A Novel Dual Quaternion based Dynamic Motion Primitives for Acrobatic flight

Renshan Zhang¹, Yongyang Hu¹, Kuang Zhao¹, Su Cao²*
¹Nanjing Telecommunication Technology Research Institute, Nanjing, China
²Institute of Unmanned Systems, National University of Defense Technology, Changsha, China
zhangrenshan_tx@126.com, caosu_93@foxmail.com

Abstract—The realization of motion description is a challenging work for fixed-wing Unmanned Aerial Vehicle (UAV) acrobatic flight, due to the inherent coupling problem in translational-rotational motion. This paper aims to develop a novel maneuver description method through the idea of imitation learning, and there are two main contributions of our work: 1) A dual quaternion based dynamic motion primitives (DQ-DMP) is proposed and the state equations of the position and attitude can be combined without loss of accuracy. 2) An online hardware-in-the-loop (HITL) training system is established. Based on the DQ-DMP method, the geometric features of the demonstrated maneuver can be obtained in real-time, and the stability of the DQ-DMP is theoretically proved. The simulation results illustrate the superiority of the proposed method compared to the traditional position/attitude decoupling method.

Keywords- dual quaternion, DMP, fixed wing acrobatic flight, trajectory learning, imitation learning

I. INTRODUCTION

The acrobatic of UAV is usually accompanied by fast change in position and attitude, which is a relatively complex movement process[1]. Traditional methods are difficult to describe this kind of action. Imitation learning provides a way of thinking, typically the dynamic motion primitive (DMP) can represent complex motion, and guarantee the stability and continuity of the trajectory, which is a scheme worth exploring[2].

Motion primitives theory uses the sequencing of biological systems and the ability to adapt to motion units to explain the execution of complex motions. DMPs originated from the motion control of biological systems, and can be regarded as a rigorous mathematical formula for stable non-linear dynamic systems of motion primitives[3]. DMPs are a trajectory planning method proposed by Stefan Schaal in [4] and updated by Auke Ijspeert in [5]. For another Pastor et al. [6] first extended DMP to rotational motion and proposed quaternion DMP, Ude et al. [7] extended the rotation matrix quaternion DMP, Saveriano et al. [8] used the Lyapunov method to prove the stability of their system. These works extend DMP from \( \mathbb{R} \) space to \( SO(3) \) space.

On the application, DMPs are widely used in robotic arms, humanoid robots, medical enhancement robots, teleoperation robots, as well as UAV and other mobile robots. In the field of drones, Perk and Slotine [9] use DMP to define the flight path and obstacle avoidance of UAV, where the trajectory is generated by the movement of the joystick that controls the UAV. Later, Fang et al. [10] extended the method to encode the drone data demonstrated by the user, extract and encode the linear part of the flight trajectory, and combine them into flight control actions. In addition Tomić et al. [11] formulated the motion of the drone as an optimal control problem, the output of the optimal control solver is encoded using DMP, so that they could apply modifications to the UAV’s flight trajectory in real time. Lee et al. [12] also incorporated DMP into the control scheme to modify the flight trajectory and avoid obstacles in flight. DMP has played an important role in avoiding real-time obstacles.

These works apply DMP to the field of UAV and have made a lot of progress, but they have not considered the coupling of position and attitude. Traditional methods usually decouple position and attitude. This method undermines the integrity of the problem, showing disadvantages and limitations that cannot be balanced. The dual quaternion uses only eight real numbers to describe the motion of a general rigid body, which can concurrently express rotation and translation as well as the coupling relationship between each other, which can realize the integration of pose in the true sense[13]. Based on quaternion DMP, this paper proposes DQ-DMP, expanding DMP to \( SE(3) \) space. Finally, we apply it to describe the complex motion of fixed-wing UAV, and train DQ-DMP through expert teaching data, the simulation results show the feasibility of this method.

II. FORMULATION OF DMPs

In this section, we introduce several theoretical models of DMP. First, we introduce the classic DMP and the quaternion DMP, and then briefly introduce the dual quaternion to provide a basis for the next section of the dual quaternion DMP.

A. Classical DMP

For a typical second-order point attractor system, such as mass spring damping system, which can be expressed using the following equation \( \ddot{y} = \alpha \left[ \beta (g - y) - \dot{y} \right] \), where \( g \) is the end point of the system, \( y \) is the state of the system, \( \alpha \) and \( \beta \) is the gain, the control goal is to make the system reach the specified end point. In order to make the
system move according to the trajectory we expect, the phase variable and forcing term are introduced[5]

\[
\tau \dot{y} = z
\]
\[
\tau \dot{z} = \alpha \left[ \beta_t (g - y) - z \right] + f(x)
\]

where the phase variable is the exponentially decayed clock signal from 1 to 0, obtained by the so-called regular system integration \( \tau x = -\alpha x \), time scale factor \( \tau \) can change the duration of motion, \( f(x) \) is defined as the weighted average of \( N \) Gaussian kernel functions, its function is to change the acceleration of the system at different moments, and drive the system along a free will smooth trajectory from the initial position \( y_o \) to the end point \( g \), where forcing term

\[
f(x) = \sum_{i=1}^{N} \lambda_i \psi_i(x) \]
\[
\psi_i(x) = \exp(-h_i (x - c_i)^2)
\]

where \( c_i \) is the center of Gaussian functions distributed along the motion phase, and \( h_i \) is their width. For a given \( N \), set \( \tau \) equal to the total duration of the desired movement. Generally speaking, we can define parameters

\[
c_i = \exp\left(-\alpha \frac{i-1}{N-1}\right), h_i = \frac{1}{(c_{i+1} - c_i)^2}, h_0 = h_{N-1}.
\]

(3)

For each degree of freedom, the weight \( \lambda_i \) should be adjusted according to the measured data in order to achieve the desired behavior. For a given trajectory \( y_d(t) \) and its derivative \( \dot{y}_d(t) \) and \( \ddot{y}_d(t) \), \( t=0, \Delta t, ..., T \) the required values \( f_d(t) \) can be obtained as follows

\[
f_d(t) = \tau^2 \ddot{y}_d(t) - \alpha \left[ \beta_t (g - y_d(t)) - \tau \dot{y}_d(t) \right].
\]

(4)

We use \( f(x) \) formed by basis function to approach the target driving term, the driving term and weight can be approximated as a linear relationship, that is

\[
f(x) = A \lambda \approx f_d(t).
\]

\[
A = \begin{bmatrix}
\frac{\psi_1(x_1)}{\sum_{i=1}^{N} \psi_i(x_1)} & \cdots & \frac{\psi_N(x_1)}{\sum_{i=1}^{N} \psi_i(x_1)} \\
\vdots & \ddots & \vdots \\
\frac{\psi_1(x_T)}{\sum_{i=1}^{N} \psi_i(x_T)} & \cdots & \frac{\psi_N(x_T)}{\sum_{i=1}^{N} \psi_i(x_T)}
\end{bmatrix}
\]

(5)

It can be obtained that \( \lambda = A^T f_d(t) \), \( A^T \) is the pseudo-inverse of \( A \) and the parameter \( \lambda_i \) can be calculated by the weighted least square method.

It should be noted that the above system defaults choose the inertial coordinate system.

For controlling a robot system with multiple degrees of freedom, we use a different system of equations (1) to express the motion of each degree of freedom, but use a common phase to synchronize them. It can be seen that the dynamic motion primitive is driven by the attractor dynamics differential equation, and represented by a combination of the nonlinear force term and the attractor force term. Nonlinear force can represent complex motion, and attractor force represents the target state. The nonlinear force weakens with time, and finally the attractor force dominates, so the dynamic motion primitive can smoothly converge to the target state. DMPs ensure the smoothness and continuity of the trajectory, and can express nonlinear motion without losing stability.

B. Quaternion DMP

In Cartesian space, attitude can be expressed as a unit quaternion \( q = [\eta, e] \) \( e \in SO(3) \), where \( SO(3) \) is the unit ball of three-dimensional space, and \( \eta \) is scalar part, \( e \) is the vector part of the quaternion, \( \eta^2 + \|e\|^2 = 1 \). Compared with the rotation matrix, the unit quaternion has fewer parameters, and there is no singularity in contrast to Euler angles, so it is frequently used to describe rotation operations in engineering.

The quaternion DMP of the inertial coordinate system is expressed as follows[7]

\[
\tau q = \frac{1}{2} \tilde{\omega} \otimes q
\]

\[
\tau \dot{w} = K^s \left[ e_s (q_d \otimes q^*) - d^s_b (x) + f^s_a (x) \right] - D^s \omega
\]

where \( \omega \in \mathbb{R}^3 \) is the angular velocity, \( \tilde{\omega} \in \mathbb{R}^3 \) is the angular acceleration, \( d^s_b (x) = (q_u - q_v) x \) prevent jumping at the beginning. \( f^s_a (x) \) is obtained by (2). \( \tilde{\omega} = [0, \omega] \) is a quaternion, the scalar part is zero, and the vector part is the angular velocity. The error between the two quaternions \( e_s (\cdot) \) is a nonlinear function, it has multiple definitions, in this work we choose the orientation error between \( q_1 \) and \( q_2 \) as \( e_o = \text{vec} \left( q_1 \otimes q_2^* \right) \) [8], where \( \text{vec}(q) \) back to the vector part of \( q \).

Define the differential of a quaternion

\[
q(t + \Delta t) = \exp \left( \frac{\Delta t}{2} \omega(t) \right) \otimes q(t)
\]

where \( \Delta t \) is the sampling time, so \( q \) can be calculated by the above formula, where the exponential mapping of the quaternion \( \exp(\cdot) : \mathbb{R}^3 \mapsto SO(3) \), a rotation vector is exponentially mapped to a unit quaternion

\[
\exp(r) = \begin{cases}
\left[ \cos(\|r\|), \sin(\|r\|) \frac{r}{\|r\|} \right], & r \neq 0 \\
[1,0,0,0]^T, & \text{otherwise}
\end{cases}
\]

(8)
C. Introduction to dual quaternions

The dual number was invented by Clifford in 1873 and further expanded by Study in 1891. The dual number is defined as
\[ z = a + \epsilon b \quad \text{with} \quad \epsilon^2 = 0, \quad \text{but} \quad \epsilon \neq 0 \]
where \( a \) and \( b \) are real numbers, called the real part and the dual part respectively, and \( \epsilon \) is the nilpotent term.

A dual quaternion \( \mathbf{q} = \mathbf{q}_r + \epsilon \mathbf{q}_p \in \mathbb{R} \times \mathbb{R} \) is a dual number whose elements are quaternion, where \( \mathbf{q}_r \) is the real part, representing rotation, \( \mathbf{q}_p \) is an even part, representing translation, which are all quaternion and \( \epsilon \) is nilpotent term. This method of representation can unify the translation and rotation in one space, the general rigid body motion described by the rotation \( \mathbf{q} \) followed by the translation \( \mathbf{p} \) can be described as \( \mathbf{q} = q + \frac{\epsilon}{2} q \otimes p^b \), where \( p^b \) represents a position in the body coordinate system [13].

The error of the dual quaternion can be expressed as
\[ \tilde{q}_e = q_{se} + \epsilon q_{pe} = q_{se} + \frac{\epsilon}{2} q_{se} \otimes p^b \]
and we can get \( p^b = 2q_{se} \otimes q_{pe} \).

The kinematic equation of dual quaternion is
\[ \dot{\mathbf{q}} = \frac{1}{2} \mathbf{\gamma} \otimes \dot{\mathbf{q}} \]
(10)
where \( p^b \) is the quaternion expanded by adding zero elements at the position, \( \mathbf{\gamma} \) is the dual quaternion expanded by spinor \( \mathbf{\gamma} \in se(3) \).

\[ \mathbf{\gamma} = \mathbf{W} \otimes \mathbf{w} \]
(11)
They are the spinor in the inertial coordinate system and the spinor in the body coordinate system, \( p^b \) and \( p^f \) are the projections of the translation motion contained in the dual quaternion.

III. DUAL QUATERNION BASED DMPs

A. Dual Quaternion DMP

The dual quaternion DMP in the body coordinate system is expressed as follows
\[ \tau\dot{\mathbf{q}} = \frac{1}{2} \dot{\mathbf{q}} \otimes \mathbf{\gamma} \]
(12)
where \( \tau \) is the time scale, \( d_1^b(x) = d_2^b(x) = 0 \) prevent jumping at the beginning, \( f^b(x) \) is defined by (2), \( K^b, D^b \) is linear stiffness gain and damping gain, including attitude part and position part.

For a given trajectory \( \mathbf{q}_d(t) \), the forcing term is
\[ \dot{f}^b(x) = (K^b)^{-1} (\tau^f \frac{z^b}{\tau^f} + \tau D^f z^b) - e_v (\mathbf{q}^* \otimes \dot{\mathbf{q}}_d) + d_1^f(x) \]
(13)
f^b(x) can be used to encode any sampled directional trajectory \( \mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d, \mathbf{r}_d \), and weight \( \lambda \) can be obtained by solving the following linear equations
\[ \sum_{i=1}^{N} \lambda_i \psi_i(x) = \sum_{i=1}^{N} \psi_i(x) \]
(14)
In the calculation process, it is necessary to define the orientation error between the dual quaternions \( \mathbf{q} \) and \( \mathbf{q}_d \), this paper adopts the following logarithmic mapping
\[ \exp{\mathbf{\Phi}(\mathbf{q})} = \mathbf{q}_d \]
(15)
Using kinematic equations, we can define the dual differential quaternion as follows
\[ \dot{\mathbf{q}}(t + \Delta t) = \dot{\mathbf{q}}(t) \exp{\left( \frac{\Delta t}{2} \mathbf{\gamma} \right)} \]
(16)
And set \( \mathbf{\gamma} = [r, v] \), we define the exponential mapping of the spinor as \( \exp{\mathbf{\Phi}(\mathbf{\gamma})} \)
\[ \exp{\mathbf{\Phi}(\mathbf{\gamma})} = \left[ \begin{array}{c} \cos(|r|), \sin(|r|) r^T + \mathbf{e} \end{array} \right] + \epsilon |r|, r \neq 0 \]
(17)
and \( r = 0 \), \( v \).

B. Stability analysis

When \( t \to \infty \), \( d_1^b(x), f^b(x) \) gradually vanish with the \( x \) decay from 1 to 0, the system degenerates to \( \tau \hat{\mathbf{z}} = K^b e_v (\mathbf{\gamma} \otimes \dot{\mathbf{q}}_d) - D^b \mathbf{\gamma} \).

Theorem 1. For the above dual quaternion DMP system, \( x \) is a clock signal with exponential decay from 1 to 0 with time. The error is defined as \( \mathbf{e} = \left[ \mathbf{vec}(\mathbf{q}_w), \mathbf{vec}(p_s) \right] \), the system globally asymptotically converges to \( \dot{\mathbf{q}} = \dot{\mathbf{q}}_d \) with \( \mathbf{\gamma} = 0 \). If the state quantity is selected \( x = [\hat{\mathbf{q}}, \mathbf{\gamma}] \), it is \( x \) close to \( \mathbf{q}_d, 0 \).

Proof: For a unified description, the following derivation selects the body coordinate system, without loss of generality, we set \( \tau = 1 \) and define \( \mathbf{\gamma} = [\omega, v] \).

\[ \hat{\mathbf{z}} = K^b e_v (\mathbf{\gamma} \otimes \dot{\mathbf{q}}_d) - D^b \mathbf{\gamma} \]
(18)
We can get the following formula
\[ \dot{\omega} = K^* \text{vec} (q^*_t \otimes q^*_o) - D^* \omega \]
\[ \dot{\psi} = K^* \text{vec} (p^*_d - p) - D^* \nu \quad (19) \]
If set \( \dot{q}_d = q_o + \epsilon q_o = [\eta_o, \epsilon, \epsilon, \epsilon] + \frac{\epsilon}{2} q_o \otimes p \), we can use the following Lyapunov candidate
\[ V(x) = (\eta_o - \eta) + \| \epsilon - \epsilon \| + \frac{1}{2} \omega^T (K^*)^{-1} \omega \]
\[ + \frac{1}{2} \| p_d - p \|^2 + \frac{1}{2} \nu^T (K^*)^{-1} \nu \quad (20) \]
Considering that the rotations is a fast item and is not affected by the translation, \( V(x) \) will be split as follows
\[ V(x) = V_1(x) + V_2(x) \]
where
\[ V_1(x) = (\eta_o - \eta) + \| \epsilon - \epsilon \| + \frac{1}{2} \omega^T (K^*)^{-1} \omega \]
and
\[ V_2(x) = \frac{1}{2} \| p_d - p \|^2 + \frac{1}{2} \nu^T (K^*)^{-1} \nu \]
For \( V_1(x) \), take the derivative of both sides
\[ \dot{V}_1(x) = -2(\eta_o - \eta) \dot{\eta} + 2(\epsilon - \epsilon)^T \dot{\epsilon} + \omega^T (K^*)^{-1} \omega \]
By the formula (19) and (28)
\[ \dot{V}_1(x) = (\eta_o - \eta) \omega^T \epsilon - (\epsilon - \epsilon)^T \dot{\epsilon}_o + \omega^T (K^*)^{-1} \omega \]
\[ + \omega^T \left( \eta_o - \eta - \epsilon \dot{\epsilon}_o + \epsilon \dot{\epsilon}_o \right) \]
\[ = -\omega^T (K^*)^{-1} D^* \omega + \omega^T \left( \text{vec} (q^*_o \otimes q^*_o) - \text{vec} (q^*_o \otimes q^*_o) \right) \]
\[ = -\omega^T (K^*)^{-1} D^* \omega \quad (22) \]
Design \( K^*, D^* \) to be positive definite can ensure that \( \dot{V}_1(x) \) is negative definite, so \( q_o \) asymptotically converges to \( q_o \), with \( \omega = 0 \).
\[ \dot{V}_2(x) = -(p_d - p)^T \dot{p} + \nu^T (K^*)^{-1} \dot{\nu} \]
\[ = -(p_d - p)^T \dot{p} + \nu^T (K^*)^{-1} \left( K^* \text{vec} (p_d - p) - D^* \nu \right) \]
\[ = \nu^T \left( p_d - p \right)^T \left( p \times \omega \right) - \nu^T (K^*)^{-1} D^* \nu \]
\[ = p_d^T \left( p \times \omega \right) - \nu^T (K^*)^{-1} D^* \nu \]
\[ = \omega^T \left( p \times p_d \right) - \nu^T (K^*)^{-1} D^* \nu \quad (23) \]
Under the premise of \( q_o \) asymptotic convergence, \( \omega \) converges to 0, \( \omega^T (p \times p_d) \) will be an item as small as possible, so \( \dot{V}_2(x) \) can be inferred negative definite, \( p \) asymptotically converges to \( p_d \) with \( \nu = 0 \).

To sum up, the system globally asymptotically converges to \( q_o \) with \( \xi = 0 \).

IV. SIMULATION DESIGN

The simulation system is built by the open source flight control px4 and the flight simulation software X-Plane. The joystick sends instructions to the flight control, after processing, the control signal is sent to X-Plane to control the aircraft in X-Plane. This is a typical hardware-in-the-loop simulation system.

For different maneuvers, we collect flight data through expert teaching, and the sampling frequency is set to 100Hz. This data is used to train the DMP, the entire training process is shown in the following figure.

The Gaussian kernel function is selected as
\[ \psi_i(x) = \frac{\exp \left( -\frac{1}{2} (x - c_i)^2 h_i^{-1} \right)}{\sqrt{2\pi h_i}} \quad (24) \]
Set total duration as \( T \) and sampling time as \( \Delta t \)
\[ c_i = \exp \left( -\alpha_i \frac{t}{N + 1} \right) \quad i = 1, \ldots, N \]
\[ h_i = \left[ c_i - \exp \left( -\alpha_i \left( 1 + \frac{(T - \Delta t)(N - i)}{\Delta t(N - 1)} + \sqrt{\frac{T}{\Delta t}} \right) \right) \right]^2 \quad (25) \]
In this experiment, the somersaults motion is taken for a simulation test, which lasts 18.9 seconds, the final parameters are selected as follows:

- Pose DMP (Consider both position and quaternion):
  \[ \alpha = 0.1, N^p = 30, K^p = 10, D^p = 10\sqrt{K^p}, N^q = 50, K^q = 1 \]
  and \[ D^q = 10\sqrt{K^q}. \]

- Dual quaternion DMP:
  \[ \alpha = 0.05, N^q = 30, K^q = 1, \]
  \[ D^q = 10\sqrt{K^q}, K^q = 1, D^q = 10\sqrt{K^q}. \]

V. RESULTS AND DISCUSSION

We show the results of the training in Figure 3 to 4. It should be noted that position of traditional pose DMP is only applicable to the inertial coordinate system. However, the reference frame of angular velocity is the body coordinate, because it is generally selected by default in the data collection of the flight controller.

For the quaternion DMP in the Sec. II, the formula (6) and (7) needs to be revised as follows in the body coordinate system:

\[
\tau \dot{q} = \frac{1}{2} q \otimes \dot{\omega} \\
\tau \dot{\omega} = K^q \left[ e_q \left( q^* \otimes q_q \right) - d^q_q(x) + f^q_q(x) \right] - D^q \omega
\]

For dual quaternion operations, a unified coordinate system is required. Here, the body coordinate system is selected for operation. In addition, the position and speed of the inertial coordinate system can be obtained through coordinate system conversion. In the training process and result display, different forms of scale transformation are performed on the position.

Figure 3. This figure shows the training effect of pose DMP, the result in (c) and (g) is the change of the Gaussian kernel function \( \psi_1(x) \) and the error of the attitude in (h) is the norm of the quaternion. Gaussian kernel function 1 is used for training position and Gaussian kernel function 2 for attitude.

Figure 4. This figure demonstrates the training effect of Dual quaternion DMP, the definition of each quantity is the same as before.
It can be seen that the traditional pose DMP can achieve better results in their respective learning. However, such a system ignores the influence of attitude on position. During somersaults motion, the pitch angle and pitch angle velocity change fast, and the coupling of pose is serious. Obviously, in our dual quaternion DMP, when the attitude learning has a certain deviation, the position will also have a corresponding deviation, which is a performance of the UAV’s motion constraint. Therefore, in the traditional pose decoupling DMP, it is more likely to train an unreachable state.

For the UAV system, it has to meet certain constraints of motion differentially obviously. In this case, the dual quaternion DMP is more appropriate to describe the problem.

VI. CONCLUSION

In this paper, through the idea of imitation learning, the motion description of the acrobatic flight of the UAV is realized. Considering the coupling problem of the position and the attitude in the acrobatic flight, the dual quaternion is introduced to describe the translational and rotational motions in a unified manner. On the basis of dual quaternion DMP, the coupling problem of position and attitude is solved. Our method was successfully applied to the acrobatic flight in a hardware-in-the-loop simulation.

In the future we will strive to express more complex maneuvers through DMP.

APPENDIX

The basic operations of dual quaternions are listed below.

The product of two quaternions is

$$q_1 \otimes q_2 = \left[ q_1^\ast, q_1 \times q_2 \right]$$

Conjugation of quaternion: $$q^\ast = [\eta, -\varepsilon], q \otimes q^\ast = [1, 0, 0, 0].$$

The time derivative of a quaternion can be expanded into

$$\dot{q} = \frac{1}{2} q \otimes \omega = \frac{1}{2} \left[ \eta, \varepsilon \right] \otimes [0, \omega]$$

Dual quaternions have many similar properties to ordinary quaternions. The conjugate operation of the dual quaternion selected in this paper $$q^\ast = q_0^\ast + \varepsilon q_p^\ast$$, and the addition operation of the dual quaternion is $$q_1 + q_2 = q_0 + q_{12} + \varepsilon (q_{p1} + q_{p2})$$.

The multiplication operation of two dual quaternions is

$$q_1 \otimes q_2 = q_0 \otimes q_{12} + \varepsilon \left( q_0 \otimes q_{p2} + q_{p0} \otimes q_{p2} \right).$$

The time derivative of a dual quaternion can be expanded into

$$\dot{q} = \frac{1}{2} q \otimes \omega = \frac{1}{2} \left( q_0 + \varepsilon q_p \right) \cdot \left( \dot{\omega} + \varepsilon \dot{\nu} \right)$$

$$= \frac{1}{2} q_0 \otimes \dot{\omega} + \frac{1}{2} \left( q_0 \otimes \dot{\nu} + q_p \otimes \dot{\omega} \right).$$

The rotation matrix for linear velocity from body coordinate system to inertial coordinate system is

$$\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{z}
\end{bmatrix} = 2 \begin{bmatrix}
q_0 q_1 - q_0 q_3 \\
q_0 q_2 - q_1 q_3 \\
q_2 q_3 + q_1 q_0
\end{bmatrix} \begin{bmatrix}
q_0^2 - q_1^2 - q_2^2 + q_3^2 \\
2(q_0 q_1 - q_0 q_3) \\
2(q_0 q_2 + q_1 q_3)
\end{bmatrix}$$

where quaternion $$q = [q_0, q_1, q_2, q_3]$$.  

REFERENCES

[1] Zhang R, Zhang J, Yu H. Review of modeling and control in UAV autonomous maneuvering flight: 2018 IEEE International Conference on Mechatronics and Automation (ICMA), 2018[C]. IEEE.

[2] Shuaialong I, Zhang H, Zhou W. Review of Imitation Learning Methods and Its Application in Robotics[J]. Computer Engineering and Applications, 2019.

[3] Saveriano M, Abu-Dakka F J, Kramberger A et al. Dynamic Movement Primitives in Robotics: A Tutorial Survey[J]. arXiv preprint arXiv:2102.03861, 2021.

[4] Ijspeert A J, Nakanishi J, Schaal S. Movement imitation with non-linear dynamical systems in humanoid robots: Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292), 2002[C]. IEEE.

[5] Ijspeert A J, Nakanishi J, Hoffmann H et al. Dynamical movement primitives: learning attractor models for motor behaviors[J]. Neural Computation, 2013,25(2):328-373.

[6] Pastor P, Righetti L, Kalakrishnan M et al. Online movement adaptation based on previous sensor experiences: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2011[C]. IEEE.

[7] Ude A, Nemec B, Petric T et al. Orientation in cartesian space dynamic movement primitives: 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014[C]. IEEE.

[8] Saveriano M, Franzel F, Lee D. Merging position and orientation motion primitives: 2019 International Conference on Robotics and Automation (ICRA), 2019[C]. IEEE.

[9] Saveriano M, Franzel F, Lee D. Merging position and orientation motion primitives: 2019 International Conference on Robotics and Automation (ICRA), 2019[C]. IEEE.

[10] Sun B, Slotine J E. Motion primitives for robotic flight control[J]. arXiv preprint cs/0609140, 2006.

[11] Fang Z, Wang G, Li W et al. Control-oriented modeling of flight demonstrations for quadrotors using higher-order statistics and dynamic movement primitives: 2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE), 2014[C]. IEEE.

[12] Tomić T, Maier M, Haddadin S. Learning quadrotor maneuvers from optimal control and generalizing in real-time: 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014[C]. IEEE.

[13] Lee H, Seo H, Kim H. Trajectory Optimization and Replanning Framework for a Micro Air Vehicle in Cluttered Environments[J]. IEEE Access, 2020,8:135406-135415.

[14] Wang X, Yu C, Lin Z. A dual quaternion solution to attitude and position control for rigid-body coordination[J]. IEEE Transactions on Robotics, 2012,28(5):1162-1170.