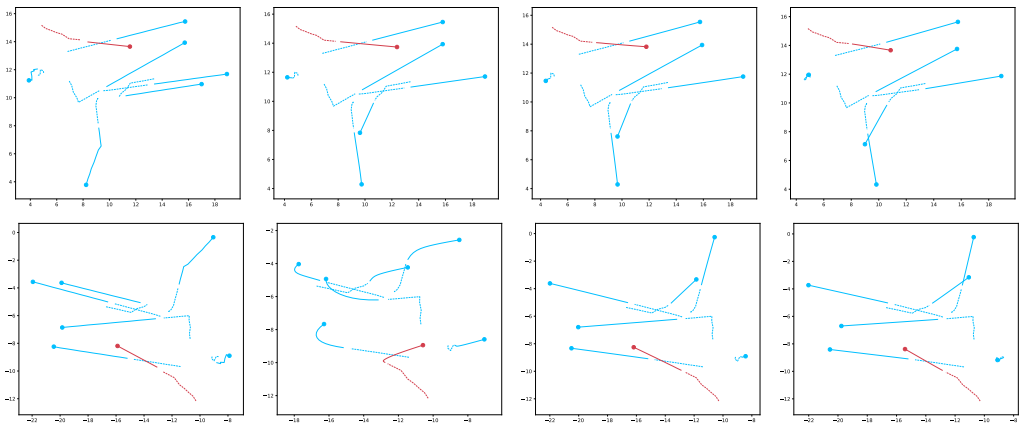

Figure 8: The x,y-axes are the position (m). The dashed line represents the 2s past trajectory. The solid line represents the 3s prediction. Red represents the agent. Top row: The predictions are made on the original data. Bottom row: We rotate the whole scene by 160° and make predictions on rotated data. From left to right are visualizations of ground truth, CtsConv, ρ_1-ECCO, ρ_{reg}-ECCO.