University of Cape Town

EEE4119F: Mechatronics II

Mishay Naidoo
NDXMIS011

Milestone 4

August 13, 2023

Contents

1 Introduction

This project focused on controlling a rocket to collide with an asteroid heading towards a city on earth to destroy it before it hits the planet. This task involved creating simulation models of both the rocket and the asteroid in MATLab. These models were then used to simulate various rocket control methods to collide with the asteroid multiple scenarios. There were three collision goals tested in simulation:

1. Collide with the asteroid when it was directly above the rocket, requiring the rocket to only fly straight up.
2. Collide with the asteroid when it was past the rocket (because it was moving too fast to hit when directly above the rocket) but as far away from the city as possible.
3. Collide with a specific weak point on the asteroid to ensure that the rocket does the most damage on detonation.

Three different controllers were designed to achieve each goal listed above.

2 Modelling

The following section details the methods used to obtain models of both the rocket and asteroid.

2.1 Rocket Modelling

The rocket used in simulation looked as follows:

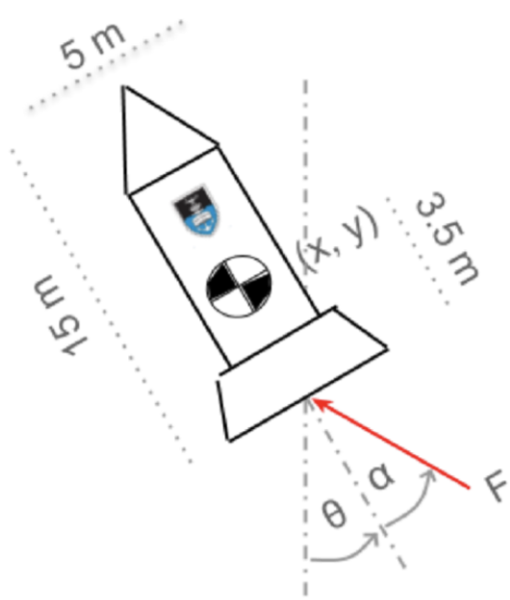

Figure 2.1.1: Diagram of Rocket

The rocket was assumed to not experience drag as it was coated in a 'miracle spray'. The following rocket and asteroid parameters were given in the simulation (these were updated in real time as the simulation ran):

Table 1: Parameters Given in Simulation

Parameter Name	Parameter Description
x	Rocket's x position
dx	Rocket's x velocity
y	Rocket's y position
dy	Rocket's y velocity
th	Rocket's angular position
dth	Rocket's angular velocity
mass	Rocket's mass = 1000kg
ast_x	Asteroid's x position
ast_y	Asteroid's y position
ast_th	Asteroid's angular position
ast_dth	Asteroid's angular velocity
ast_mass	Asteroid's mass = 10000kg
g	gravity = 9.81

Using the information for the diagram in 2.1.1, a manipulator equation was obtained for the rocket. This was done using two frames, frame 1 being the frame of the rocket and frame 0 being frame 1 rotated by theta in the z. The code used to describe the position of the rocket and the thrust force F looked as follows:

```
%Rotation Matrices
R01 = Rotz(ths);
R10 = transpose(R01);


%Position Vectors
r1_1 = [0;0;0]; %Position of Rocket in frame 1
r1_0 = R10*r1_1 + [xs;ys;0]; %Position of Rocket it frame 0

rF1_1 = [0;-3.5;0]; %Position of Force in frame 1
rF1_0 = R10*rF1_1 + [xs;ys;0]; %Position of Force in frame 0

%Force Vectors
```

```
F1_1 = [-F*sin(alpha); F*cos(alpha);0]; %Direction of Force in Frame 1
F1_0 = R10 * F1_1; %Direction of Force in Frame 0
```

The moment of inertia of the rocket was also required to define a manipulator equation and was calculated as follows:

```
%Calculating Moment of Inertia
%This was done using the formula torque = ddth * distance_centre_mass
%Values were input into the simulink model to calculate a numerical J.
%The values were F = 30000 and alpha = 0.008.

ddth = diff(dth)./diff(simulation_time); %Differentiating dth
J_array = abs(30000*sin(0.008)*3.5./ddth(3:end));%Calculating array
of J values from simulation data
J = mean(J_array,"all")%Averaging J_array to obtain final J constant value
```

The asteroid's acceleration was calculated by differentiating its velocity with respect to the simulation time. The above code obtained the moment of inertia of the rocket using its angular acceleration and the moment arm created by the thrust force acting at an angle of $\alpha = 0.08°$ (this value was arbitrarily selected). Using the force position and direction, rocket velocities and moment of inertia the manipulator equation was obtained.

2.2 Asteroid Modelling

The asteroid was not coated in 'miracle spray' and thus experienced a drag force. This drag force was required to effectively model the asteroid and was calculated using the following equations:

$$F_{drag} = c * \sqrt{ast\_dx^2 + ast\_dy^2} \tag{1}$$

$$ast\_ddx = \frac{F_{dragx}}{ast\_mass} + noise \tag{2}$$

$$ast\_ddy = \frac{F_{dragy}}{ast\_mass} - g + noise \tag{3}$$

These equations were implemented in MATLab to calculate the drag coefficient c of the rocket using the follow code:

```
%Calculating C value
c1 = (ast_ddx*ast_mass)./ast_dx(1:end-1);
%Calculating drag coefficient using x velocity and acceleration
c1avg = mean(c1,"all");
c2 = ((ast_ddy(100000:end)+g)*ast_mass)./ast_dy(100000:end-1);
%Calculating drag coefficient using y velocity and acceleration
c2avg = mean(c2,"all");
c = (c1avg+c2avg)/2
```

The c values were calculated using the asteroid's x and y accelerations for the duration of the simulation time (with a bit of matrix manipulation to account for initial transients) and then averaged. The c values calculated using the x and y values were then also added together and averaged. This obtained a final coefficient of c = 93. The noise in the simulation was mitigated by making the simulation time step size very small (0.001).

3 Control Scheme

The rocket was controlled using two variables:

1. α, which controls the angle of the thrust force applied to the rocket.
2. F, which is the thrust force applied to the rocket.

All three controllers used equations of motion with drag obtained from [1]. The equations used were:

$$v_t = ast\_mass * \frac{g}{c} \tag{4}$$

The above equation gives the terminal velocity of the asteroid.

$$x = \frac{ast\_dx * v_t}{g}(1 - e^{\frac{-g*t}{v_t}}) \tag{5}$$

5 gives the x position of the asteroid at a given time t.

$$y = \frac{v_t}{g} * (ast\_dy + v_t)(1 - e^{\frac{-gt}{v_t}}) - v_t t + ast\_y0 \tag{6}$$

6 gives the y position of the asteroid at a given time t.

3.1 Controller for Scenario 1

Scenario 1 uses Equations 4-6 to predict how long it would take for the asteroid to travel from its initial position at the start of the simulation to a position directly above the rocket. This is done by rearranging 5 to solve for time t and making x equal to the difference between the rocket's initial position and the asteroid's initial position:

$$x = rocket\_x0 - ast\_x0 \tag{7}$$

Using this calculated time t, the y position of the asteroid when it is directly above the rocket is found using 6. Using this y value and the time t calculated earlier the required rocket acceleration to meet the asteroid is found using the following equation:

$$rocket\_ddy = \frac{2 * (y - rocket\_dy0 * t)}{t^2} \tag{8}$$

Where y and t are the previously calculated values. This acceleration is then used to find the thrust force required:

$$F = mass * (rocket\_ddy + g) \tag{9}$$

F was made equal to this value and α was made to be 0.

3.2 Controller for Scenario 2

In this scenario, collision directly above the rocket is not possible and thus an x position to the right of the rocket's starting position was chosen. The chosen x coordinate was 300m.

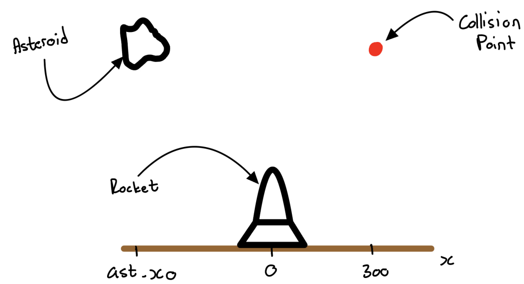

Figure 3.2.1: Diagram of Collision Point for Scenario 2

First, the x distance the asteroid needed to travel to reach the collision point was calculated:

$$x = 300 - ast\_x0 \tag{10}$$

Then, like in Scenario 1, the time t taken for the asteroid to travel this x distance was calculated by rearranging 5. Using this t value, the y position of the collision point was calculated using 6. This gave the x and y coordinates of the collision point, which were used to calculate the angle between the rocket and the collision point as follows:

$$\phi_{collision} = \arctan \frac{y - rocket\_y0}{300 - rocket\_x0} \tag{11}$$

This $\phi_{collision}$ was related to the rockets angle θ:

$$\theta_{ref} = \frac{\pi}{2} - \phi_{collision} \tag{12}$$

The controller used for this scenario then controlled α to make sure that the rocket maintained the angle θ_{ref} until it reached the collision point. This was done using the following equation:

$$\alpha = b * (\theta_{ref} - \theta) \tag{13}$$

Where b was a gain chosen through experimentation to be 0.01. The thrust was chosen to be 40000N to ensure that the collision happened far enough away from the city.

3.3 Controller for Scenario 3

For this scenario, the rocket had to hit the asteroid in a 60-degree band at the front or back of the rocket as shown below:

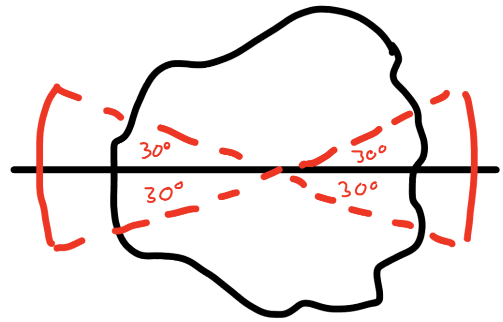

Figure 3.3.1: Required Collision Region for Scenario 3

The control principle for this scenario was exactly the same as for the previous one except that the collision position was determined differently. For this case, the asteroid was moving slow enough that the collision could take place before the asteroid had passed the x starting point of the rocket. It was then determined by inspection, that the rocket would collide with the correct region of the asteroid if the collision occurred when the asteroid had rotated $\frac{\pi}{2} rads$ from its starting orientation. The time taken for the asteroid to rotate the necessary amount was calculated using the following formula:

$$t = \frac{\frac{\pi}{2}}{ast\_dth0} \tag{14}$$

Where ast_dth0 was the initial angular velocity of the asteroid. Using this time t, the x position and y position of the collision were calculated using 5 and 6. The control process for Scenario 2 was then repeated using the new collision coordinates and it was found that a gain b = 0.0025 worked better for this case.

Lastly, for scenario's 1 and 2 the rocket was designed to detonate when it was within 150m of the asteroid.

4 Results and Discussion

The rocket was able to successfully meet the requirements for all three scenarios. The controller was tested for various asteroid 'seeds' where the asteroids behaviour was altered slightly in each seed. Scenario 1 successfully destroyed the asteroid for all seeds tested (roughly 100), Scenario 2 successfully destroyed the asteroid for almost all seeds tested (995 out of 1000). Scenario 3 was only tested using 1 seed and successfully destroyed the asteroid (by hitting its weak spot) for all tests done on that seed (100 runs).

Furthermore, it was desirable for the rocket to destroy the asteroid as far away from the city as possible. The city was located at an x coordinate of roughly 2100m from the starting point of the rocket. In Scenario 1, the asteroid was destroyed 2100m away from the city, in Scenario 2 it was destroyed 1800m away from the city and in Scenario 3 it was destroyed 2300m away from the city.

5 Conclusion

In conclusion, it was found that the method for obtaining the drag coefficient was accurate enough to predict the motion of the asteroid effectively. Furthermore, using kinematic equations to control the rocket was effective in ensuring the rocket reached its desired position in the necessary time. Whilst in all cases the asteroid was destroyed sufficiently far away from the city, this could be further optimised by constantly updating the collision point as the asteroid moved. This is as opposed to using only initial values as was done in the project.

References

[1] R. Fitzpatrick, "Projectile motion with air resistance," Mar 2011. [Online]. Available: https://farside.ph.utexas.edu/teaching/336k/Newton/node29.html