1 Quaternion Multiplication

$$
\begin{aligned}
Q_{\mathrm{Wnew}} &= Q_{\mathrm{W1}}(Q_{\mathrm{W2}}) - Q_{\mathrm{X1}}(Q_{\mathrm{X2}}) - Q_{\mathrm{Y1}}(Q_{\mathrm{Y2}}) - Q_{\mathrm{Z1}}(Q_{\mathrm{Z2}}) \\
Q_{\mathrm{Xnew}} &= Q_{\mathrm{W1}}(Q_{\mathrm{X2}}) - Q_{\mathrm{X1}}(Q_{\mathrm{W2}}) - Q_{\mathrm{Y1}}(Q_{\mathrm{Z2}}) - Q_{\mathrm{Z1}}(Q_{\mathrm{Y2}}) \\
Q_{\mathrm{Ynew}} &= Q_{\mathrm{W1}}(Q_{\mathrm{Y2}}) - Q_{\mathrm{X1}}(Q_{\mathrm{Z2}}) - Q_{\mathrm{Y1}}(Q_{\mathrm{W2}}) - Q_{\mathrm{Z1}}(Q_{\mathrm{X2}}) \\
Q_{\mathrm{Znew}} &= Q_{\mathrm{W1}}(Q_{\mathrm{Z2}}) - Q_{\mathrm{X1}}(Q_{\mathrm{Y2}}) - Q_{\mathrm{Y1}}(Q_{\mathrm{X2}}) - Q_{\mathrm{Z1}}(Q_{\mathrm{W2}})
\end{aligned}
$$

2 Quaternion Division

$$
\begin{aligned}
Q_{\mathrm{Wnew}} &= Q_{\mathrm{W1}}(Q_{\mathrm{W2}}) + Q_{\mathrm{X1}}(Q_{\mathrm{X2}}) + Q_{\mathrm{Y1}}(Q_{\mathrm{Y2}}) + Q_{\mathrm{Z1}}(Q_{\mathrm{Z2}}) \\
Q_{\mathrm{Xnew}} &= -Q_{\mathrm{W1}}(Q_{\mathrm{X2}}) + Q_{\mathrm{X1}}(Q_{\mathrm{W2}}) + Q_{\mathrm{Y1}}(Q_{\mathrm{Z2}}) - Q_{\mathrm{Z1}}(Q_{\mathrm{Y2}}) \\
Q_{\mathrm{Ynew}} &= -Q_{\mathrm{W1}}(Q_{\mathrm{Y2}}) - Q_{\mathrm{X1}}(Q_{\mathrm{Z2}}) + Q_{\mathrm{Y1}}(Q_{\mathrm{W2}}) + Q_{\mathrm{Z1}}(Q_{\mathrm{X2}}) \\
Q_{\mathrm{Znew}} &= -Q_{\mathrm{W1}}(Q_{\mathrm{Z2}}) + Q_{\mathrm{X1}}(Q_{\mathrm{Y2}}) - Q_{\mathrm{Y1}}(Q_{\mathrm{X2}}) + Q_{\mathrm{Z1}}(Q_{\mathrm{W2}})
\end{aligned}
$$

3 Quaternion Conjugate

$$
\begin{aligned}
Q_{\mathrm{Wconjugate}} &= Q_{\mathrm{Winput}} \\
Q_{\mathrm{Xconjugate}} &= -Q_{\mathrm{Xinput}} \\
Q_{\mathrm{Yconjugate}} &= -Q_{\mathrm{Yinput}} \\
Q_{\mathrm{Zconjugate}} &= -Q_{\mathrm{Zinput}}
\end{aligned}
$$

4 Quaternion Normal

$$Normal_{\mathrm{rational}} = Q_W^2 + Q_X^2 + Q_Y^2 + Q_Z^2$$

5 Quaternion Multiplicative Inverse

$$Q_{\mathrm{reciprocal}} = Q_{\mathrm{conjugate}} \frac{1}{Q_{\mathrm{normal}}}$$

6 Quaternion Vector Rotation

$$\begin{aligned} Q_{\text{Wbivector}} &= \ 0 \\ Q_{\text{Xbivector}} &= V_X \\ Q_{\text{Ybivector}} &= V_Y \\ Q_{\text{Zbivector}} &= V_Z \end{aligned}$$

$$Q_{\text{rotation}} = Q_{\text{current}}(Q_{\text{bivector}})(Q_{\text{reciprocal}})$$

$$\begin{aligned} V_{\text{Xrotated}} &= Q_{\text{Xrotation}} \\ V_{\text{Yrotated}} &= Q_{\text{Yrotation}} \\ V_{\text{Zrotated}} &= Q_{\text{Zrotation}} \end{aligned}$$

7 Quaternion Vector Rotation Removal

$$\begin{aligned} Q_{\text{Wbivector}} &= \ 0 \\ Q_{\text{Xbivector}} &= V_X \\ Q_{\text{Ybivector}} &= V_Y \\ Q_{\text{Zbivector}} &= V_Z \end{aligned}$$

$$Q_{\text{rotation}} = Q_{\text{conjugate}}(Q_{\text{bivector}})(Q_{\text{reciprocal}})$$

$$\begin{aligned} V_{\text{Xrotated}} &= Q_{\text{Xrotation}} \\ V_{\text{Yrotated}} &= Q_{\text{Yrotation}} \\ V_{\text{Zrotated}} &= Q_{\text{Zrotation}} \end{aligned}$$

8 Unit Quaternion

$$\begin{aligned} Q_{\text{Wunit}} &= \tfrac{Q_{\text{Winput}}}{Q_{\text{normal}}} \\ Q_{\text{Xunit}} &= \tfrac{Q_{\text{Xinput}}}{Q_{\text{normal}}} \\ Q_{\text{Yunit}} &= \tfrac{Q_{\text{Yinput}}}{Q_{\text{normal}}} \\ Q_{\text{Zunit}} &= \tfrac{Q_{\text{Zinput}}}{Q_{\text{normal}}} \end{aligned}$$

9 Quaternion Dot Product

$$D_{\text{dot}} = Q_{\text{W1}}(Q_{\text{W2}}) + Q_{\text{X1}}(Q_{\text{X2}}) + Q_{\text{Y1}}(Q_{\text{Y2}}) + Q_{\text{Z1}}(Q_{\text{Z2}})$$

10 Quaternion Magnitude

$$M_{\text{magnitude}} = \sqrt{Q_{\text{normal}}}$$

11 Quaternion Additive Inverse

$$Q_{\text{Wnegative}} = -Q_{\text{Winput}}$$
$$Q_{\text{Xnegative}} = -Q_{\text{Xinput}}$$
$$Q_{\text{Ynegative}} = -Q_{\text{Yinput}}$$
$$Q_{\text{Znegative}} = -Q_{\text{Zinput}}$$

12 Quaternion Smooth Interpolation Between Quaternions

$Q_{\text{initial}} = Q_{\text{Unit initial}}$
$Q_{\text{final}} = Q_{\text{Unit final}}$

$D_{\text{dot}} = Q_{\text{initial}} \cdot Q_{\text{final}}$

$$Q_{\text{initial}} = \begin{cases} Q_{\text{initial}} = Q_{\text{initial(additive inverse)}}, & \text{if } D_{\text{dot}} < 0. \\ Q_{\text{initial}}, & \text{otherwise.} \end{cases}$$

$D_{\text{dot}} = |D_{\text{dot}}|$

$$D_{\text{dot}} = \begin{cases} 1, & \text{if } D_{\text{dot}} > 1. \\ D_{\text{dot}}, & \text{otherwise.} \end{cases}$$

$\theta = arccos(D_{\text{dot}}) \times ratio$

$Q_{\text{orthonomal}} = Q_{\text{final}} - Q_{\text{initial}} \times D_{\text{dot}}$
$Q_{\text{output}} = Q_{initial} \times cos(\theta) + Q_{\text{orthonomal}} \times sin(\theta)$

13 Quadcopter Combined Thrust Vector

$Q_{\text{change}} = (\frac{2\times(Q_{\text{target}}-Q_{\text{current}})\times Q_{\text{current conjugate}}}{dT})$

$V_{\text{Xchange}} = Q_{\text{Xchange}}$
$V_{\text{Ychange}} = Q_{\text{Ychange}}$
$V_{\text{Zchange}} = Q_{\text{Zchange}}$

$V_{\text{RotationOutput}} = FeedbackController_{\text{rotation}}.Calculate(0, V_{\text{change}})$
$V_{\text{PositionOutput}} = FeedbackController_{\text{position}}.Calculate(0, V_{\text{CurrentPosition}} - V_{\text{TargetPosition}})$

$V_{\text{YThrusterBOutput}} = -V_{\text{XRotationOutput}} + V_{\text{ZRotationOutput}} - V_{\text{YRotationOutput}}$
$V_{\text{YThrusterCOutput}} = -V_{\text{XRotationOutput}} - V_{\text{ZRotationOutput}} + V_{\text{YRotationOutput}}$
$V_{\text{YThrusterDOutput}} = \quad V_{\text{XRotationOutput}} - V_{\text{ZRotationOutput}} - V_{\text{YRotationOutput}}$
$V_{\text{YThrusterEOutput}} = \quad V_{\text{XRotationOutput}} + V_{\text{ZRotationOutput}} + V_{\text{YRotationOutput}}$

$V_{\text{HoverAngles}} = RotationToHoverAngles(Q_{\text{CurrentRotation}})$

$V_{\text{PositionOutput}} = CalculateRotationOffset(Q_{\text{CurrentRotation}}).RotateVector(V_{\text{PositionOutput}})$

$V_{\text{XPositionOutput}} = V_{\text{XPositionOutput}} + V_{\text{ZHoverAngles}}$
$V_{\text{ZPositionOutput}} = V_{\text{ZPositionOutput}} - V_{\text{XHoverAngles}}$

$V_{\text{ThrusterBOutput}} = V_{\text{ThrusterBOutput}} + V_{\text{PositionOutput}}$
$V_{\text{ThrusterCOutput}} = V_{\text{ThrusterCOutput}} + V_{\text{PositionOutput}}$
$V_{\text{ThrusterDOutput}} = V_{\text{ThrusterDOutput}} + V_{\text{PositionOutput}}$
$V_{\text{ThrusterEOutput}} = V_{\text{ThrusterEOutput}} + V_{\text{PositionOutput}}$

14 Quadcopter Thruster Position Calculation

$V_{\text{ThrusterBPosition}} = Q_{\text{CurrentRotation}}.RotateVector(V_{\text{ThrusterBOffset}}) + V_{\text{TargetPosition}}$
$V_{\text{ThrusterCPosition}} = Q_{\text{CurrentRotation}}.RotateVector(V_{\text{ThrusterCOffset}}) + V_{\text{TargetPosition}}$
$V_{\text{ThrusterDPosition}} = Q_{\text{CurrentRotation}}.RotateVector(V_{\text{ThrusterDOffset}}) + V_{\text{TargetPosition}}$
$V_{\text{ThrusterEPosition}} = Q_{\text{CurrentRotation}}.RotateVector(V_{\text{ThrusterEOffset}}) + V_{\text{TargetPosition}}$

15 Quadcopter Hover Angle Calculation

$DA_{\text{Direction}} = RotationMatrix.RotateVector(EA_{\text{rotate}}(0, -90, 0), DA_{\text{Direction}})$

$DA_{\text{Direction}} = RotationMatrix.RotateVector(EA_{\text{rotate}}(0, DA_{\text{Rotation}}, 0), DA_{\text{Direction}})$

$D_{\text{InnerJoint}} = RadiansToDegrees(arcsin(D_{\text{DirectionVectorZ}}))$
$D_{\text{OuterJoint}} = RadiansToDegrees(arctan2(D_{\text{DirectionVectorX}}, D_{\text{DirectionVectorY}}))$

16 Quadcopter Estimate Position

$V_{\text{TBThrust}} = Vector(0, ThrustBOutputY, 0)$
$V_{\text{TCThrust}} = Vector(0, ThrustCOutputY, 0)$
$V_{\text{TDThrust}} = Vector(0, ThrustDOutputY, 0)$
$V_{\text{TEThrust}} = Vector(0, ThrustEOutputY, 0)$

$Q_{\text{TBR}} = EA(ThrustBOutput.X, 0, -ThrustBOutput.Z)$
$Q_{\text{TCR}} = EA(ThrustCOutput.X, 0, -ThrustCOutput.Z)$
$Q_{\text{TDR}} = EA(ThrustDOutput.X, 0, -ThrustDOutput.Z)$
$Q_{\text{TER}} = EA(ThrustEOutput.X, 0, -ThrustEOutput.Z)$

$V_{\text{TBThrust}} = Q_{\text{TBR}}.RotateVector(TBThrust)$
$V_{\text{TCThrust}} = Q_{\text{TCR}}.RotateVector(TCThrust)$
$V_{\text{TDThrust}} = Q_{\text{TDR}}.RotateVector(TDThrust)$
$V_{\text{TEThrust}} = Q_{\text{TER}}.RotateVector(TEThrust)$

$V_{\text{ThrustSum}} = V_{\text{TBThrust}} + V_{\text{TCThrust}} + V_{\text{TDThrust}} + V_{\text{TEThrust}}$
$V_{\text{ThrustSum}} = Q_{\text{current}}.RotateVector(V_{\text{ThrustSum}})$

$V_{\text{XDragForce}} = D_{\text{AirDensity}} \times D^2_{\text{XCurrentVelocity}} \times D_{\text{DragCoefficient}} \times Sign(XCurrentVelocity)$
$V_{\text{YDragForce}} = D_{\text{AirDensity}} \times D^2_{\text{YCurrentVelocity}} \times D_{\text{DragCoefficient}} \times Sign(YCurrentVelocity)$
$V_{\text{ZDragForce}} = D_{\text{AirDensity}} \times D^2_{\text{ZCurrentVelocity}} \times D_{\text{DragCoefficient}} \times Sign(ZCurrentVelocity)$

$V_{\text{CurrentAcceleration}} = V_{\text{ThrustSum}} + V_{\text{WorldAcceleration}}$
$V_{\text{CurrentVelocity}} = V_{\text{CurrentVelocity}} + V_{\text{CurrentAcceleration}} \times D_{\text{TimeDerivative}}$
$V_{\text{CurrentPosition}} = V_{\text{CurrentPosition}} + V_{\text{CurrentVelocity}} \times D_{\text{TimeDerivative}}$