Control System Design for Spacecraft Docking Maneuver

AE322A

Department of Aerospace Engineering

Indian Institute of Technology Kanpur

Group Members	**Roll no**
Krishna Dantu	210299
Akshat Kumar	210091
Prabal Pratap Singh	210729
Atharv Soni	210229

April 23, 2024

Abstract

Spacecraft docking is a critical aspect of space missions, facilitating crewed spaceflights, satellite servicing, and space station operations. This paper presents a control system design for spacecraft docking, focusing on both translational (r-direction) and rotational (θ-direction) control. The control system architecture employs a combination of PID controller for translational control and PD controller for rotational control, optimized using the Nichols-Ziegler method. By considering constraints on settling time, rise time, and maximum overshoot, the controllers are tailored to ensure stable and efficient docking maneuvers. The results demonstrate successful optimization of controller gains, leading to satisfactory performance metrics such as settling time, rise time, and maximum overshoot. Through simulations and analysis, the effectiveness of the proposed control system in achieving precise and reliable spacecraft docking is validated.

1 Introduction

Spacecraft docking plays a crucial role in various space missions, including crewed spaceflights, satellite servicing, and space station operations. Achieving precise and reliable docking maneuvers is essential for ensuring the success and safety of these missions. In recent years, advances in control system design have enabled the development of sophisticated docking mechanisms capable of autonomous and semi-autonomous operations.

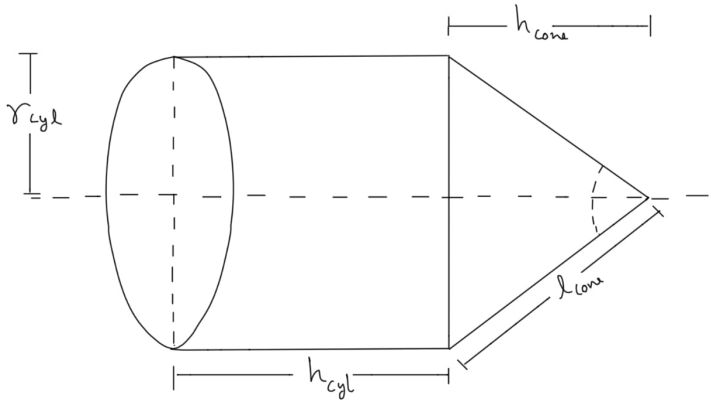
In this paper, we present a control system design for spacecraft docking, focusing on both the translational (r-direction) and rotational (θ-direction) control aspects. The control system architecture employs a combination of PID controllers for translational control and PD controllers for rotational control, tailored to meet the specific requirements of spacecraft docking tasks.

The design process begins with the derivation of plant functions for the r-direction and θ-direction, followed by the optimization of control parameters using the Nichols-Ziegler method. By considering constraints on settling time, rise time, and maximum overshoot, we aim to develop controllers that ensure stable and efficient docking maneuvers.

The results of our control system design demonstrate successful optimization of controller gains, leading to satisfactory performance metrics such as settling time, rise time, and maximum overshoot. Through simulations and analysis, we validate the effectiveness of the proposed control system in achieving precise and reliable spacecraft docking.

2 Mathematical formulations

Our simulation model comprises a hollow cone mounted on a hollow cylinder, both constructed from stainless steel or iron, with a density of 7.71 g/cc. The dimensions of the components are as follows:



Dimension	Length
r_{cyl}	8cm
r_{cone}	8cm
h_{cyl}	15cm
h_{cone}	10.5cm
l_{cone}	13.2cm
t	0.1cm

Figure 1: Body structure and measurements

Using these dimensions, the mass of the system is calculated as:

$$m = \rho \times (2\pi rht + \pi r(l + r)t) = 990.9g \approx 1000g \tag{1}$$

Hence, the total mass of the system is approximately 990.9 g. Additional components can be added to the cylinder to reach a mass of 1 kg.

The moment of inertia about the symmetric axis is determined by:

$$I = m_{\text{cyl}} r_{\text{cyl}}^2 + \frac{3}{10} m_{\text{cone}} r_{\text{cone}}^2 = 4.56 \times 10^{-3}\,\text{kgm}^2 \tag{2}$$

The movement of this body is confined to two degrees of freedom (2-DOF), as illustrated in Firure(2).

Based on this, the force and moment equations can be constructed as follows:

$$m\ddot{r}(t) + c\dot{r}(t) = F(t) \tag{3}$$

$$I\ddot{\theta}(t) = M(t) \tag{4}$$

In Equation (3), the damping coefficient c is included to account for various interacting forces acting on the body, such as solar radiation, electromagnetic fields, gas, and dust. However, these forces do not interfere with the rotation of the object around its symmetric axis hence it's omission in Equation (4). If $c = 2$ then the above equations will be written as

$$\ddot{r}(t) + 2\dot{r}(t) = F(t) \tag{5}$$

$$4.56 \times 10^{-3} \times \ddot{\theta}(t) = M(t) \tag{6}$$

3 Control Law formulations

3.1 Plant function

In our system the plant function consists of two components: one governing the translational motion (r-direction) and the other governing the rotational motion (θ-

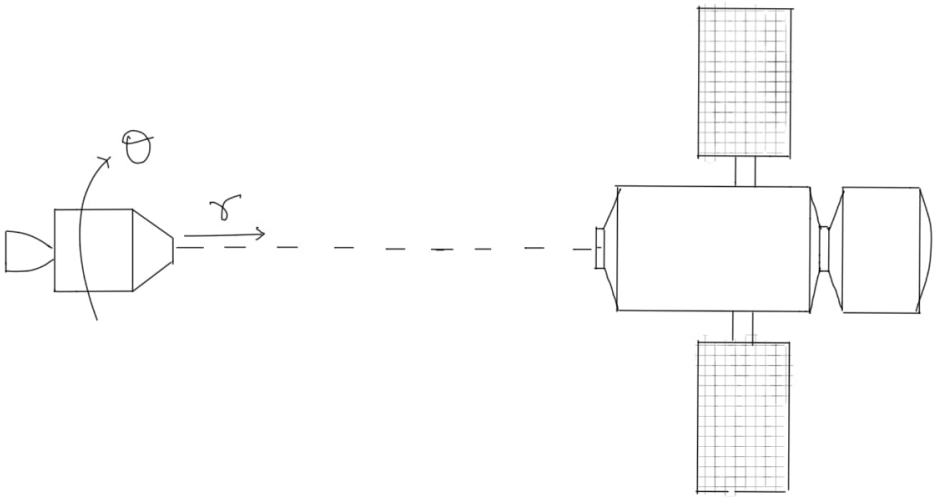


Figure 2: Path

direction). These components are represented by transfer functions derived from their respective equations of motion.

Applying the Laplace transform to Equations (5) and (6) yields the following plant functions in the Laplace domain.

For the r-direction

$$\mathcal{L}(\ddot{r}(t) + 2\dot{r}(t) = F(t)) \implies G_p(s) = \frac{R(s)}{F(s)} = \frac{1}{s^2 + 2s} \tag{7}$$

For the $\theta - direction$

$$\mathcal{L}(4.56 \times 10^{-3} \times \ddot{\theta}(t) = I(t)) \implies G_p(s) = \frac{\Theta(s)}{I(s)} = \frac{1}{4.56 \times 10^{-3} \times s^2} \tag{8}$$

3.2 Actuator model

The actuator model function $A(s)$, represents the dynamics of the actuator within the control system. In many control systems, the actuator introduces a delay or filtering effect due to its physical properties and response characteristics. In our case, the actuator function is represented by the transfer function:

$$A(s) = \frac{1}{0.08s + 1} \tag{9}$$

The denominator $0.08s + 1$ represents a first-order system with a time constant of $\tau = 0.08$ seconds. This time constant reflects the rate at which the actuator responds to changes in the control signal. A smaller time constant indicates a faster response, while a larger time constant implies a slower response.

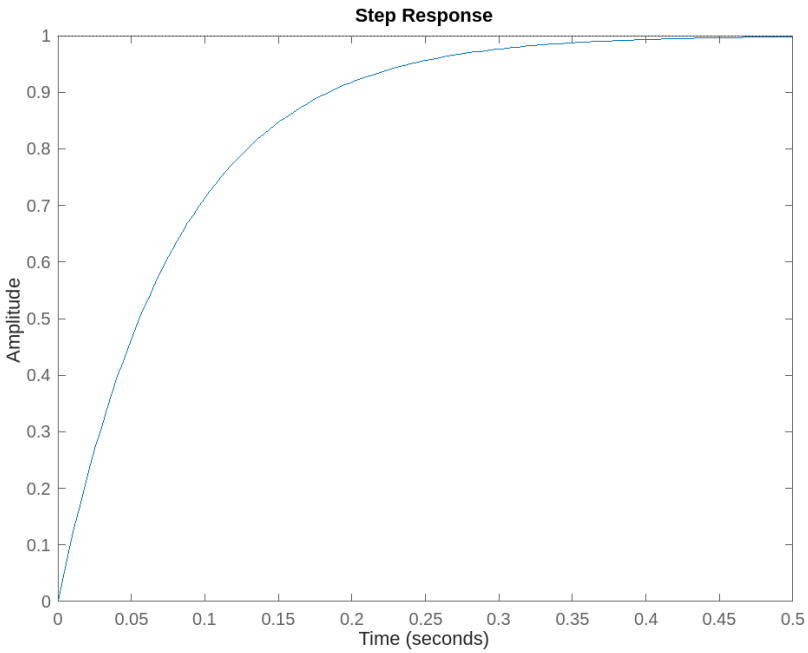


Figure 3: Actuator response

3.3 Controller function

3.3.1 Translational motion

We'll generate the plant functions starting with the r-direction. We'll consider the plant equation as a PID controller with gains k_p, k_d and k_i resulting in the following closed-loop PID block diagram:

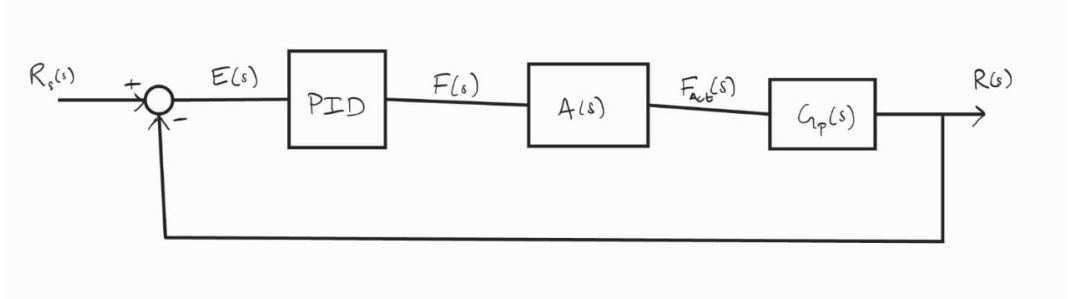


Figure 4: Closed-loop PID block diagram

$$F(t) = k_p e(t) + k_d \dot{e}(t) + k_i \int_0^t e(\tau) d\tau$$

applying transformation on F(t) gives

$$F(s) = E(s) \left(k_p + k_d s + \frac{k_i}{s} \right)$$

To determine the gains, we'll perform a Nichols-Ziegler optimization on the control loop by setting $k_d = k_i = 0$. The open-loop transfer function is then:

$$G_o(s) = G_c(s)A(s)G_p(s)$$

$$G_o(s) = k_p(s) \times \frac{1}{0.08s + 1} \times \frac{1}{s^2 + s} \tag{10}$$

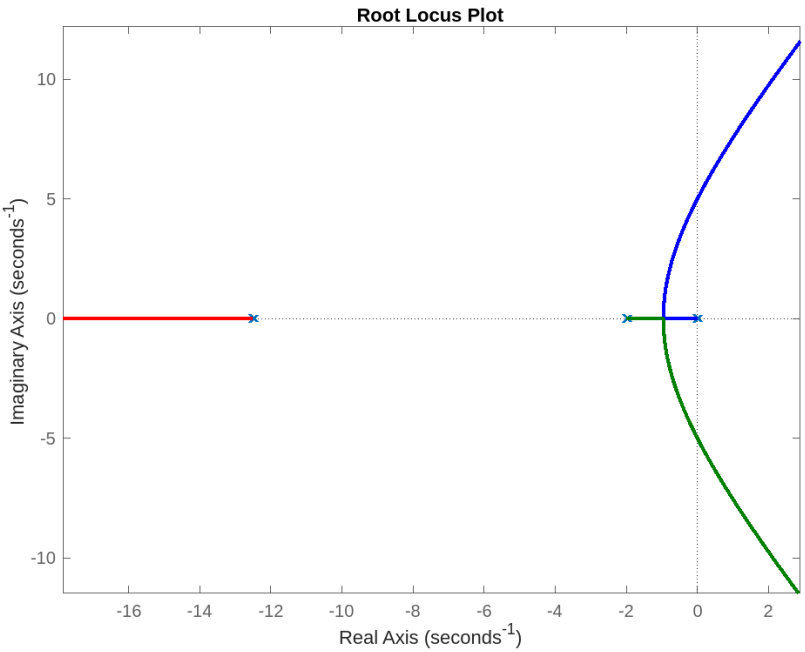


Figure 5: Root locus

The root locus plot of this open-loop transfer function is shown in Figure 5 found using code(1). The ultimate gain K_u and oscillation time period T_u are found by solving the characteristic equation:

$$0.08s^3 + 1.16s^2 + 2s + k_p \tag{11}$$

substituting $s = j\omega$ in Equation (11) and equating it to zero will give our ultimate gain and time period

$$-0.08j\omega^3 - 1.16w^2 + 2j\omega + k_p = 0$$

$$j\omega(2 - 0.08j\omega^2) = 0 \quad \& \quad k_p - 1.16\omega^2 = 0$$

$$\omega, k_p = 0, 0 \quad \& \quad \omega, k_p = 5, 29 \tag{12}$$

Hence K_u=29 and T_u=1.256 seconds. Ziegler-Nichols tuning method[1] suggests the gains in Table 1 for different controller types.

Figure 6 contains the plots of Ziegler-Nichols P, PD and PID controllers by using Matlab code(2).

Controller Type	Parameters
P	$k_p = 0.6K_u$
PD	$k_p = 0.8K_u$ $k_d = 0.10K_uT_u$
PID	$k_p = 0.6K_u$ $k_i = 1.2K_u/Tu$ $k_d = 0.075K_uT_u$

Table 1: Ziegler-Nichols Tuning Method

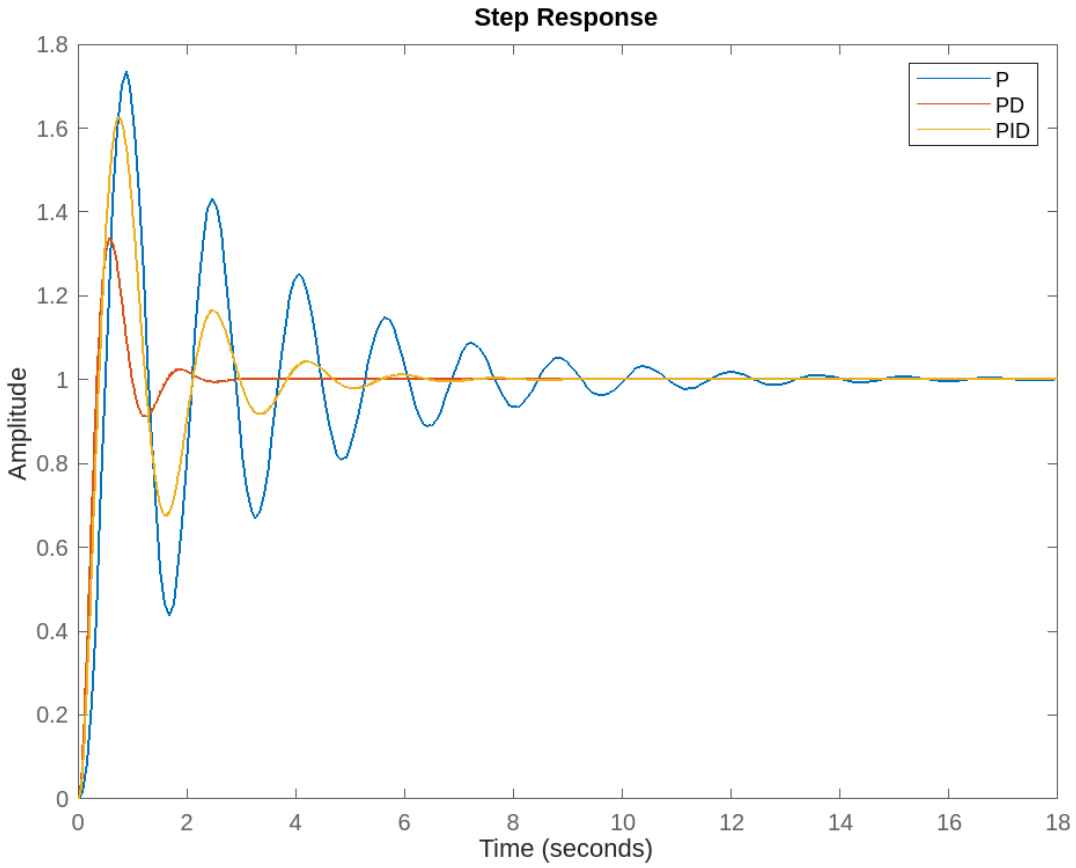


Figure 6: P, PD & PID responses

But for our problem it's important to have minimal to no overshoot as it's constrained on destination side due to presence of docking station. Hence we'll change these gain values so we can get very minimum overshoot. The solution for this was obtained by using gain values stated in Table 2 provided by microstar labs[2].

Using values in Table 2 the step response obtained is shown in Figure 7 and some info of the loop using code(3) was obtained.

The information obtained from code(3) for Figure 7 was

Gain		Relation
k_p	=	$0.2K_u$
k_i	=	$0.4\frac{K_u}{T_u}$
k_d	=	$0.667\frac{K_u}{T_u}$

Table 2: No overshoot gains

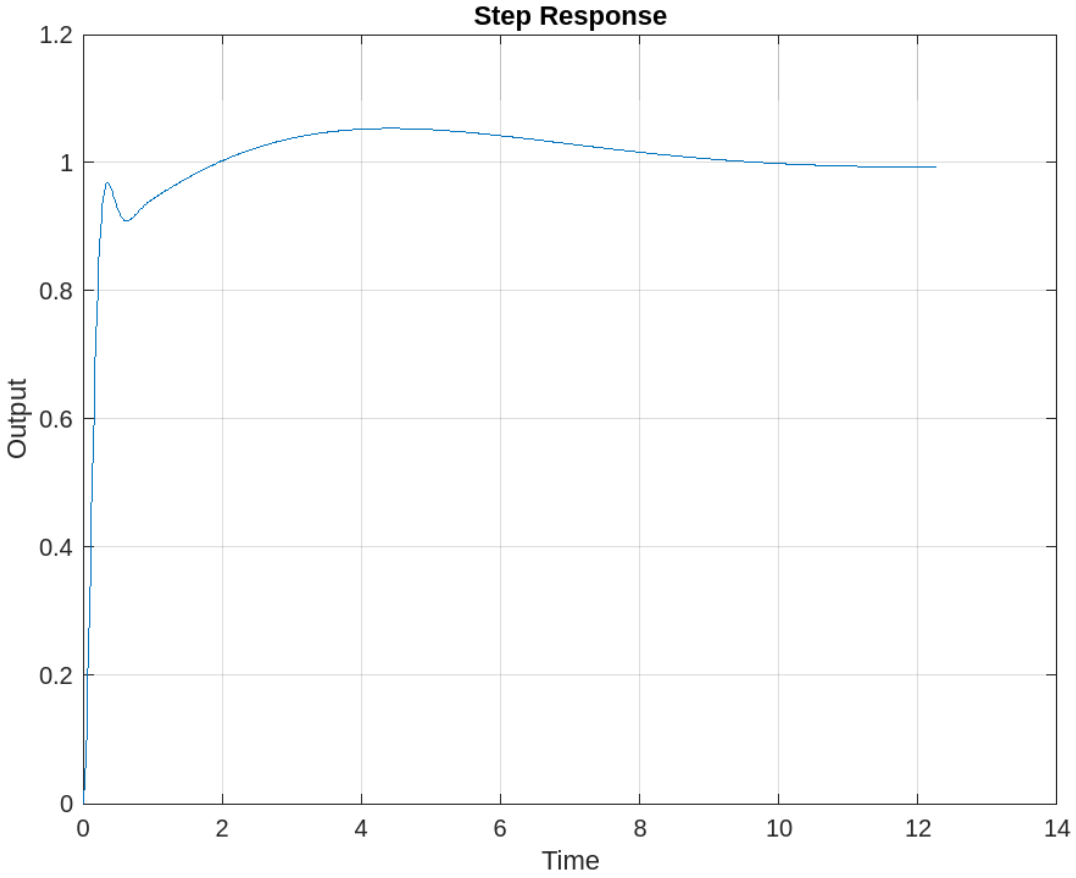


Figure 7: No overshoot step response

```
Settling time: 7.70 seconds
Maximum Overshoot: 5.34%
Time when step response becomes one: 1.94 seconds
```

3.3.2 Rotational motion

Moving on to the θ-direction, we used a closed-loop PD controller without an actuator model. The control laws used were

$$e(t) = r_s(t) - \theta(t)$$

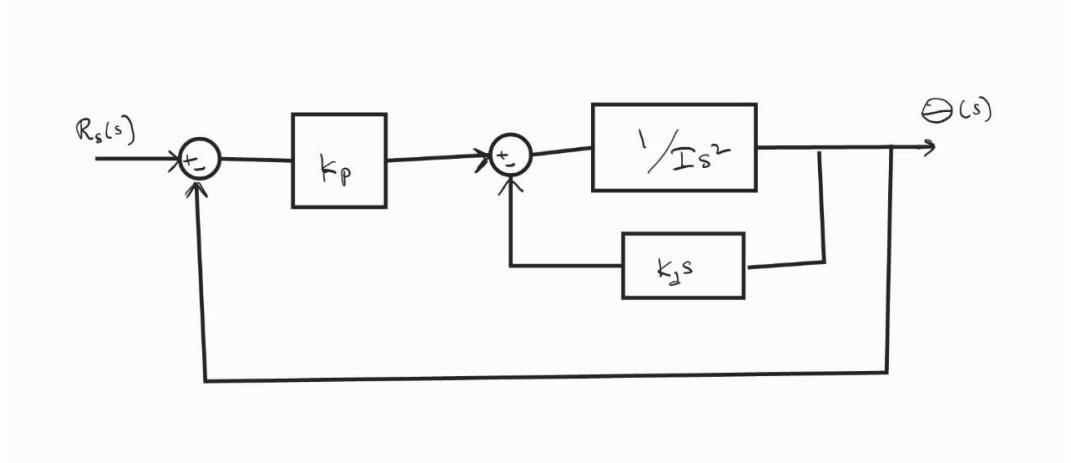


Figure 8: PD control loop

$$\dot{e}(t) = -\dot{\theta}(t)$$
$$I(t) = k_p e(t) + k_d \dot{e}(t)$$
$$\mathcal{L}\{I(t)\} = \mathcal{L}\{k_p e(t) - k_d \dot{\theta}(t)\}$$
$$I(s) = k_p E(s) - k_d s \Theta(s)$$

For the block diagram shown in Figure 8 the closed loop transfer function of this will be

$$\frac{\Theta(s)}{R_s(s)} = \frac{G_p(s) G_c(s)}{1 + G_p(s) G_c(s)}$$

where, $G_p(s) = \frac{1}{Is^2 + k_d s}$ and $G_c(s) = k_p$

$$\frac{\Theta(s)}{R_s(s)} = \frac{\left(\frac{k_p}{I}\right)}{s^2 + \left(\frac{k_d}{I}\right) s + \left(\frac{k_p}{I}\right)} \tag{13}$$

we'll define $\omega_n^2 = \frac{k_p}{I}$ & $2\zeta\omega_n = \frac{k_d}{I}$ where ζ is damping ratio and ω_n is natural frequency now rewriting equation in terms of ω_n & ζ

$$\frac{\Theta(s)}{R_s(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n + \omega_n^2} \tag{14}$$

and from further derivations we get rise time t_r, settling time t_s and maximum overshoot M_p

$$t_r = \frac{\pi - \tan^{-1}\left(\frac{\sqrt{1-\zeta^2}}{\zeta}\right)}{w_n \sqrt{1-\zeta^2}}$$

$$t_s = \frac{-\ln(0.02\sqrt{1-\zeta^2})}{\zeta\omega_n}$$

$$M_p = e^{-\pi\frac{\zeta}{\sqrt{1-\zeta^2}}} \times 100\%$$

applying constraints on these equations as $t_r < 1.94$, $t_s < 7.7$ and $M_p = 30\%$ we get

$$\zeta = 0.36(\text{from } M_p)$$

$$\omega_n > 1.07(\text{from } t_r)$$

$$\omega_n > 1.436(\text{from } t_s)$$

hence $\zeta = 0.36$ and $\omega_n > 1.436$. Figure 9 shows the step response for $\zeta = 0.36$ and $\omega_n = 1.5$, for which $t_s = 7.37$, $t_r = 1.38$ and $M_p = 30\%$. The plot for this is obtained by code(4)

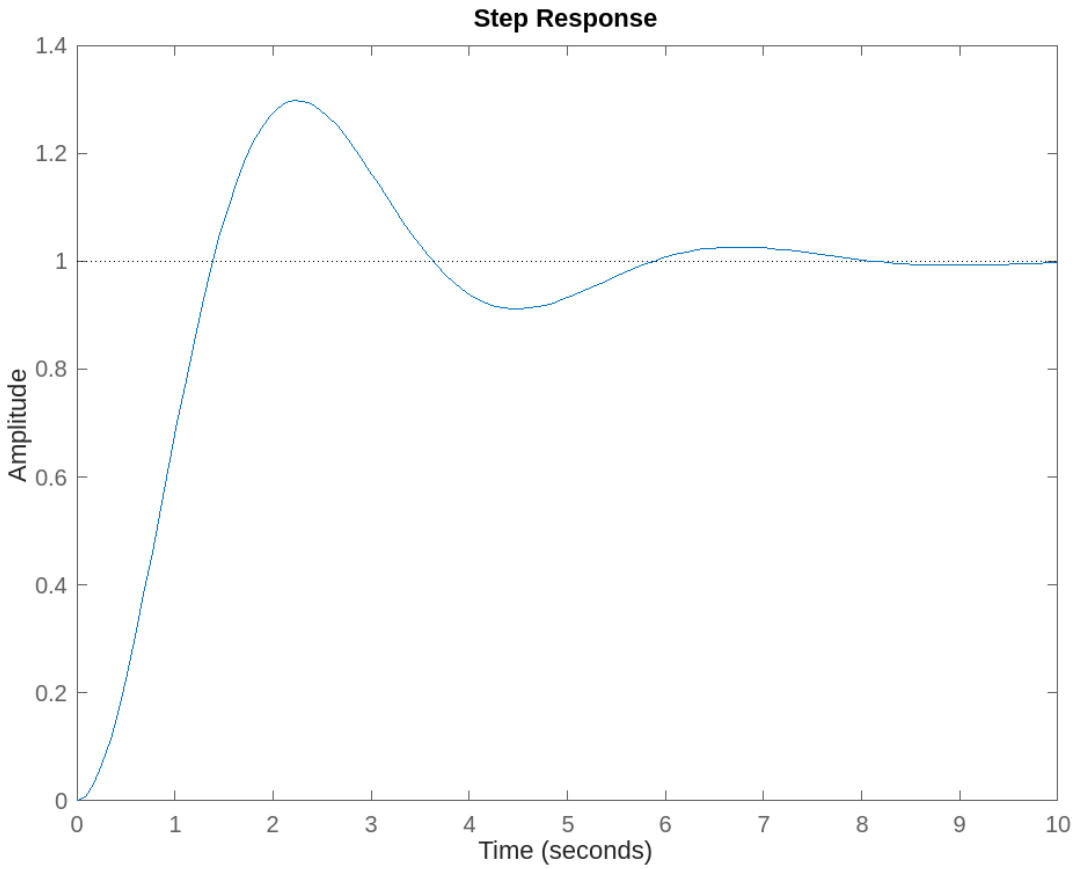


Figure 9: Final step response

4 Results

In this section, we present the results of our control system design for the spacecraft docking application. We begin by deriving the plant functions for both the r-direction and θ-direction. For the r-direction, a PID controller was employed with gains k_p, k_d, and k_i. The open-loop transfer function was optimized using the Nichols-Ziegler method, resulting in an ultimate gain $K_u = 29$ and oscillation time period $T_u = 1.256$ seconds.

Applying constraints for minimal overshoot, the gains were adjusted, leading to $k_p = 0.2K_u$, $k_i = 0.4\frac{K_u}{T_u}$, and $k_d = 0.667\frac{K_u}{T_u}$. The step response of the system exhibited a settling time of 7.70 seconds, a maximum overshoot of 5.34%, and a time when the step response becomes one at 1.94 seconds.

Moving on to the θ-direction, a closed-loop PD controller without an actuator model was utilized. Through derivations based on damping ratio (ζ) and natural frequency (ω_n), we obtained constraints for rise time ($t_r < 1.94$), settling time ($t_s < 7.7$), and maximum overshoot ($M_p = 30\%$). The resulting parameters were $\zeta = 0.36$ and $\omega_n < 1.436$.

Figure 9 depicts the step response for $\zeta = 0.36$ and $\omega_n = 1.5$, showcasing a settling time of 7.37 seconds, a rise time of 1.38 seconds, and a maximum overshoot of 30%.

5 Conclusion

In this paper, we proposed a control system design for spacecraft docking, focusing on both the r-direction and θ-direction control. Through the utilization of PID controllers and PD controllers, we aimed to achieve stable and accurate docking maneuvers while minimizing overshoot and settling time.

The results of our design process indicate successful optimization of the control system parameters, leading to satisfactory performance metrics such as settling time, rise time, and maximum overshoot. By applying the Nichols-Ziegler method and considering constraints on damping ratio and natural frequency, we obtained well-tuned controllers that meet the requirements of the docking task.

Overall, the developed control system demonstrates promising capabilities for spacecraft docking applications, offering a robust and efficient solution for precise maneuvering in space environments. Future work may involve further refinement of the control algorithms and validation through simulations or real-world testing scenarios.

6 Code

Listing 1: Root-locus Matlab code

```
% Define the transfer function
s = tf('s');
sys = 1 / ((0.08*s + 1) * (s^2 + 2*s));
```

```
% Plot figure
figure;
rlocus(sys);
xlabel('Real Axis');
ylabel('Imaginary Axis');
title('Root Locus Plot');
```

Listing 2: Ziegler-Nichols Matlab code

```
s = tf('s');

% Given values
ku = 29;
tu = 2 * pi / 5;

% Controller parameters for P controller
kp1 = 0.6 * ku;
ki1 = 0;
kd1 = 0;

% Controller parameters for PD controller
kp2 = 0.8 * ku;
ki2 = 0;
kd2 = 0.10 * ku * tu;

% Controller parameters for PID controller
kp3 = 0.6 * ku;
ki3 = 1.2 * ku / tu;
kd3 = 0.075 * ku * tu;

% Transfer functions for controllers
con1 = (kp1 * s + kd1 * s^2 + ki1) / s;
con2 = (kp2 * s + kd2 * s^2 + ki2) / s;
con3 = (kp3 * s + kd3 * s^2 + ki3) / s;

% Transfer function for actuator
act = 1 / (0.08 * s + 1);

% Transfer function for plant
plant = 1 / (s^2 + 2 * s);

% Overall transfer functions
op1 = con1 * act * plant;
op2 = con2 * act * plant;
op3 = con3 * act * plant;
```

```
% Closed-loop systems
sys1 = op1 / (1 + op1);
sys2 = op2 / (1 + op2);
sys3 = op3 / (1 + op3);

% Plot step responses
step(sys1, sys2, sys3);
legend('P', 'PD', 'PID');
```

Listing 3: No overshoot Matlab code

```
s = tf('s');
ku = 29;
tu = 2*pi/5;
kp = 0.2 * ku;
kd = 0.4 * ku / tu;
ki = 0.0667 * ku * tu;

% Controller
con = (kp*s + kd*s^2 + ki) / s;

% Actuator
act = 1 / (0.08*s + 1);

% Plant
plant = 1 / (s^2 + 2*s);

% Open-loop transfer function
op = con * act * plant;

% Closed-loop system
sys = feedback(op, 1);

% Step response
[y, t] = step(sys);

% Calculate settling time
settling_time = stepinfo(sys).SettlingTime;
overshoot = stepinfo(sys).Overshoot;

% Find index when step response becomes one
index_one = find(y >= 1, 1);

% Time when step response becomes one
time_to_one = t(index_one);
```

```
% Plot step response
figure;
plot(t, y);
xlabel('Time');
ylabel('Output');
title('Step Response');
grid on;

fprintf('Settling time: %.2f seconds\n', settling_time);
fprintf('Maximum Overshoot: %.2f \n', overshoot);
fprintf('Rise time: %.2f seconds\n', time_to_one);
```

Listing 4: Theta direction step response

```
s = tf('s');

% Define system parameters
w = 1.5; % Natural frequency
zeta = 0.36; % Damping ratio

% Define the transfer function
sys = tf(w^2, [1, 2*zeta*w, w^2]);

% Plot the step response
step(sys);
```

7 References

[1] *Optimum Settings for Automatic Controllers*. URL: https://web.archive.org/web/20170918055307/http://staff.guilan.ac.ir/staff/users/chaibakhsh/fckeditor%5C_repo/file/documents/Optimum%5C%20Settings%5C%20for%5C%20Automatic%5C%20Controllers%5C%20(Ziegler%5C%20and%5C%20Nichols,%5C%201942).pdf.

[2] *Ziegler-Nichols Tuning Rules for PID*. URL: https://www.mstarlabs.com/control/znrule.html.