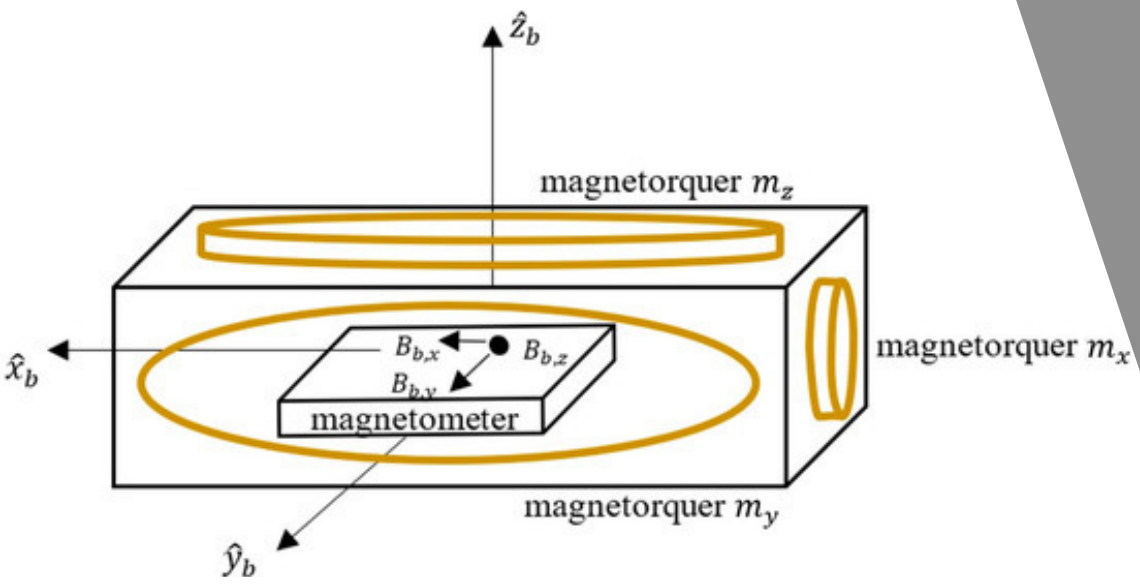
2025

SATELLITE DETUMBLING SIMULATION USING (B-DOT) CONTROL

GUIDANCE, NAVIGATION, AND CONTROL (GNC)



PREPARED BY

- **MOHAMED HARHASH**

1. Introduction

Controlling a satellite's attitude (its orientation in space) is crucial for mission success. When satellites are deployed into orbit, they often tumble due to deployment forces. This tumbling must be stopped or reduced — a process known as **detumbling** — before more precise attitude control can begin.

The **B-dot algorithm** is a simple, low-power method for detumbling. It relies solely on the **Earth's magnetic field** and onboard **magnetorquers** — devices that produce magnetic dipole moments to create torques.

This simulation models the detumbling of a small satellite (e.g., CubeSat) using B-dot control in MATLAB.

2. Assumptions

To simplify the model and focus on the core B-dot control concept, the following assumptions are made:

Assumption	Description
Fixed Orbital Point	The satellite is considered to be stationary at a single orbital point (500 km, equator) — no orbital propagation is included.
Perfect Magnetometer	The magnetic field vector is assumed to be measured with no noise or bias.
No External Disturbances	No gravity-gradient, atmospheric drag, or solar pressure torques.
Rigid Body	The satellite is a rigid body with constant diagonal inertia.

3. Satellite Physical Model

The satellite is modeled as a rigid body governed by Euler's rotational dynamics:

3.1 Equations of Motion

(1) Rotational Dynamics (Euler's Equation):

$$\mathbf{I} \cdot \frac{d\boldsymbol{\omega}}{dt} = \mathbf{T}_{\text{control}} - \boldsymbol{\omega} \times (\mathbf{I} \cdot \boldsymbol{\omega})$$

- $\mathbf{I}$: Inertia matrix (3×3 diagonal matrix)
- $\boldsymbol{\omega}$: Angular velocity vector [rad/s]
- $\mathbf{T}_{\text{control}}$: Control torque (magnetic in this case)

(2) Quaternion Kinematics:

To represent 3D orientation, unit quaternions are used:

$$\frac{d\mathbf{q}}{dt} = \frac{1}{2}\Omega(\boldsymbol{\omega}) \cdot \mathbf{q}$$

Where:

$$\Omega(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix}$$

4. B-dot Control Concept

4.1 Magnetic Torque

A magnetic dipole **m** in a magnetic field **B** generates a torque:

$$\mathbf{T}_{\text{mag}} = \mathbf{m} \times \mathbf{B}$$

4.2 B-dot Control Law

The B-dot controller calculates a magnetic dipole that opposes changes in the magnetic field vector (i.e., rotational motion):

$$\mathbf{m} = k_{\text{bdot}} \cdot \frac{d\mathbf{B}}{dt}$$

Where:

- $k_{\text{bdot}} < 0$ is a gain to oppose the rotation
- $\frac{d\mathbf{B}}{dt}$ is the time derivative of the magnetic field in the **body frame**

To prevent actuator saturation, the dipole is limited:

$$\mathbf{m}_{\text{sat}} = \begin{cases} \mathbf{m}, & \text{if } \|\mathbf{m}\| \leq m_{\max} \\ \frac{m_{\max}}{\|\mathbf{m}\|} \cdot \mathbf{m}, & \text{otherwise} \end{cases}$$

5. MATLAB Simulation Details

5.1 Parameters

Parameter	Value
Time step (dt)	1 s
Simulation time	100,000 s
Inertia Matrix (I)	diag([0.01, 0.01, 0.02]) kg·m^2
Initial Angular Velocity	[1; -1; 0.5] deg/s
Initial Attitude (quaternion)	[1; 0; 0; 0]
B-dot gain	−1000-1000
Max magnetic dipole moment	1 A·m^2
Orbit altitude	500 km

- Code Snippet:

```
%% --- Simulation Parameters ---
dt = 1;                     % Time step [s]
T = 100000;                  % Total simulation time [s]
N = T/dt;                   % Number of steps
time = 0:dt:T;              % Time vector

%% --- Satellite Parameters ---
I = diag([0.01, 0.01, 0.02]);     % Inertia matrix [kg·m²]
invI = inv(I);
k_bdot = -1e3;                    % B-dot gain
m_max = 1;                      % Max dipole [A·m²]

%% --- Initial Conditions ---
w = deg2rad([1; -1; 0.5]);        % Angular velocity [rad/s]
q = [1; 0; 0; 0];                 % Quaternion [w x y z] (scalar-first)

%% --- Orbit Parameters (example location) ---
lat = 0;                          % Latitude [deg]
lon = 0;                          % Longitude [deg]
alt = 500;                        % Altitude [km]
```

6. Simulation Loop Workflow

At each time step:

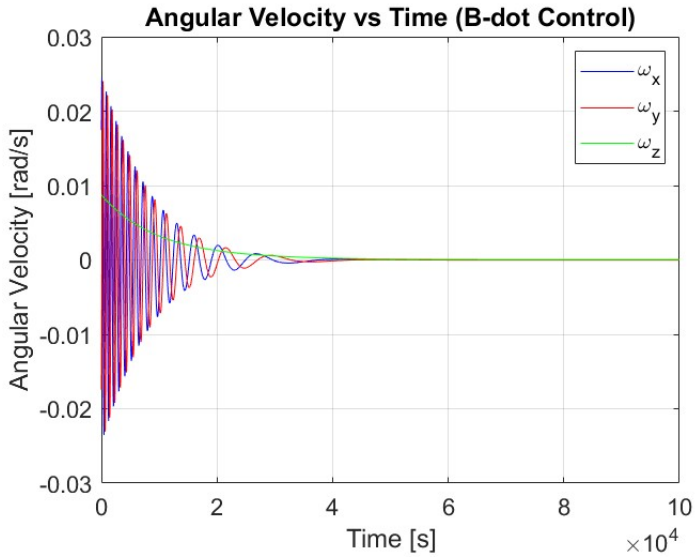
1. Get Earth's magnetic field vector from **IGRF model**
2. Transform it to the satellite **body frame** using the **rotation matrix** from the quaternion
3. Compute dB/dt
4. Apply B-dot control law
5. Compute torque and update dynamics
6. Normalize quaternion

```
%% --- Main Simulation Loop ---
for i = 1:N
    % Get current magnetic field in ECI
    t_now = startDate + seconds((i-1)*dt);
    decimalYear = year(t_now) + (day(t_now, 'dayofyear') - 1)/365;
    [Bx, By, Bz] = igrfmagm(alt, lat, lon, decimalYear);
    B_eci = 1e-9 * [Bx(1); By(1); Bz(1)];  % Convert nT to T

    % Rotate B field into body frame
    C = quat2dcm(q');
    B_body = C * B_eci;
    B_body_hist(:,i) = B_body;

    % Estimate dB/dt
    if i == 1
        dB = zeros(3,1);
    else
        dB = (B_body - B_body_hist(:, i-1)) / dt;
    end

    % B-dot control law with saturation
    m = k_bdot * dB;
    if norm(m) > m_max
        m = m_max * m / norm(m);
    end
    % Dynamics update
    Tm = cross(m, B_body);                      % Magnetic torque
    dw = invI * (Tm - cross(w, I*w));           % Angular acceleration
    w = w + dw * dt;                            % Update angular velocity

    % Quaternion update
    Omega = [  0,   -w(1), -w(2), -w(3);
              w(1),   0,   w(3), -w(2);
              w(2), -w(3),  0,    w(1);
              w(3),  w(2), -w(1),  0  ];
    q = q + 0.5 * Omega * q * dt;
    q = q / norm(q);                            % Normalize

    % Store history
    w_hist(:,i+1) = w;
    q_hist(:,i+1) = q;
end
```

7. Results

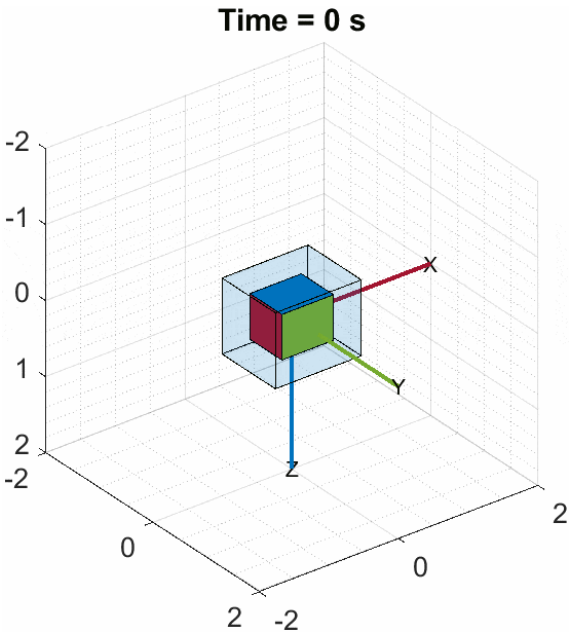
7.1 Angular Velocity Plot

As expected, the angular velocities ωx,ωy,ωz converge to near zero, showing successful detumbling.



7.2 Attitude Visualization

Using MATLAB's poseplot, we can visualize the satellite's orientation as it detumbles.



8. Educational Insights

- **Why B-dot Works**: The satellite's rotation causes the Earth's magnetic field vector to appear time-varying. By generating a dipole that opposes this change, we apply a torque that slows down rotation.
- **Why Saturation is Necessary**: Real magnetorquers have current limits. Saturating prevents applying unrealistic torques and reflects practical actuator constraints.
- **Why Use Quaternions**: They avoid singularities (gimbal lock) and are numerically stable for representing 3D rotations, unlike Euler angles.
- **Why the Satellite Doesn't Fully Stop**: The B-dot algorithm is a passive controller designed to reduce rotational speed, not achieve exact zero rotation. As angular velocity becomes small, the rate of change of the magnetic field also becomes small, resulting in diminishing control torques. Eventually, the control effect becomes negligible, and the satellite coasts with a very slow residual spin. This behavior is acceptable for transitioning into a fine-pointing control mode using more precise actuators.

9. Limitations

- Fixed orbit position: no true orbital motion
- No sensor or actuator modeling (noise, delay)
- No environmental disturbances

These are acceptable for early-stage controller validation.

10. Conclusion

This simulation successfully demonstrates the detumbling of a satellite using the B-dot control method. Even with simplified assumptions, it highlights the core physics and control logic behind one of the most used passive attitude control techniques for small satellites.

11. Future Improvements

- Simulate true orbital motion with TLE or Keplerian elements
- Add sensor noise and bias to simulate realistic magnetometers
- Include other disturbances (gravity-gradient, drag)
- Extend to hybrid attitude control (e.g., momentum wheels + B-dot)

12. Source Code:

Click Here