Interpolation on the special orthogonal group with high-dimensional Kuramoto model

Aladin Crnkić and Zinaid Kapić
University of Bihać, Faculty of Technical Engineering, Bosnia and Herzegovina

E-mail: aladin.crnkic@unbi.ba

Abstract. The construction of smooth interpolation trajectories in different non-Euclidean spaces finds application in robotics, computer graphics, and many other engineering fields. This paper proposes a method for generating interpolation trajectories on the special orthogonal group SO(3), called the rotation group. Our method is based on a high-dimensional generalization of the Kuramoto model which is a well-known mathematical description of self-organization in large populations of coupled oscillators. We present the method through several simulations and visualize each simulation as trajectories on unit spheres $𝕊^2$. In addition, we applied our method to the specific problem of object rotation interpolation.

1. Introduction

3D rotations find usage in many different applications in robotics, computer vision, computer graphics, etc. Different representations of 3D rotations, notably rotational matrices, Euler angles and unit quaternions, are used for solving different engineering problems [1]. For our purpose in this paper, we use rotational SO(3) matrices and Euler angles as its parameterization and explain it in the second section in more detail. The most common problems when dealing with rotation data are rotation averaging and interpolation. Rotation averaging is extensively studied in many papers and finds different applications in the field of structure-from-motion [1-2].

Construction of smooth interpolation curves in non-Euclidean spaces finds application in many scientific fields, for example, robotics, computer animation, and computer graphics [1,3]. Problems in robotics such as motion control, obstacle avoidance, and object approach are in constant need of solving rotation interpolation problems [4]. Besides that, it is also applied in computer animations and computer graphics for different use-cases such as smooth interpolation between camera frames [5]. This paper focuses on rotation interpolation between two rotations, where we introduce a new model for rotation interpolation on the group manifold SO(3).

Different approaches are used for interpolation on Lie groups SO(3) and $𝕊^3$. The most common rotation interpolation algorithm on group $𝕊^3$ is a famous Spherical Linear Interpolation algorithm (SLERP) that uses quaternionic representation and gives the shortest and the most direct path between two rotations [6]. Interpolation on unit sphere $𝕊^3$ based on the high-dimensional Kuramoto model is introduced and explained in the paper [7]. For better understanding, it is important to emphasize that a result of the multiplication between two unit quaternions is always a unit quaternion. This characteristic of unit quaternions allows a spherical interpolation between rotations. On the other hand, a result of the multiplication between two rotational SO(3) matrices is not always a new rotational matrix. This characteristic of rotational matrices produces an anomaly in creating an interpolation...
curve because the point in the interpolation trajectory does not always preserve a unit sphere [1]. Common algorithms for rotation interpolation with SO(3) matrices and Euler angles are a so called Linear Matrix Interpolation (LinMat) and Linear Euler Interpolation (LinEul), respectively [8]. These algorithms are subjected to the abovementioned anomaly, and there are several papers that solve this problem by producing spherical interpolation of rotations parameterized with rotational matrices or Euler angles [9-10]. In our paper, we present two novel models, one for rotation interpolation with rotational matrices and other with Euler angles. These models always preserve SO(3) group and give the shortest and the most direct spherical path between two rotations.

Model used in this paper is based on one generalization of a famous Kuramoto model which is the most adequate for studying collective behaviour and self-organization in large populations of coupled oscillators [11]. Through the last decades, Kuramoto model has been subjected to various generalizations on higher dimensions. One of the most studied generalization has been introduced by Lohe and is known as the non-Abelian Kuramoto model, or just Lohe model [12]. Non-Abelian Kuramoto model can be used for clustering of static and stream data [13-14], coordination and consensus on groups $S^3$ and SO(3) [15], rotation averaging [16], and other applications in different scientific and engineering fields [17].

In the next section we explain parameterizations used in this paper and mapping between them. In the section 3 we describe our dynamical model and present our method through an algorithm. Simulations are implemented using programming environment Wolfram Mathematica and illustrated in the third section. Finally, in the last section, we draw some conclusion and present an outlook on future research.

2. Mapping between special orthogonal group SO(3) and Euler angles
As it is mentioned above, different parameterizations can be used for representing rotations. In this paper we use rotational SO(3) matrices and Euler angles. SO(3) is the group of all rotations about the origin in three dimensional Euclidean space. This group is represented by $3 \times 3$ rotation matrices with determinant 1, as it follows [1]:

$$SO(3) := \{ R \in \mathbb{R}^{3x3} | R^T R = R R^T = I_3, \det(R) = +1 \}.$$  (1)

In (1) $I_3$ represents identity $3 \times 3$ matrix, $\det(\cdot)$ represents determinant of the matrix, and $R^T$ is a transpose of the matrix $R$. Besides SO(3) matrices, interesting representation of rotations are so called Euler angles [6]. These angles are easy to visualize and understand. Euler angles are consisted of roll, pitch and yaw, as it is visualized in figure 1.

![Figure 1. Euler angles as parameterization of rotations](image-url)
Roll angle represents rotation about the x-axis and is denoted as $\psi$, pitch angle refers to rotation about y-axis and is denoted as $\theta$ and yaw angle is a rotation about z-axis and is denoted as $\phi$. It is possible to map Euler angles to SO(3) matrices, and vice-versa. Let us assume that a rotation about axis $x$ by an angle $\psi$ is represented as [18]:

$$ R_{x,\psi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \psi & -\sin \psi \\ 0 & \sin \psi & \cos \psi \end{bmatrix}. $$

With the same analogy we represent a rotation about axis $y$ by an angle $\theta$, and a rotation about axis $z$ by an angle $\phi$:

$$ R_{y,\theta} = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix} \quad \text{and} \quad R_{z,\phi} = \begin{bmatrix} \cos \phi & -\sin \phi & 0 \\ \sin \phi & \cos \phi & 0 \\ 0 & 0 & 1 \end{bmatrix}. $$

A rotational matrix can be obtained by performing a rolling about $x$ axis by an angle $\psi$, pitching about $y$ by an angle $\theta$ and yawing about $z$ by an angle $\phi$. In this case, $R_{(\phi,\theta,\psi)} = R_{x,\phi}R_{y,\theta}R_{x,\psi}$, yielding:

$$ R_{(\phi,\theta,\psi)} = \begin{bmatrix} \cos \theta \cos \phi & -\cos \phi \sin \psi + \sin \phi \sin \theta \cos \phi & \sin \phi \sin \psi + \cos \phi \sin \theta \cos \phi \\ \sin \theta \sin \phi & \cos \psi \cos \phi + \sin \psi \sin \theta \sin \phi & -\sin \psi \cos \phi + \cos \psi \sin \theta \sin \phi \\ -\sin \theta \cos \phi & \sin \psi \cos \phi + \cos \psi \sin \theta \sin \phi & \cos \psi \cos \phi \cos \theta \end{bmatrix}. \quad (2) $$

Obtaining Euler angles from a given rotational matrix from (2) one has that roll, pitch and yaw angle are, respectively [18]:

$$ \psi = \arctan \left( \frac{R_{(3,2)}}{R_{(3,3)}} \right), \quad \theta = -\arcsin(R_{31}), \quad \phi = \arctan \left( \frac{R_{(2,1)}}{R_{(1,1)}} \right). $$

Every rotation representation has its own preferred application, advantages, and disadvantages. Rotation matrices are used in computer graphics and computer vision, while Euler angles are mainly used in robotics [1]. The advantages of rotation matrices are their well-known mathematical background and the ability to represent all other transformations such as translation, scaling, and projection. On the other hand, matrix representation is redundant because it uses 9 parameters for rotation representation, while Euler angles use only 3. The biggest drawback for the usage of Euler angles is a Gimbal lock problem. Gimbal lock is a loss of one degree of rotational freedom [8]. Quaternions, that are not subject of research in this paper, are a better solution for interpolation purposes as it is seen in paper [7], but Euler angles are also used because they are simpler to understand and visualize, especially for those unfamiliar with quaternion algebra.

3. Algorithm

In this section we will introduce our method for interpolation on the rotation group SO(3). As we have said our method is based on the non-Abelian Kuramoto model that is defined as:

$$ i\dot{U}_j U_j^* = H_j - \frac{iK}{2N} \sum_{i=1}^{N}(U_j U_i^* - U_i U_j^*), \quad j = 1, \ldots, N. \quad (3) $$

In the abovementioned equation, $N$ is a number of generalized oscillators, $U_j(n) \in U(n)$ is unitary matrix representig the state of generalized oscillator $j$, $H_j(n) \in u(n)$ represents intrinsic frequency of generalized oscillator $j$. The matrix $U_j^*$ is conjugate transpose matrix of the matrix $U_j$. Finally, $K$ is a strength of a global coupling.

Firstly, let us suppose that the rotations are given with SO(3) rotational matrices, where $R_S$ represents a starting rotation and $R_E$ represents the final one.

Consider the following matrix-valued ordinary differential equation which can be easily obtained from the system (3):

$$ R'(t) = R_E - R(t) R_E^T R(t) $$

with initial condition $R(0) = R_S$. In above equation, $R_E^T$ corresponds to a transposed matrix of the matrix $R_E$. It is important to emphasize that (4) defines dynamics on the group SO(3), meaning that
The abovementioned system corresponds to non-degenerate case where $\theta \neq \frac{\pi}{2}$ and $\theta \neq -\frac{\pi}{2}$. For a degenerate case of $\theta = \frac{\pi}{2}$ or $\theta = -\frac{\pi}{2}$ system enters in a so called Gimbal lock.

For a better presentation, we will explain our methods through algorithms. Algorithm for rotation interpolation with rotations given with SO(3) rotational matrices is as follows:

1: Enter $R_S, R_F$
2: Choose tolerance $\varepsilon$, step $\delta$, and define $T=0$
3: Solve (4) with $R(0) = R_S$
4: loop
5: if $\|R_E - R(T)\| < \varepsilon$ then
6: return $R(t)$ for $t \in [0, T]$
7: else
8: $T = T + \delta$
9: end if
10: end loop

Algorithm for rotation interpolation with rotations given with Euler angles is equivalent to the abovementioned algorithm for rotation interpolation with rotations given as SO(3) rotational matrices, but the process is performed by solving a system of differential equations (5) and the initial and final positions are given with Euler angles $\psi, \theta$, and $\phi$.

4. Simulations
In this section we present and illustrate simulation results of our method. It is well known that is difficult to visualize points or curves on SO(3). Therefore, we have visualized three different curves
on unit spheres $S^2$, obtained with $R(t) \cdot v_i \in S^2$ for $i = 1,2,3$, where $v_1 = [1 0 0]^T$, $v_2 = [0 1 0]^T$, $v_3 = [0 0 1]^T$ are unit vectors and $R(t) \in SO(3)$ for all $t > 0$.

For the first simulation we have used initial and final rotation (represented by $SO(3)$ rotational matrices) given as:

$$R_S = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad R_E = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$  

Interpolation curve from a starting rotation $R_s$ to the final rotation $R_E$ is presented in figure 2. It is obtained using the algorithm with model (4), and by setting initial condition $R(0) = R_s$, tolerance $\varepsilon = 10^{-5}$ and step $\delta = 0.01$. Final point is reached at the moment $T = 11.66$ with a set tolerance $\varepsilon$.

![Figure 2](image1.png)

**Figure 2.** Interpolation curve from a starting rotation $R_s$ to the final rotation $R_E$: a) $R(t) \cdot v_1$ for all $t \in [0,T]$; b) $R(t) \cdot v_2$ for all $t \in [0,T]$; c) $R(t) \cdot v_3$ for all $t \in [0,T]$.

For the second simulation we will use Euler parametrization of rotations. Initial and final rotation are given as Euler angles. Let us assume that the initial rotation is given as: $\phi_S = 2\pi / 3$, $\theta_S = 0$, $\psi_S = 0$, and final rotation as: $\phi_E = 0$, $\theta_E = 0$, $\psi_E = \pi / 2$.

In this case we use the algorithm with model (5). Initial conditions for this simulation are as follows: $\psi(0) = \psi_S$, $\theta(0) = \theta_S$, and $\phi(0) = \phi_S$, tolerance is $\varepsilon = 10^{-5}$ and step is $\delta = 0.01$. At the moment $T = 13.1$ the stopping criteria in the algorithms step 4 is satisfied.

For illustration of this simulation result we have simulated an objects interpolation from a starting position $P_s$ to the final position $P_E$ shown in figure 3.

![Figure 3](image2.png)

**Figure 3.** Object positions at specific moments: a) $t = 0$ ($P_s$); b) $t = 0.4$ ($P_1$); c) $t = 1.2$ ($P_2$); d) $T = 13.1$ ($P_E$).
Table 1 shows numeric values of rotations presented using Euler parameterization and their equivalent SO(3) matrices at specific positions given in figure 3.

| Position | SO(3) matrix | Euler angles |
|----------|--------------|--------------|
| **P_S**  | \[
\begin{bmatrix}
-0.5 & -0.866 & 0 \\
0.866 & -0.5 & 0 \\
0 & 0 & 1
\end{bmatrix}
\] | \(\phi = 2\pi/3,\) \(\theta = 0,\) \(\psi = 0.\) |
| **P_1**  | \[
\begin{bmatrix}
0.0026 & -0.934 & 0.358 \\
0.934 & -0.126 & -0.335 \\
0.358 & 0.335 & 0.871
\end{bmatrix}
\] | \(\phi = 1.5681\pi,\) \(\theta = -0.3660\pi,\) \(\psi = 0.3670\pi.\) |
| **P_2**  | \[
\begin{bmatrix}
0.727 & -0.558 & 0.400 \\
0.558 & 0.140 & -0.818 \\
0.400 & 0.818 & 0.413
\end{bmatrix}
\] | \(\phi = 0.6544\pi,\) \(\theta = -0.4118\pi,\) \(\psi = 1.1032\pi.\) |
| **P_E**  | \[
\begin{bmatrix}
1 & 0 & 0 \\
0 & 0 & -1 \\
0 & 1 & 0
\end{bmatrix}
\] | \(\phi = 0,\) \(\theta = 0,\) \(\psi = \pi/2.\) |

Notice that our method produces a spherical interpolation curve. That means that at each moment, model (4) preserves the SO(3) group. For better illustration, we have also provided a video that showcases object's movement from starting to final position.1

5. Conclusion and outlook
In this paper we have introduced a novel model for rotation interpolation between two points using rotational matrices SO(3) and Euler angles as parameterizations. This method is based on high-dimensional generalization of Kuramoto model and offers a most direct and shortest way from initial to final rotation. Simulation results are visualized as a trajectory on a unit sphere and for a better insight we have provided one short video illustration of objects movement.

Approach presented in this paper offers universal solution for rotation interpolation with any given parameterization. This method finds spherical interpolation curve for rotations in SO(3) form or Euler angles, like paper [9] that uses quaternions for these purposes. We find it interesting to integrate all rotation parameterizations into a model that utilizes high-dimensional Kuramoto model for interpolation purposes.

The algorithm proposed in this paper can be used in computer graphics for creating animations and robotics for motion control and robotic hand movement. It is important to mention that the model can be used only for interpolation between two given rotations. Therefore, we find it interesting to upgrade this model for multiple rotations interpolation. This open question will be addressed in our future work.

References
[1] Hartley R, Trumpf J, Dai Y and Li H 2013 Rotation averaging International Journal of Computer Vision 103 267–305 doi: 10.1007/s11263-012-0601-0
[2] Purkait P, Chin T-J and Reid I 2020 NeuRoRA: Neural Robust Rotation Averaging Computer Vision – ECCV 2020 137–54 doi: 10.1007/978-3-030-58586-0_9

1 Supplementary video is available at: https://www.youtube.com/watch?v=bP6C7ZlCC18.
[3] Zhao X, Liu Y and Xie Z 2021 The application of acceleration-level quaternion interpolation in the visual-servoing process 2021 13th International Conference on Computer and Automation Engineering (ICCAE) doi: 10.1109/iccae51876.2021.9426103

[4] Legnani G, Fassi I, Tasora A and Fusiad 2021 A practical algorithm for smooth interpolation between different angular positions Mechanism and Machine Theory 162 104341 doi: 10.1016/j.mechmachtheory.2021.104341

[5] Haarbach A, Birdal T and Ilic S 2018 Survey of higher order rigid body motion interpolation methods for keyframe animation and continuous-time trajectory estimation 2018 International Conference on 3D Vision (3DV) doi: 10.1109/3DV.2018.00051

[6] Shoemake K 1985 Animating rotation with quaternion curves Proceedings of the 12th annual conference on Computer graphics and interactive techniques - SIGGRAPH '85 doi: 10.1145/325334.325242

[7] Kapic Z and Crnkic A 2021 Interpolating rotations with non-Abelian Kuramoto model on the 3-sphere Advanced Technologies, Systems, and Applications VI - Proceedings of the International Symposium on Innovative and Interdisciplinary Applications of Advanced Technologies (IAT 2021) doi: 10.1007/978-3-030-90055-7_48

[8] Dam E, Koch M and Lilholm M 1998 Quaternions, interpolation and animation CiteSeerX

[9] Shingel T 2008 Interpolation in special orthogonal groups IMA Journal of Numerical Analysis 29 731–45 doi: 10.1093/imanum/drm033

[10] Wu J, Liu M, Ding J and Deng M 2019 Robust rotation Interpolation based on So(n) geodesic distance Lecture Notes in Computer Science 126–32 doi: 10.1007/978-3-030-34995-0_12

[11] Kuramoto Y 1975 Self-entrainment of a population of coupled non-linear oscillators International Symposium on Mathematical Problems in Theoretical Physics 420–2 doi: 10.1007/bfb0013365

[12] Lohe M A 2009 Non-Abelian Kuramoto models and synchronization Journal of Physics A: Mathematical and Theoretical 42 395101 doi: 10.1088/1751-8121/42/39/395101

[13] Crnkic A and Jaćimović V 2018 Data clustering based on quantum synchronization Natural Computing 18 907–11 doi: 10.1007/s11107-017-9720-z

[14] Crnkic A, Ivanović I, Jaćimović V and Mijajlović N 2020 Swarms on the 3-sphere for online clustering of multivariate time series and data streams Future Generation Computer Systems 112 11–7 doi: 10.1016/j.future.2020.05.018

[15] Crnkic A, Jaćimović M, Jaćimović V and Mijajlović N 2020 Consensus and coordination on groups SO(3) and S3 over constant and state-dependent communication graphs Automatica 62 76–83 doi: 10.1016/j.automatica.2019.06.018

[16] Kapic Z, Crnkic A, Jacimovic V and Mijajlovic N 2021 A new dynamical model for solving rotation averaging problem 2021 20th International Symposium INFOTEH-JAHORINA (INFOTEH) doi: 10.1109/INFOTEH51037.2021.9400663

[17] Ha S-Y and Park H 2020 A dynamical systems approach for the shape matching of polytopes along rigid-body motions arXiv.org arXiv:2006.11533

[18] Hashim H A 2019 Special orthogonal Group SO(3), Euler Angles, Angle-axis, Rodriguez vector and Unit-quaternion: Overview, mapping and challenges arXiv.org arXiv:1909.06669