DELVING INTO DISCRETE NORMALIZING FLOWS ON SO(3) MANIFOLD FOR PROBABILISTIC ROTATION MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Normalizing flows (NFs) provide a powerful tool to construct an expressive distribution by a sequence of trackable transformations of a base distribution and form a probabilistic model of underlying data. Rotation, as an important quantity in computer vision, graphics and robotics, can exhibit many ambiguities when occlusion and symmetry occur and thus demands such probabilistic models. Though various NFs in Euclidean space have been proposed, there are no normalizing flows tailored for $\mathrm{SO}(3)$ manifold. Given the unique non-Euclidean properties of the rotation manifold, adapting the existing NFs to $\mathrm{SO}(3)$ manifold is non-trivial. In this paper, we propose a novel normalizing flow on $\mathrm{SO}(3)$ by combining a Möbius transformation-based layer and a quaternion affine transformation. With our proposed rotation normalizing flows, one can not only effectively express arbitrary distributions on $\mathrm{SO}(3)$, but also conditionally build the target distribution given input observations. Extensive experiments show that our rotation normalizing flows significantly outperform the baselines on both unconditional and conditional tasks.

1 INTRODUCTION

Endowing a neural network with the ability to express uncertainty along with the prediction is of crucial influence to safety and interpretability-critical systems and provides valuable information for downstream tasks (Schwalbe & Schels, 2020; Leibig et al., 2017; Ching et al., 2018). As a widely used technique in computer vision and robotics, rotation regression can also benefit from such uncertainty-aware predictions and enable many applications (Sattler et al., 2019; Glover et al., 2012; Crassidis & Markley, 2003).

To this end, recent years have witnessed much effort in modeling the uncertainty of rotation via probabilistic modeling of the $\mathrm{SO}(3)$ space, including von-Mises distribution for Euler angles, Bingham distribution for quaternions, matrix Fisher distribution for rotation matrices, etc. Those distributions are all single-modal, which fall short on modeling objects with continuous symmetry, which are ubiquitous in our daily life. Taking *cup* as an example, it exhibits rotational symmetry for which modeling with the unimodal or the mixture of distributions is clearly insufficient. How to model an *arbitrary* distribution on $\mathrm{SO}(3)$ manifold is still a challenging open problem.

One choice is to model distributions on $\mathrm{SO}(3)$ manifold via discretization, which has been proposed by Implicit-PDF (Murphy et al., 2021). Implicit-PDF takes input an arbitrary rotation and outputs its corresponding **unnormalized** pdf, thus reconstructing the unnormalized distribution with sampling strategies and is required to compute the normalization constant via dense sampling to obtain the full discrete approximation of the target distribution. Obviously, the influence of discretization error and the computational cost is a trade-off pair. Moreover, the discretized distribution does not offer an efficient method to sample from the target distributions.

Normalizing flows (Rezende & Mohamed, 2015), which maps samples from a simple base distribution to samples from the target distributions via invertible transformations, provides a flexible way to express complex distributions and appears to be a strong candidate in modeling arbitrary rotation distributions.

However, despite the great progress in normalizing flows in Euclidean space (Dinh et al., 2014; 2016; Kingma & Dhariwal, 2018; Ho et al., 2019; Behrmann et al., 2019; Chen et al., 2019), there is little trial on how to effectively construct flows on the non-Euclidean $\mathrm{SO}(3)$ manifold and converting the existing flows into the rotation space is highly non-trivial due to the uniqueness of the $\mathrm{SO}(3)$ manifold.

In this work, we propose a novel rotation normalizing flows on $\mathrm{SO}(3)$ manifold. We, for the first time, build discrete normalizing flows on rotation matrix representation, and have no singularity encounter in Falorsi et al. (2019). Our flows can be arbitrarily flexible, and provide a way for efficient sampling and probability inference.

One block of our discrete rotation normalizing flow is composed of a Möbius coupling layer and an affine transformation in quaternion space, connected by conversions between rotations and quaternions. Our Möbius coupling layer leverages the Möbius transformation, whose parameter is conditioned on one column of the rotation matrix, to map another column while maintaining unit length and thus effectively rotates the other two columns of the rotation matrix. Multiple Möbius coupling layers can be linearly combined to improve efficiency and we propose a constraint to resolve the involved discontinuity. Furthermore, to further improve the expressivity of this flow, we propose to convert the rotation to a quaternion and perform an affine transformation in quaternion space. This affine transformation is complementary to the Mobius coupling layer and can serve global rotation as well as condensing or dilating local likelihood in $\mathrm{SO}(3)$. Together with the two modules, our discrete rotation normalizing flow is very expressive and maintains the invertibility.

We conduct extensive experiments to validate the expressivity and stability of our proposed rotation normalizing flows. The results show that our rotation normalizing flows are able to either effectively fit the target distributions on $\mathrm{SO}(3)$ with distinct shapes, or regress the target distribution given input image conditions, where our method achieves superior performance over all the baselines.

2 Normalizing Flows on Riemannian Manifold

Normalizing flows (NFs) provide a flexible way to construct complex distributions in high-dimensional Euclidean space by transforming a base distribution through an invertible and differential mapping. It can be extended to Riemannian manifolds embedded in a higher dimensional space (Papamakarios et al., 2021)(Gemici et al., 2016). Formally, normalizing flows transform base distributions $\pi(x), x \in \mathcal{M}$ to target distributions $p(y), y \in \mathcal{N}$, where $\mathcal{M}, \mathcal{N}$ are Riemannian manifold and have the same topology, via diffeomorphisms T. The probability density function(pdf) of x can be calculated by change of variable formulas:

$$p(x) = \pi(T^{-1}(x))|\det J_{T^{-1}}(x)|, \tag{1}$$

where $J_{T^{-1}}(x) = \frac{\partial(T(x))}{\partial x}$ is the $D \times D$ partial derivatives of T^{-1} at x.

As diffeomorphisms are composable, in practice, the transformation T is often implemented via a sequence of simple transformations $T = T_K \circ \cdots \circ T_2 \circ T_1$, whose Jacobian determinants are easy to evaluate. The determinant of the composed transformation is given by:

$$\det(J_T(u)) = \Pi_{i=1}^{k} \det(J_{T_{i+1}}(T_i(u))) \tag{2}$$

The advantage of normalizing flows is that it enables both forward and inverse processes and one can calculate $p(x)$ during the process, so to fit the target distribution we can maximize the negative log-likelihood (NLL) of the training data in the inverse process, and to sample from the model distribution, one can sample from the base distribution and transform it in the forward process.

3 METHOD

The first question for a flow on $\mathrm{SO}(3)$ manifold is the representation of rotations. As shown in Zhou et al. (2019), representations in Euclidean space of less than 5 Dimension are discontinuous, so we choose rotation matrix, a 9D representation, and design 2 diffeophsisms on $\mathrm{SO}(3)$ manifold. One is the Möbius coupling directly built on rotation matrix, and the other is constructed by a conversion from rotation on quaternion followed by affine transformation defined in quaternion space and a

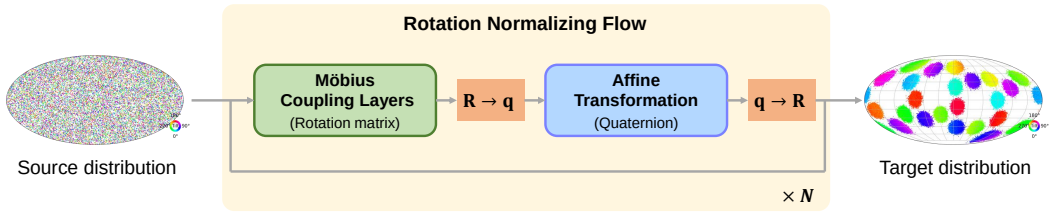


Figure 1: **Pipeline overview.** Our flow model takes rotations as input and outputs transformed rotations and log determinants of Jacobian, transforming a uniform distribution to a target one. Our flow is done by iteratively altering Möbius coupling on Rotation matrix representation and Affine transformation on Quaternion for K times. For probability inference, data are fed into the flow to the corresponding rotation which is of uniform distribution and predict log-likelihood; while in the sampling process, our flow runs inversely, generating new data by transforming samples from the uniform distribution. The distribution visualization is borrowed from (Murphy et al., 2021) where $\mathrm{SO}(3)$ is projected to a 2D sphere by Hopf fibration, points on the 2D sphere indicate the direction of a canonical z-axis, the colors represent the tilt angle about that axis, the direction of a canonical z-axis and the sizes of points show the probability density.

conversion back to rotation matrix. Intuitively, our model is similar to Glow(Kingma & Dhariwal, 2018) with the first part serving as coupling layer which can create complex transformation using multilayer perceptron(MLP) and the last part serving as 1×1 convolution which can make the first part more flexible using learnable permutation.

Our flow is composed of multiple blocks of layers, each of which consists of a Möbius flow and an Affine flow. The Möbius flow is constructed via Möbius transformation on rotation matrix, while the Affine flow is built using affine transformations on quaternion. By iteratively adding Möbius flow and Affine flow to our model, it is able to learn complex distributions. An illustration of our model is shown in Figure. 1.

3.1 Möbius Transformation on rotation matrix

Möbius Transformation is defined on a D-dimensional sphere S^D. Rezende et al. (2020) has applied it to build expressive normalizing flows on the 2D circle S^1. Here we begin by a brief introduction to Möbius Transformation, and show our adoption to $\mathrm{SO}(3)$ manifold afterwards.

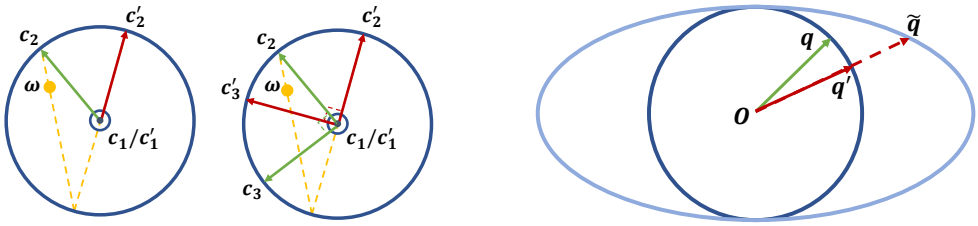
Revisit Möbius transformation. Möbius Transformation on $\mathcal{S}^D$ can be parameterized by an $\omega \in \mathbb{R}^{n+1}$ that satisfies its norm $||\omega|| < 1$ and thus is a point strictly inside the unit sphere $\mathcal{S}^D$. For a point $z \in \mathcal{S}^D$, Möbius transformation f_ω is defined as:

$$f_\omega(z) = \frac{1 - \|\omega\|^2}{\|z - \omega\|}(z - \omega) - \omega \tag{3}$$

This transformation has a very straightforward geometric meaning: connect z and ω with a straight line, which intersects with $\mathcal{S}^D$ at z', and the Möbius transformation then maps z to $-z'$(see illustration of transformation on $\mathrm{SO}(3)$ in Figure 2.(a). Note that when ω is at the origin, the transformation f_ω becomes the identity transformation; when ω is not at the origin, the transformation f_ω concentrates the part away from ω; and when ω is very close to the surface of unit sphere S^D, the transformation maps almost all points on $\mathcal{S}^D$ to around $-\omega$.

Möbius Coupling Layer A 3×3 rotation matrix $R \in \mathrm{SO}(3)$ satisfies $RR^T = I$ and $\det R = +1$. It thus can be expressed as three orthonormal vectors $[c_1, c_2, c_3]$ that satisfy $||c_i|| = 1$ (or, in other words, $c_i \in \mathcal{S}^2$) and $c_i \cdot c_j = 0$ for all $i \neq j$.

To build a normalizing flow on $\mathrm{SO}(3)$, we thus consider to apply the idea of *coupling* layer introduced in (Dinh et al., 2014),(Dinh et al., 2016) to the orthonormal vectors. In each *coupling* layer, the input is divided into a condition part that remains unchanged after the flow and a transformed part that may change according to the condition part.



(a) Möbius coupling for rotation matrix (b) Scaling and normalization for quaternion

Figure 2: **Illustration of Möbius transformation on rotation matrix and Scaling and Normalization on quaternion**. (a). Möbius coupling layer given c_1 as condition and drawn on the plane P perpendicular to c_1. c_1 is unchanged, c_2 is transformed to the opposite of intersection point of line of c_2 and ω, and $c_3^{'}$ is computed via cross product of c_1 and c_2. (b). Quaternion affine transformation. Point $q \in \mathcal{S}^3$ is first scaled to point $\tilde{\mathbf{q}}$ on an ellipsoid, and then projected back to $q' \in \mathcal{S}^3$, which is the intersection point of the line connecting $\tilde{\mathbf{q}}$ and the origin O and the unit sphere.

We utilize similar structure to build Möbius transformation for rotation matrices. We divide a rotation matrix into 2 parts with the condition part to be c_i ($i = 1, 2, 3$) and the transform parts to be the rest two columns $\{c_j \mid j \neq i\}$. Take $i = 1$ as an example: conditioning on c_1, we can transform c_2 to c_2'. Then c_3 is already determined by $c_3 = c_1 \times c_2$. The coupling layer needs to ensure that: 1) $||c_2'|| = 1$, *i.e.* $c_2' \in \mathcal{S}^2$; and 2) c_2' is orthogonal to c_1.

Given the condition 1), we thus consider to use a Möbius tranformation on $\mathcal{S}^2$ to transform c_2. To further meet the condition 2), we notice that all valid c_2 and c_3 forms a plane P that passes the origin and c_1 is perpendicular to P. After the transformation, c_2' need to stay in P. This can be achieved by constraining ω inside P. Therefore, we propose to learn a neural network that maps the condition c_1 to $\mathbb{R}^3$ and then projects it to P, as shown below:

$$\omega = \omega' - c_1(c_1 \cdot \omega') \tag{4}$$

where ω' is the unconstrained parameters generated by the neural network.

Note that, given c_1, there is only 1 DoF left for the rest two columns. So, our Möbius coupling layer is essentially rotating c_2 and c_3 about c_1 simultaneously by an angle $\theta \in [-\pi, \pi)$ conditioned on c_1.

Linear combination of multiple coupling layers. To further increase the expressivity of the Möbius transformation, Rezende et al. (2020) leveraged linear combination of the transformation, *i.e.*, the weighted sum of the angles $\{\theta_i\}$. We adapted this idea to our Möbius Coupling Layer and the weights are conditioned on c_1 and generated by a learnable neural network conditioned on c_1 and a softmax.

However, such naive implementation has singularities. Take 2 combination point with weights [0.5, 0.5] for example. Assume θ_1 is 30°, θ_2 is $-178°$, the combined angle θ is $-74°$. However, when θ_2 slightly changed to $-182°$ that is 178° as $\theta \in [-\pi, \pi]$, the combined angle θ becomes 104°. This discontinuity of slight change of θ_i resulting in huge jump in combined θ when θ_i is very near to $-\pi, \pi$ may reduce the networks' performance and add difficulties in learning.

We alleviate such discontinuity by restricting ω within $\sqrt{2}/2$ sphere. With this constriction, θ is within $(-\pi/2, \pi/2)$. See supplementary for the proof.

Note that although it is impossible to calculate the inverse of this combination analytically, Rezende et al. (2020) proposed to find the inverse using binary search as c_2 and c_2' always rotate in the same direction. Though the restriction on the norm of ω may reduce the expressivity of our flows, the avoidance of discontinuity stabilizes our network so in general it is beneficial.

3.2 Quaternion Affine Transformation

Kingma & Dhariwal (2018) proposed to use 1×1 convolution before each coupling layer for flexible permutation, as it can decompose the Euclidean space into the condition part and the transform part

flexibly. In Möbius Coupling Layer on $\mathrm{SO}(3)$ this process is also needed as using one of only $[c_1, c_2, c_3]$ as condition part is not flexible enough.

Inspired by the 1×1 convolution in Euclidean space, we propose a quaternion affine transformation, which consists of a linear transformation of 4D vectors $\mathbf{q}$ followed by a projection to the unit sphere $\mathcal{S}^3$. The explicit expression for this transformation is as follows:

$$g(q) = \frac{W\mathbf{q}}{\|W\mathbf{q}\|} \tag{5}$$

where W is a 4×4 invertible matrix, $\mathbf{q}$ is the quaternion representation of a rotation. The inverse of the transformation is simplely by replace W by its inverse matrix W^{-1}. The transformation looks similar to the 1×1 convolution in Glow(Kingma & Dhariwal, 2018), in which an invertible matrix is multiplied on picture channels as a generalization of permutation in the split of the coupling layers, and we have a geometry explanation for its effect.

Geometric explanation of affine. We name after this transformation *affine*, since it resembles the affine transformation in Euclean space $f = ax + b$, where a is the scaling parameter, and b is a displacement term. By SVD decomposition, the 4×4 invertible matrix can be decomposed into an orthogonal matrix U, a diagnose matrix S and another orthogonal matrix V. Multiplying U, V globally rotates the distribution pattern and acts as a displacement on $\mathrm{SO}(3)$ manifold, while the diagnal matrix S serves as the scaling term.

As multiplying an orthogonal matrix won't change the length of a vector, the term $\|W\mathbf{q}\|$ is equal to $\|S(V\mathbf{q})\|$, so affine transformation can be decomposed into:

$$g(\mathbf{q}) = U\frac{S}{\|S(V\mathbf{q})\|}(V\mathbf{q}) \tag{6}$$

This can be seen as a 4-step transformation, first a rotation from $\mathbf{q}$ to $\mathbf{Vq}$, followed by multiplying each coordinate by scaling factors $s_i, (i = 1, \cdots, 4)$, a normalization to a unit vector, and finally followed by a rotation U.

Why rotation U is needed?. Our Möbius coupling layer allows a distribution to flow in the vertical plane of c_i, however it is very difficult to learn a global rotation of distributions on $\mathrm{SO}(3)$. The introduced rotation operation in quaternion space exactly alleviates this problem. Rotating quaternions thus serves a generalization of permutation in splitting condition or transformed columns of rotation matrix, which has similar effects to the 1×1 Convolution introduced in Glow (Kingma & Dhariwal, 2018). Leveraging this rotation, we can choose arbitrary axis to be the condition, not limited by c_1, c_2, and c_3.

We choose to implement this rotation operation in quaternion space rather than directly in $\mathrm{SO}(3)$, because it is naturally covered by quaternion affine transformation, and also, parametrize a 3×3 rotation matrix is harder than a 4×4 invertible matrix. In practice, we represent 4×4 invertible matrix by a unconstrained 4×4 matrix as the probability for a non-invertible matrix is near-zero.

Why scaling and normalization?. Multiplying diagonal matrix S results in multiplying coordinate of a quaternion $q = (w, x, y, z)$ by scaling s_1, s_2, s_3, s_3,

$$(w, x, y, z) \rightarrow (s_1 w, s_2 x, s_3 y, s_4 z) \tag{7}$$

transformating the unit sphere to an ellipsoid, as shown in Figure.2(b)(a sketch viewed in 2D sections for 4D quaternion, and as an illustration, we set $s_1 = 2, s_2 = 1, s_3 = 1, s_4 = 1$), a point on the sphere is transformed to a point on the oval B. The transformation is followed by a projection to sphere: term $1/\|Sq\|$ normalizes the transformed vector. The final point is the intersected point of OB on the sphere D.

When $s_1 = s_2 = s_3 = s_4$, the Scaling and Normalization transformation is the identity. When s_i are not equivalent, it expands areas around the axis with small s and contracts the others. As the combination in Möbius Flow requires constraint on the length of ω, it is not able to create a high peak with Möbius Flow, so we need this scaling as the diagonal elements of S can be arbitrarily high. Note that this process also has the effect of changing the way of decomposition in a different way from the rotation on quaternion.

$$f_{(s_1,s_2,s_3,s_4)}(w, x, y, z) = \frac{(s_1 w, s_2 x, s_3 y, s_4 z)}{\sqrt{(s_1 w)^2 + (s_2 x)^2 + (s_3 y)^2 + (s_4 z)^2}} \tag{8}$$

One advantage of the transformation is its inverse is tractable and easy to be implemented. The inverse of $f(s_1, s_2, s_3, s_4)$ is just $f(1/s_1, 1/s_2, 1/s_3, 1/s_4)$.

3.3 Interchange of Rotation Matrix and Quaternion Representation

Quaternion is another representation of rotation, defined as a unit vector on the 4D sphere. Let θ be the angle of the rotation and (x, y, z) be the axis of rotation, then q can be computed as $(\cos\frac{\theta}{2}, x\sin\frac{\theta}{2}, y\sin\frac{\theta}{2}, z\sin\frac{\theta}{2})$.

As we choose 3×3 rotation matrix as our representation for $\mathrm{SO}(3)$, to implement affine transformation we need first convert a rotation matrix to the quaternion and then convert a quaternion to the rotation matrix after the transformation. However, as $\mathrm{SO}(3)$ is diffeomorphic to the real projective space RP^3, quaternions have the topology of anti-polar symmetry, that is, $q, -q$ represent the same rotation. We mathematically extend the $\mathrm{SO}(3)$ space to the whole sphere of quaternion, and define the distribution $p_R(R) = p_q(q) = p_q(-q)$ invariant under anti-polar operation.

As our affine transformation is equivariant to anti-polar operation,

$$g(-q) = -g(q) \tag{9}$$

the transformed distribution $p_{q'}(q')$ is still invariant under anti-polar operation (see Köhler et al. (2020)) for more details). The transformed quaternion space maintains the anti-polar symmetry and is diffeomorphic to the $\mathrm{SO}(3)$ manifold. The corresponding R' is the transformed rotation matrix. In practice, no matter which convention(q or $-q$) we make to implement the conversion of rotation matrix to the quaternion, transformation from R to R' is a diffeomorphism and suffers no discontinuity by change of representation.

Please see the Supplementary Material for calculations of log determinant, proof of $\frac{\sqrt{2}}{2}$ choice in Möbius combination, visualizations, implementation details.

3.4 Application

Sample and Entropy. One outstanding feature of our Normalizing Flows compared to other probability inference methods on $\mathrm{SO}(3)$ (like Murphy et al. (2021)) is its ability for efficient samples. Murphy et al. (2021) can only sample by querying a large amount of rotations and calculating the probability density function. For highly peaked distribution, this method may fail as it is hard to have queries that are enough close to the peak such that the probability is not close to zero. However, we can sample by transforming z sampled from a base distribution (we choose uniform distribution) through our flows. Efficient sampling makes it possible to estimate properties of data x, for example, entropy can be estimated via Monte Carlo:

$$S = E[\log p(x)]. \tag{10}$$

Conditional Normalizing Flow. There are cases when we need to infer a distribution depending on condition, for example when we need to infer the rotation of a symmetric or occluded object in an image. Our flow can be easily extended to be conditional using methods in Winkler et al. (2019) as we can simply concatenate the condition with fixed columns to generate parameters for Möbius flow. As different target distributions is needed when different condition is given often have a similar pattern, we add a conditional Affine flow at the end of our flow while remaining the former Affine flow unconditional to ensure that the decomposition in Möbius flow is fixed. We show an example of our conditional normalizing flow on learning multi-mode rotation distribution from symmetric object images (see Experiments for detail). The conditional features can be computed by a ResNet or other CNN network in this task.

4 Experiments

In this section, we conduct multiple experiments to validate the capacity of our proposed normalizing flows to model distributions on $\mathrm{SO}(3)$. We train all experiments with negative log-likelihood (NLL) loss.

Table 1: Comparisons on learning to fit various distributions. We adopt log-likelihood as the evaluation metric (higher is better).

	Avg.	Sharp	Cube-like	Cone-like	Head pose
Mathieu & Nickel (2020)	7.18	13.47	1.02	**8.82**	5.41
Falorsi et al. (2019)	-	Failed	3.27	5.32	4.74
Murphy et al. (2021)	7.27	10.52	4.52	8.36	5.71
Mixture matrix Fisher	5.47	7.30	4.33	4.75	5.51
Ours	**8.72**	**13.80**	**4.68**	**8.82**	**7.48**

4.1 Learning to model various distributions

As in common, we first evaluate and compare our normalizing flows and baseline methods by learning to fit distributions on $\mathrm{SO}(3)$ with distinct properties.

Datasets We design three challenging distributions on $\mathrm{SO}(3)$ manifold: a very sharp single modal (matrix Fisher) distribution, a 24-peaked multi-modal distribution and a cyclic distribution. The 24-peaked distribution and the cyclic distribution are designed to simulate the symmetry property of *cube* and *cone* solids. We also include the distribution of real-world *head pose* data from UP-3D (Lassner et al., 2017), 3DPW (Von Marcard et al., 2018) and Human3.6M (Ionescu et al., 2013) datasets. We adopt the visualization tool of Murphy et al. (2021) and show the target distributions as follows.

Baselines Falorsi et al. (2019) introduces the reparameterization trick for Lie groups and allows for constructing flows on the Lie algebra of $\mathrm{SO}(3)$. Mathieu & Nickel (2020) proposes continuous normalizing flows on Riemennian manifold, and we apply it to $\mathrm{SO}(3)$ manifold. Murphy et al. (2021) models the distribution implicit by the neural networks, where the $\mathrm{SO}(3)$ space is uniformly discretized. Finally, we compare to the mixture of matrix Fisher distribution with 500 components.

Results The results are reported in Tab. 1, where our method consistently achieves state-of-the-art performance among all baselines, demonstrating the ability of our method to fit arbitrary distributions in various shapes. In our experiment, baseline Falorsi et al. (2019) fails to fit the sharp distribution around identity due to its singularity around that point.

4.2 Rotation regression with conditional normalizing flows

Modeling arbitrary distributions on $\mathrm{SO}(3)$ manifold well handles the rotation regression task for symmetric objects with multiple (or infinite) correct poses. In this experiment, we leverage our normalizing flows for conditional rotation regression tasks given a single image.

Datasets We experiment on SYMSOL dataset introduced by Murphy et al. (2021). SYMSOL I dataset contains images with solids with high order of symmetry, e.g., tetrahedron, cube, cone, cylinder, which challenges probabilistic approaches to learn complex pose distributions. SYMSOL II dataset includes solids with small markers to break the regular symmetries. We follow the experiment settings of Murphy et al. (2021).

Baselines We compare our method to Implicit-PDF Murphy et al. (2021) as well as several works which parameterize multimodal distributions on $\mathrm{SO}(3)$ for the purpose of pose estimation, including von-Mises distribution Prokudin et al. (2018) and Bingham distribution Deng et al. (2022); Gilitschenski et al. (2019). We quote numbers of baselines from Murphy et al. (2021).

Results The log-likelihood scores are reported in Table 2. We can see from the table that on both SYMSOL I and II datasets, our proposed rotation normalizing flows obtain a significant and consistent performance improvement over all the baselines. We further evaluate our method under *spread* metric in Tab. 3.

4.3 Entropy estimation of arbitrary distributions

As a bijection of source distribution to the target distribution, normalizing flows is able to effectively sample from the target distribution with known probability density of the samples, which enables the

Table 2: Results of rotation regression with conditional normalizing flows on SYMSOL I and II. We adopt log-likelihood as the evaluation metric. Note that we use the convention that a minimally informative uniform distribution has an average log likelihood of 0, which is different from IPDF's convention of -2.29.

	SYMSOL I (log likelihood ↑)						SYMSOL II (log likelihood ↑)			
	avg.	cone	cyl.	tet.	cube	ico.	avg.	sphX	cylO	tetX
Deng et al. (2022)	0.81	2.45	1.34	2.56	-2.15	-0.16	4.86	3.41	5.28	5.90
Gilitschenski et al. (2019)	1.86	6.13	3.17	0.00	0.00	0.00	5.99	5.61	7.17	5.19
Prokudin et al. (2018)	0.42	-1.05	1.01	0.43	1.79	-0.10	2.77	-1.90	6.45	3.77
Murphy et al. (2021)	6.39	6.74	6.55	7.99	7.10	3.57	9.86	9.59	9.20	10.78
Ours	**8.42**	**9.24**	**8.74**	**10.46**	**9.21**	**4.47**	**12.23**	**11.03**	**12.82**	**12.83**

Table 3: Spread estimation on SYMSOL. This metric evaluates how closely the probability mass is centered on any of the equivalent ground truths. We follow Murphy et al. (2021) to evaluate it on SYMSOL I, where all ground truths are known at test time. Values are in degrees.

	cone	cyl.	cube	ico.	tet.
Deng et al. (2022)	10.1	15.2	16.7	40.7	29.5
Murphy et al. (2021)	1.4	1.4	4.6	**4.0**	8.4
Ours	**0.7**	**0.6**	**1.4**	16.7	**1.2**

computation of *entropy* of the target distribution. In this experiment, we compare our rotation NFs with Implicit-PDF in approximating the entropy of the target distributions. In order to obtain the ground truth entropy for evaluation, we adopt multiple matrix Fisher distributions (whose entropy can be analytically computed) with different parameters as the target distributions. We sample 600k points from each target distribution as the training data and evaluate both our method and Implicit-PDF by randomly sampling N (N=5, 10, 100, 1k, 10k) points from the learned distributions. The results are shown in Fig. 3. We can see that even when the sampling size is small, our rotation normalizing flows still achieve accurate estimation of entropy for different target distributions, while Implicit PDF fails to do so.

4.4 Ablation Study

In this experiment, we evaluate the effectiveness of each proposed component in our rotation normalizing flows. We conduct experiments with the same experiment settings as in Sec. 4.1 and the results are reported in Table 4. As shown in the table, Möbius transformation is of crucial importance especially for modeling the cube-like multimodal distributions, and affine transformation generally does good for the overall performance.

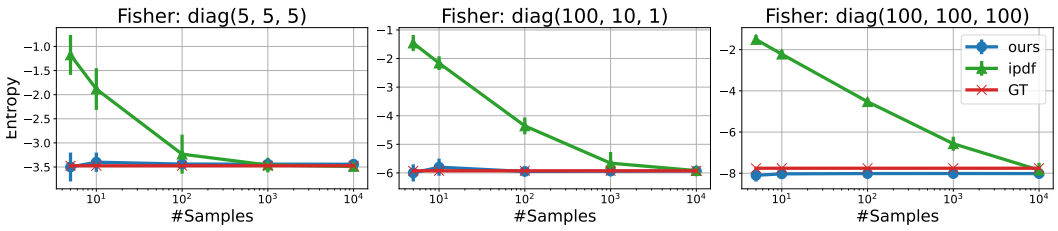


Figure 3: Results of entropy estimation for three different target distributions. We compare the mean and variance of estimated entropy after 100 times of sampling. Note that the horizontal axis is in log scale.

Table 4: Ablation study on the experiments learning to fit various distributions. We adopt log-likelihood as the evaluation metric (higher is better).

	Sharp	Cube-like	Cone-like	Head pose
Mobius Transformation	13.56	4.48	8.76	**5.55**
Affine Transformation	13.50	-0.001	**8.84**	5.32
Mobius + Affine (Ours)	**13.80**	**4.68**	8.82	**5.55**

5 Related Work

Normalizing flows on Euclidean space Most of the Normalizing flows are constructed in Euclidean space. Many of them are constructed using coupling layers(Dinh et al., 2014; 2016), and Glow(Kingma & Dhariwal, 2018) improved it using 1×1 convolution for flexible permutation. Flow++(Ho et al., 2019) combines multiple cumulative distribution functions to make the transformation in the coupling layer more expressive. Invertible ResNet(Behrmann et al., 2019) and Residual Flow(Chen et al., 2019) proposed residual flow which is more flexible and is also possible to be extended to $\mathrm{SO}(3)$. Neural ODE(Chen et al., 2018) and FFJORD(Grathwohl et al., 2018) treated the transformation as a continuous movement of the vectors and use an ordinary differential equation (ODE) to parameterize it, and RNODE(Finlay et al., 2020) further improved it by adding constraints to make the dynamics smoother.

Normalizing flows on non-Euclidean space ReLie (Falorsi et al., 2019) proposed a method to construct normalizing flow on Lie groups by firstly using normalizing flow on Euclidean space and then mapping it onto the target Lie group using the exponential map. However, this method suffers from discontinuity and is unstable and even intractable if some special rotations are inputted. Rezende et al. (2020) proposed three methods to construct flow on tori and sphere: Möbius transformation, Circular splines and Non-compact projection, and here we use Möbius transformation to construct flow on $\mathrm{SO}(3)$. Also, Mathieu & Nickel (2020), Lou et al. (2020) and Falorsi & Forré (2020) extended continuous normalizing flow to Riemann manifold by parameterizing the "velocity" in the tangent space, so their method can also be used in $\mathrm{SO}(3)$.

Distributions for rotation Several works leverage probabilistic distributions on $\mathrm{SO}(3)$ for the purpose of rotation regression. Prokudin et al. (2018) uses the mixture of von Mises distributions over Euler angles. Gilitschenski et al. (2019) and Deng et al. (2022) utilize Bingham distribution over quaternion to jointly estimate a distribution over all axes. They also extend to the mixture of Bingham distributions. Mohlin et al. (2020) leverages matrix Fisher distribution for deep rotation regression with unconstrained Euclidean parameters. Different from the parametric distributions above, Murphy et al. (2021) proposes an implicit pdf over $\mathrm{SO}(3)$ modeled by the parameters of the neural network. It discretizes the $\mathrm{SO}(3)$ space into equivolumetric grids and predicts a pdf for each grid. In this work, our rotation normalizing flow is capable of constructing an arbitrary distribution on $\mathrm{SO}(3)$, with the benefit of efficient sampling.

6 Conclusion

In this work, we show the capability of our proposed novel discrete normalizing flows for rotations to learn various kinds of distributions on $\mathcal{S}^3$. The proposed flow is numerically stable and very expressive, thanks to the complementary role of our proposed Mobius coupling layer and quaternion affine transformation. Our extensive experiments demonstrate that our flow is able to fit complex distributions on $\mathrm{SO}(3)$ and achieves the best performance in both unconditional and conditional tasks.

REFERENCES

Jens Behrmann, Will Grathwohl, Ricky TQ Chen, David Duvenaud, and Jörn-Henrik Jacobsen. Invertible residual networks. In *International Conference on Machine Learning*, pp. 573–582. PMLR, 2019.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Ricky TQ Chen, Jens Behrmann, David K Duvenaud, and Jörn-Henrik Jacobsen. Residual flows for invertible generative modeling. *Advances in Neural Information Processing Systems*, 32, 2019.

Travers Ching, Daniel S Himmelstein, Brett K Beaulieu-Jones, Alexandr A Kalinin, Brian T Do, Gregory P Way, Enrico Ferrero, Paul-Michael Agapow, Michael Zietz, Michael M Hoffman, et al. Opportunities and obstacles for deep learning in biology and medicine. *Journal of The Royal Society Interface*, 15(141):20170387, 2018.

John L Crassidis and F Landis Markley. Unscented filtering for spacecraft attitude estimation. *Journal of guidance, control, and dynamics*, 26(4):536–542, 2003.

Haowen Deng, Mai Bui, Nassir Navab, Leonidas Guibas, Slobodan Ilic, and Tolga Birdal. Deep bingham networks: Dealing with uncertainty and ambiguity in pose estimation. *International Journal of Computer Vision*, pp. 1–28, 2022.

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Luca Falorsi and Patrick Forré. Neural ordinary differential equations on manifolds. *arXiv preprint arXiv:2006.06663*, 2020.

Luca Falorsi, Pim de Haan, Tim R Davidson, and Patrick Forré. Reparameterizing distributions on lie groups. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 3244–3253. PMLR, 2019.

Chris Finlay, Jörn-Henrik Jacobsen, Levon Nurbekyan, and Adam M Oberman. How to train your neural ode. *arXiv preprint arXiv:2002.02798*, 2020.

Mevlana C Gemici, Danilo Rezende, and Shakir Mohamed. Normalizing flows on riemannian manifolds. *arXiv preprint arXiv:1611.02304*, 2016.

Igor Gilitschenski, Roshni Sahoo, Wilko Schwarting, Alexander Amini, Sertac Karaman, and Daniela Rus. Deep orientation uncertainty learning based on a bingham loss. In *International Conference on Learning Representations*, 2019.

Jared Glover, Gary Bradski, and Radu Bogdan Rusu. Monte carlo pose estimation with quaternion kernels and the bingham distribution. In *Robotics: science and systems*, volume 7, pp. 97, 2012.

Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pp. 2722–2730. PMLR, 2019.

Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

Jonas Köhler, Leon Klein, and Frank Noé. Equivariant flows: exact likelihood generative learning for symmetric densities. In *International conference on machine learning*, pp. 5361–5370. PMLR, 2020.

Christoph Lassner, Javier Romero, Martin Kiefel, Federica Bogo, Michael J Black, and Peter V Gehler. Unite the people: Closing the loop between 3d and 2d human representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6050–6059, 2017.

Christian Leibig, Vaneeda Allken, Murat Seçkin Ayhan, Philipp Berens, and Siegfried Wahl. Leveraging uncertainty information from deep neural networks for disease detection. *Scientific reports*, 7(1):1–14, 2017.

Aaron Lou, Derek Lim, Isay Katsman, Leo Huang, Qingxuan Jiang, Ser Nam Lim, and Christopher M De Sa. Neural manifold ordinary differential equations. *Advances in Neural Information Processing Systems*, 33:17548–17558, 2020.

Emile Mathieu and Maximilian Nickel. Riemannian continuous normalizing flows. *Advances in Neural Information Processing Systems*, 33:2503–2515, 2020.

David Mohlin, Josephine Sullivan, and Gérald Bianchi. Probabilistic orientation estimation with matrix fisher distributions. *Advances in Neural Information Processing Systems*, 33:4884–4893, 2020.

Kieran Murphy, Carlos Esteves, Varun Jampani, Srikumar Ramalingam, and Ameesh Makadia. Implicit-pdf: Non-parametric representation of probability distributions on the rotation manifold. *arXiv preprint arXiv:2106.05965*, 2021.

George Papamakarios, Eric T Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *J. Mach. Learn. Res.*, 22(57):1–64, 2021.

Sergey Prokudin, Peter Gehler, and Sebastian Nowozin. Deep directional statistics: Pose estimation with uncertainty quantification. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 534–551, 2018.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pp. 1530–1538. PMLR, 2015.

Danilo Jimenez Rezende, George Papamakarios, Sébastien Racaniere, Michael Albergo, Gurtej Kanwar, Phiala Shanahan, and Kyle Cranmer. Normalizing flows on tori and spheres. In *International Conference on Machine Learning*, pp. 8083–8092. PMLR, 2020.

Torsten Sattler, Qunjie Zhou, Marc Pollefeys, and Laura Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3302–3312, 2019.

Gesina Schwalbe and Martin Schels. A survey on methods for the safety assurance of machine learning based systems. In *10th European Congress on Embedded Real Time Software and Systems (ERTS 2020)*, 2020.

Timo Von Marcard, Roberto Henschel, Michael J Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 601–617, 2018.

Christina Winkler, Daniel Worrall, Emiel Hoogeboom, and Max Welling. Learning likelihoods with conditional normalizing flows. *arXiv preprint arXiv:1912.00042*, 2019.

Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5738–5746, 2019. doi: 10.1109/CVPR.2019.00589.

A CALCULATIONS OF DETERMINATE OF JACOBIAN

A.1 MÖBIUS COUPLING LAYERS

The explicit expression for the forward of Möbius Transformation is as follows:

$$f_\omega(z) = \frac{1 - \|\omega\|^2}{\|z - \omega\|^2}(z - \omega) - \omega. \tag{11}$$

The Jacobian matrix which is the partial differential of z to f_ω is:

$$\frac{\mathbf{d}f_\omega(z)}{\mathbf{d}z} = \frac{1 - \|\omega\|^2}{\|z - \omega\|^2}(I - 2\frac{(z - \omega)^T(z - \omega)}{\|z - \omega\|^2}) \tag{12}$$

where I is the $3\times$ identity matrix.

As the transformation is only 1 DoF, variable z can be expressed by an angle θ to c_2, through:

$$z = \cos\theta c_2 + \sin\theta c_1 \tag{13}$$

The differential of z to θ is given by:

$$\frac{\mathbf{d}z}{\mathbf{d}\theta} = -\sin\theta c_2 + \cos\theta c_3 \tag{14}$$

The differential of $f_\omega(z)$ to θ is given by the change of variable formula:

$$\frac{\mathbf{d}f_\omega(z)}{\mathbf{d}\theta} = \frac{\mathbf{d}f_\omega(z)}{\mathbf{d}z}\frac{\mathbf{d}z}{\mathbf{d}\theta} \tag{15}$$

As $\mathbf{d}f_\omega(z)$ is also a unit point on sphere and have 1 DoF, it can be converted to θ'. Therefore the determinate of Jacobian is given by:

$$|\det J| = |\frac{\mathbf{d}\theta'}{\mathbf{d}\theta}| = \|\frac{f_\omega(z)}{\mathbf{d}\theta}\| \tag{16}$$

B CALCULATIONS FOR AFFINE TRANSFORMATION

The forward and inverse of affine transformation is given by:

$$\mathbf{q}' = \frac{\mathbf{W}\mathbf{q}}{\|\mathbf{W}\mathbf{q}\|} \tag{17}$$

The determinate of Jacobian of affine transformation is very straightforward and is by:

$$\det J(\mathbf{q}) = \frac{\det \mathbf{W}}{\|\mathbf{W}\mathbf{q}\|^4} \tag{18}$$

C PROOF OF THE NEED TO CONSTRAINT $\|\omega\|$

Although the naive implementation of the combination of Möbius transformation has singularities, this problem can be solved by simply restricting ω within the $\frac{\sqrt{2}}{2}$ sphere, which is shown in fig.4. (c). The solution can be solved as follows:

If we constraint $\|\omega\|$ to be less than 1, then the intersection point of line $c_2\omega$ and the unit circle won't be close to c_2, so c_2' won't be close to $-c_2$, and if we keep decreasing $\|\omega\|$, then the upper bound of the degree between c_2 and c_2' will also decrease. If we constraint the upper bound of the degree to be less than $\frac{\pi}{2}$, which means to constraint $\|\omega\|$ to be less than $\frac{\sqrt{2}}{2}$, then we can constraint c_2' to be within one semi-circle and the weighted sum can be calculated without singularity.

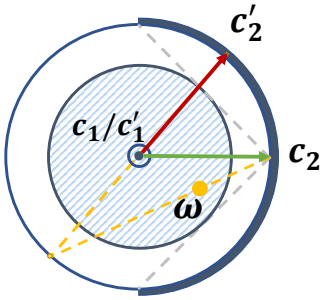


Figure 4: Proof of the Need to Constraint $\|\omega\|$. It can be shown that by constraining $\|\omega\|$, c_2' would be within in the same semi-circle with c_2 and thus the weighted sum is able to be calculated without singularity.

D Implementation Details

We use one multi-layer perceptron (MLP) with 4 hidden layers, 64 hidden units, ReLU as activation function and a residual connection between the first and the last layer in each Möbius Coupling Layer to compute ω with the condition part as input.

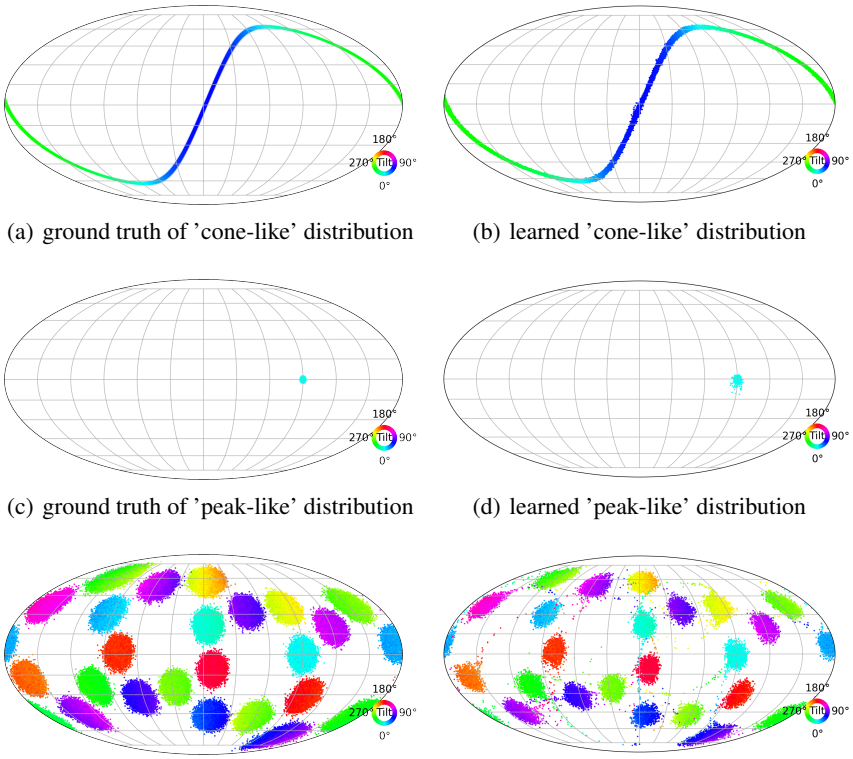
The parameters in Affine Flow is optimized directly with MLP's parameters using Adam with a learning rate of 1e-4.

In unconditional experiments we use 48 layers of flow with 64 segments and a batch size of 256 for 25k steps. In experiments on SYMSOL we use 42 layers of flow with 64 segments and a batch size of 128 for 600k iterations on SYMSOLI and 400k iterations on SYMSOLII for each category.

In experiments on SYMSOL we use an ImageNet pre-trained Resnet50 to capture image features following Murphy et al. (2021).

E Visualization

We show the visualization of our learned various distributions in Figure.5. The rotations are generated by first sampling from uniform distributions and transformed to the targeted ones.



(a) ground truth of 'cone-like' distribution (b) learned 'cone-like' distribution

(c) ground truth of 'peak-like' distribution (d) learned 'peak-like' distribution

(e) ground truth of 'cube-like' distribution (f) learned 'cube-like' distribution

Figure 5: **Visualization of Learned distributions**.(a),(b).The 'cone-like' distribution has one circle of equivalent orientations under symmetry. (c),(d). The peak has a very sharp distribution. (e),(f). The 'cube-like' distribution has 24-mode.