Orysya Stus
MAE 8 Prof. Pham
Winter 2015 Final Project

Projectile Motion of a Tennis Ball on a 3-D Terrain

Overview:

The objective of this project is to track the projectile motion of a tennis ball on a 3-D terrain, using a MATLAB code. The project is accomplished through three tasks: **(1)** a code computes the time *(T)*, maximum and end positions *(X, Y, Z)*, and velocities *(U, V, W)* considering the only gravitational effect on the ball's motion given initial positions (X_o, Y_o, Z_o), initial velocity (U_{mag}), and launch direction (Θ, ϕ) of four trajectories, **(2)** the effects of air resistance on the trajectory of the ball are considered using different coefficients of friction *(C)*, and **(3)** the location of a tennis ball as it hits a 3-D terrain is identified. Note: gravity is set to g = 9.81 m/s^2, Δt = 0.001 second, and five separate .m files were utilized, with **project.m** reporting the requested answers. Through this project, a real-word physics problem involving projectile motion was presented and solved using MATLAB code applicable to any given input and terrain.

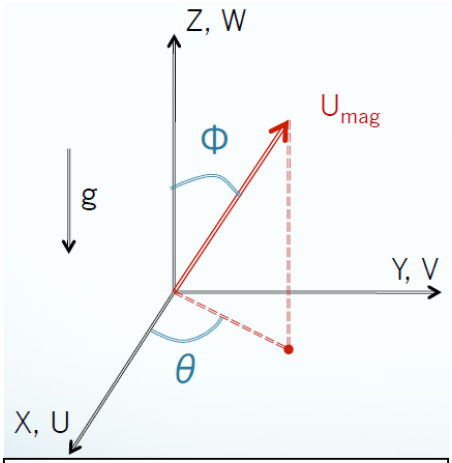


Fig. 1. The parameters utilized to describe the initial and computed trajectories.

Task 1A:

A .m file called **projectile3d** is written to compute the time *(T)*, position *(X, Y, Z)*, and the velocities *(U, V, W)* for given trajectories given initial positions *(Xo, Yo, Zo)*, initial launch velocity (U_{mag}), and launch direction in degrees (Θ, ϕ). The following equations are considered, ignoring air resistance:

$$\frac{dX}{dt} = U \qquad \frac{dU}{dt} = 0$$
$$\frac{dY}{dt} = V \qquad \frac{dV}{dt} = 0$$
$$\frac{dZ}{dt} = W \qquad \frac{dW}{dt} = -g$$

Task 1B:

The initial values of four trajectories are inputted into **projectile3d**.

Trajectory	X_o (m)	Y_o (m)	Z_o (m)	U_{mag} (m/s)	Θ (º)	ϕ (º)
1	0	0	0	40	0	45
2	0	0	0	40	90	45
3	5	5	5	40	30	45
4	5	5	5	40	60	45

Table 1. The initial values of the four trajectories inputted into **projectile3d**.

Utilizing the outputted *[T, X, Y, Z, U, V, W]* values from **projectile3d**, a data structured table is created and reported as table1 in **report.m**. table1 contains four fields: trajectory, time (total), max_height_position, and final_position. The time (total) field corresponds to the time needed for the trajectory to hit the ground. The max_height_position field considers when *Z max* and *W = 0* for each trajectory and lists the associated positions *(X, Y, Z)* and velocities *(U, V, W)* in a 6-element row vector. The final_position field corresponds to the end values of each trajectory and lists them into a 6-element row vector as well.

Trajectory	**time (total)**	**max_height_position [X, Y, Z, U, V, W]**	**final_position [X, Y, Z, U, V, W]**
1	5.768	[81.57, 0, 40.78, 28.28, 0, 0]	[163.12, 0, 0, 28.28, 0, -28.29]
2	5.768	[0, 81.57, 40.78, 0, 28.28, 0]	[0, 163.12, 0, 0, 28.28, -28.29]
3	5.940	[75.64, 45.78, 45.78, 24.49, 14.14, 0]	[150.47, 88.99, 0, 24.49, 14.14, -29.98]
4	5.940	[45.78, 75.64, 45.78, 14.14, 24.49, 0]	[88.89, 150.47, 0, 14.14, 24.94, -29.98]

Table 2. Data structured table1 reported in **report.m**

Since there was no air resistance, the velocities of the four trajectories remained the same until landing. The accelerations in the X and Y direction were zero since the velocities were constant, but the acceleration in the z direction was equal to gravity bringing the ball to the ground.

Task 1C:

Figure1 is created and reported in **report.m** showing projectiles of the four different launches utilizing the outputted *(X, Y, Z)* positions from **projectile3d**.

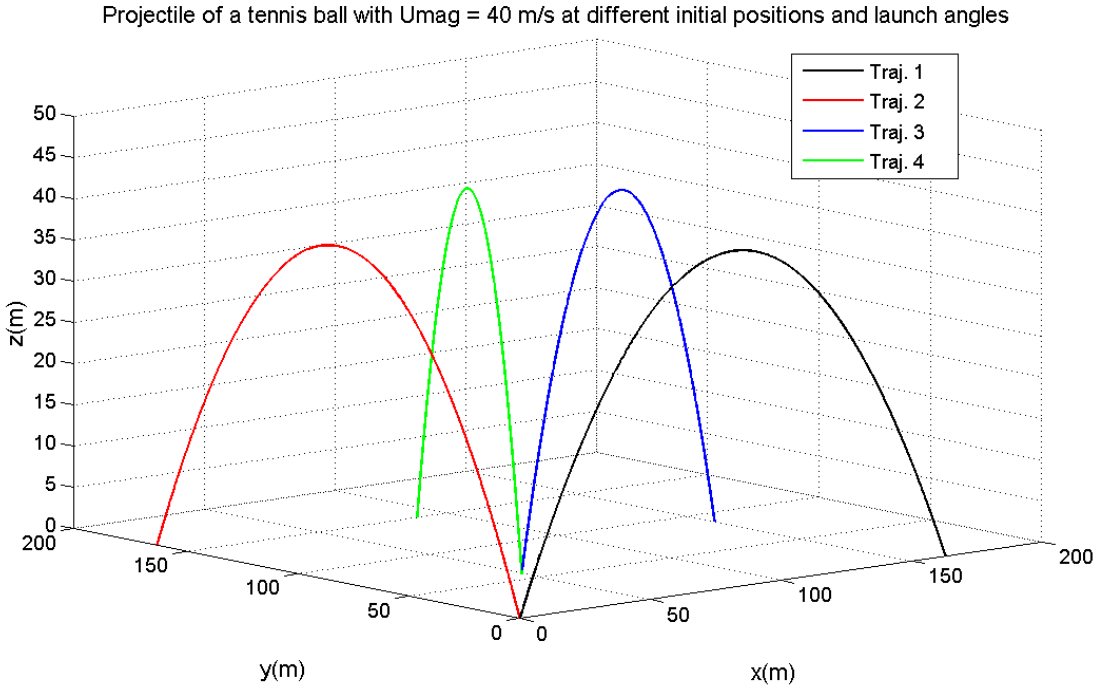


Fig. 2. The trajectories of four different launches.

Task 2A:
A .m file called **projectile3d_2** is a modification of **projectile3d**, and considers the effects of air resistance. Given the same inputs as in **projectile3d**, with the additional of a coefficient of friction *(C)*, the time *(T*), position *(X, Y, Z)*, and the velocities *(U, V, W)* for given trajectories. The following equations are used, accounting for air resistance:

$$\frac{dX}{dt} = U \qquad \frac{dU}{dt} = -C\frac{p_a A}{2m}U\sqrt{U^2+V^2+W^2}$$
$$\frac{dY}{dt} = V \qquad \frac{dV}{dt} = -C\frac{p_a A}{2m}V\sqrt{U^2+V^2+W^2}$$
$$\frac{dZ}{dt} = W \qquad \frac{dW}{dt} = -g - C\frac{p_a A}{2m}W\sqrt{U^2+V^2+W^2}$$

Where $A = \pi r^2$, r = 0.04 m., m = 0.15 kg., p_a = 1.2 kg/m^3 for air.

Task 2B:
The initial values of four trajectories are set as $X_o = Y_o = Z_o = 0$, U_{mag} = 40 m/s, $\Theta = \phi = 45^o$ with four different coefficients of friction: C = 0, 0.2, 0.4, and 0.6 and inputted into **projectile3d_2**.

Utilizing the outputted *[T, X, Y, Z, U, V, W]* values from **projectile3d_2**, a data structured table is created and reported as table2 in **report.m**. table2 contains four fields: trajectory, time (total), max_height_position, and final_position. The time (total) field corresponds to the time needed for the trajectory to hit the ground. The max_height_position field considers when *Z max* and *W = 0* for each trajectory and lists the associated positions *(X, Y, Z)* and velocities *(U, V, W)* in a 6-element row vector. The final_position field corresponds to the end values of each trajectory and lists them into a 6-element row vector as well.

Trajectory	**time (total)**	**max_height_position [X, Y, Z, U, V, W]**	**final_position [X, Y, Z, U, V, W]**
1	5.768	[57.68, 57.68, 40.78, 20.00, 20.00, 0]	[115.34, 115.34, 0, 20.00, 20.00, -28.29]
2	5.147	[42.24, 42.24, 32.61, 15.10, 15.10, 0]	[78.34, 78.34, 0, 11.72, 11.72, -23.12]
3	4.734	[34.18, 34.18, 27.77, 12.63, 12.63, 0]	[60.91, 60.91, 0, 8.50, 8.50, -20.30]
4	4.432	[29.06, 29.06, 24.44, 11.07, 11.07, 0]	[50.46, 50.46, 0, 6.75, 6.75, -18.41]

Table 3. Data structured table2 reported in **report.m**

Due to the effects of air resistance, all of the velocities change between trajectories. The ball's mass and the surface area of the launched ball are considered in order to determine the acceleration experienced by each trajectory.

Task 2C:
Figure2 is created and reported in **report.m** showing projectiles of four different launches associated with different coefficients of friction, utilizing the outputted *(X, Y, Z)* positions from **projectile3d_2**.

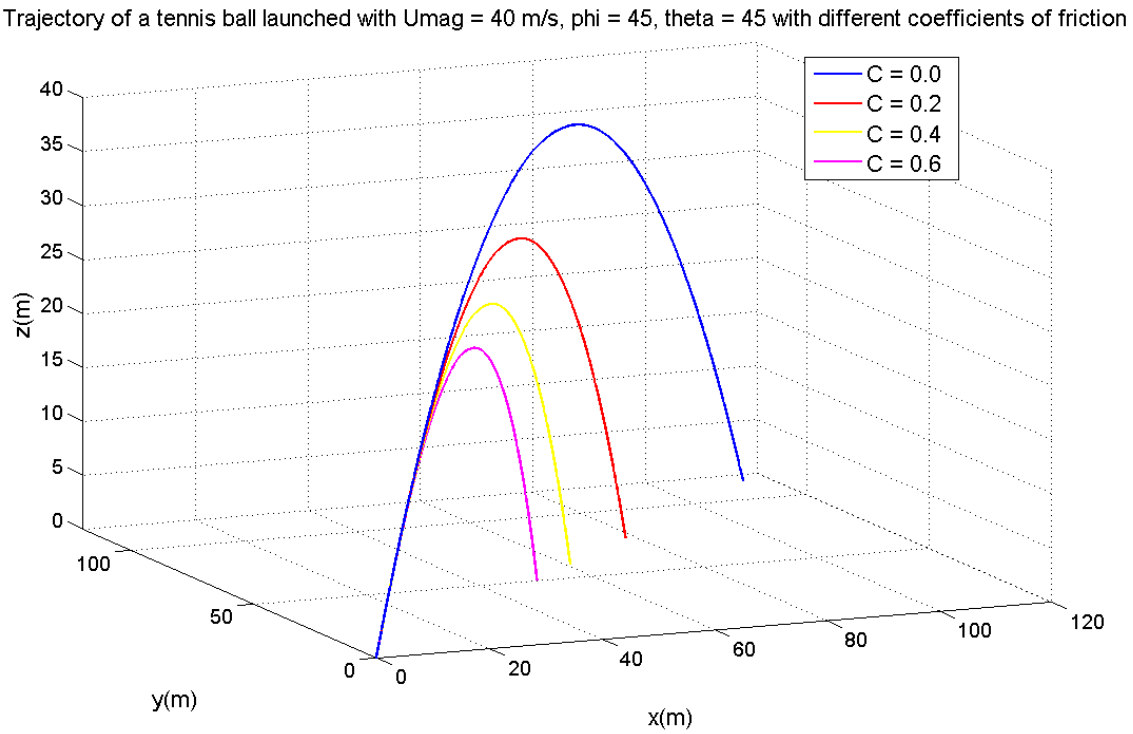


Fig. 3. The trajectories of four different launches with varying coefficients of friction.

Task 3A:
A .m file called **trackprojectile** is written as a modification of **projectile3d_2**, and incorporates a given 3-dimentional terrain as opposed to a flat, level ground used in Task1 and Task2. **trackprojectile** limits the *Z* value of the ball to the height of the terrain. The **terrain.mat** file was downloaded from TED in order to accomplish this task. **trackprojectile** also calls upon the .m file **bilinear** to interpolate *Z* value, considering the *X* and *Y* values of the projectile. The following initial values of four trajectories were inputted into **trackprojectile**

Trajectory	**X_o (m)**	**Y_o (m)**	**Z_o (m)**	**U_{mag} (m/s)**	**Θ (º)**	**ϕ (º)**	**C**
1	0	0	0	40	45	45	0.2
2	1	2	3	40	10	60	0.2
3	2	3	4	40	60	55	0.2

Table 4. The initial values of the four trajectories inputted into **trackprojectile**.

The **trackprojectile.**m file computes the time *(T)*, position *(X, Y, Z)*, and the velocities *(U, V, W)* for the given trajectories.

Task 3B:
Utilizing the outputted *[T, X, Y, Z, U, V, W]* values from **trackprojectile**, a data structured table is created and reported as table3 in **report.m**. table3 contains three fields: trajectory, time (total), and final_position. The time (total) field corresponds to the time needed for the trajectory to hit the ground. The final_position field corresponds to the end values of each trajectory and lists them into a 6-element row vector.

Trajectory	**Time (total)**	**final_position [X, Y, Z, U, V, W]**
1	4.150	[66.01, 66.01, 19.25, 13.01, 13.01, -15.39]
2	3.264	[91.88, 18.02, 10.06, 23.06, 4.06, -13.34]
3	3.646	[50.05, 86.22, 14.00, 10.79, 18.70, -14.60]

Table 5. Data structured table3 reported in **report.m**

Task 3C:
Figure3 is created and reported in **report.m** showing projectiles of three different trajectories on a 3-D terrain, utilizing the outputted *(X, Y, Z)* positions from **trackprojectile**.

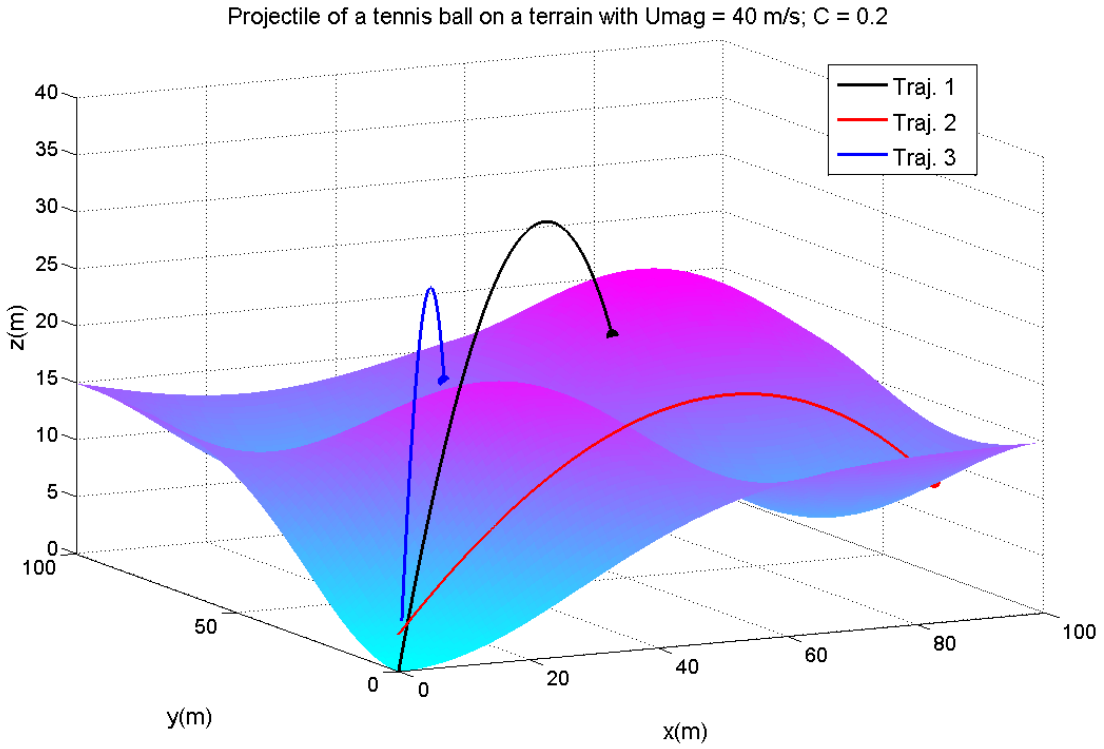
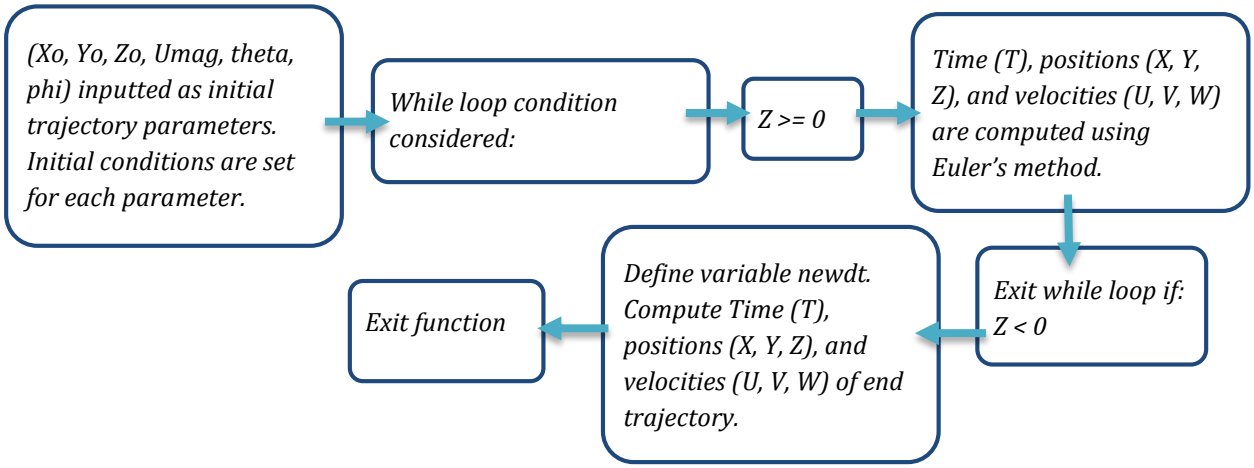


Fig. 4. The trajectories of four different launches over a 3-D terrain (from **terrain.mat** file).

Appendix: Projectile Motion of a Tennis Ball on a 3-D Terrain

report.m
Clears and closes previous contents of the workspace. Outputs 'name', 'id', and 'hw_num'. **report.m** runs the other scripts and function described below, and reports table1-table 3 as well as figure1-figure3.

Task 1:
projectile3d.m



This function uses Euler's method in order to compute the time, position, and velocities of the trajectories given initial position *(Xo, Yo, Zo)*, initial velocity, *(Umag)*, and launch direction *(Θ, ϕ):*

```
function [T, X, Y, Z, U, V, W] = projectile3d(Xo, Yo, Zo, Umag, theta, phi)
```

Initial conditions are established with time interval $\Delta t = 0.001$ second. This function ignores air resistance and only considers gravitational effect where $g = 9.81\ m^2/s$. The initial inputs enter a while loop if the condition *Z >= 0* is satisfied, then the trajectory outputs are computed. If *Z <0* (when the ball hits the ground), the while loop ends and an if statement is considered to compute the end trajectory outputs using a newly defined variable called newdt.

The following is reported in **report.m**:
The following initial parameters for four different trajectories are inputted into **projectile3d**. The corresponding outputs of *[Tn, Xn, Yn, Un, Vn, Wn]* are computed with n corresponding to the trajectory number and used to create table1 and figure 1.

```
[T1, X1, Y1, Z1, U1, V1, W1] = projectile3d(0,0,0,40,0,45);
[T2, X2, Y2, Z2, U2, V2, W2] = projectile3d(0,0,0,40,90,45);
[T3, X3, Y3, Z3, U3, V3, W3] = projectile3d(5,5,5,40,30,45);
[T4, X4, Y4, Z4, U4, V4, W4] = projectile3d(5,5,5,40,60,45);
```

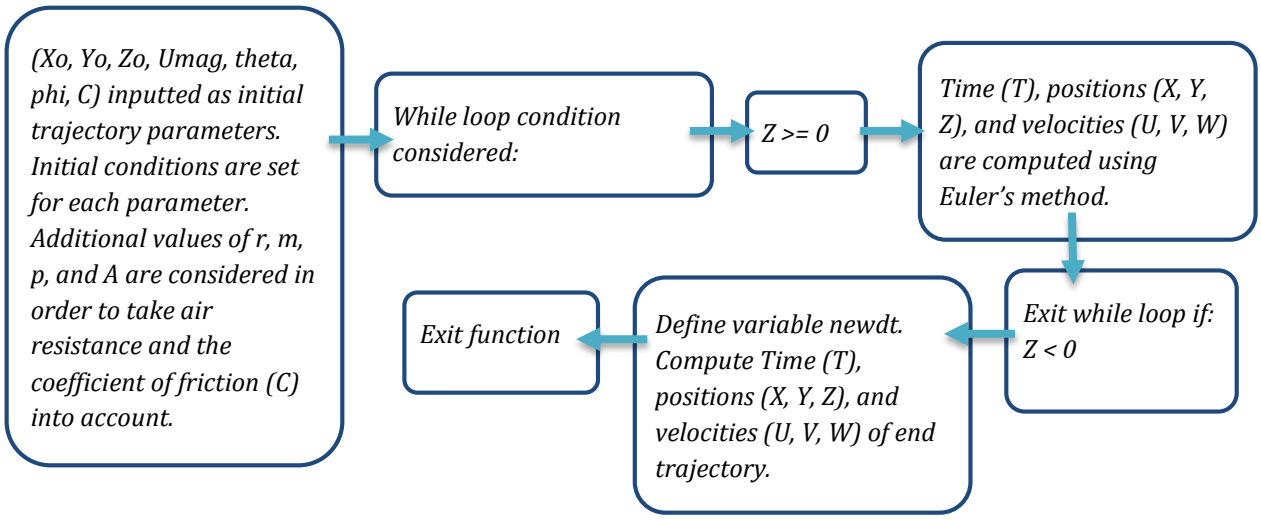
table1 is a data structured table which utilizes the MATLAB function struct. table1 is a 4-element with the following field names: trajectory, time, max_height_position, and

final_position. A variable i is defined in order to find the index at which *Zn* is at the maximum, using this index the corresponding *Xn* and *Yn* positions as well as velocities (*U, V, W*) are found. Below is the code for table1(1) associated with the first trajectory:

```
i1 = find(Z1 == max(Z1(:)));
table1(1) = struct('Trajectory', 1, 'total_time',
T1(end),'max_height_position', [max(X1(i1))],'final_position', [X1(end)]);
table1(1).max_height_position = [max(X1(i1)), max(Y1(i1)), max(Z1),(U1(i1)),
(V1(i1)), (W1(i1))];
table1(1).final_position = [X1(end), Y1(end), Z1(end), U1(end), V1(end),
W1(end)];
```

figure1 uses the position outputs *(Xn, Yn, Zn)* from **projectile3d** for each trajectory and plots the projectiles for each trajectory using the MATLAB function plot3. Each trajectory is distinguished by a different color and the title, legend, and axes (Xrange: [0, 200], Yrange: [0, 200], and Zrange: [0, 50]) are labeled.

Task 2:
projectile3d_2.m



Like **projectile3d**, this function uses Euler's method in order to compute the time, position, and velocities of the trajectories given initial position *(Xo, Yo, Zo)*, initial velocity, *(Umag)*, launch direction (Θ, ϕ), as well as a coefficient of friction *(C)*:

```
function [T, X, Y, Z, U, V, W] = projectile3d_2(Xo, Yo, Zo, Umag, theta, phi, C)
```

Initial conditions are established with time interval $\Delta t = 0.001$ second. This function takes air resistance into account by utilizing a new initial trajectory *C*, which corresponds to the coefficient of friction. The initial inputs enter a while loop if the condition $Z >= 0$ is satisfied, then the trajectory outputs are computed. If $Z < 0$ (when the ball hits the ground), the while loop ends and an if statement is considered to compute the end trajectory outputs using a newly defined variable called newdt.

In order to account for the effects of air resistance, *U, V, and W* were changed to include the additional variable affecting air resistance. A comparison between **projectile3d.m** and **projectile3d_2.m** is made, considering the variable *V*:

projectile3d.m:
```
V(n+1)=V(n);
```

projectile3d_2.m:
```
V(n+1)=V(n) +((-C*p*A/(2*m)) * V(n) * ((U(n)^2 + V(n)^2 + W(n)^2)^(1/2)))*dt;
```

The following is reported in **report.m**:
The following initial parameters for four different trajectories are inputted into **projectile3d_2**. The corresponding outputs of *[Tn, Xn, Yn, Un, Vn, Wn]* are computed with n corresponding to the trajectory number and used to create table2 and figure 2.

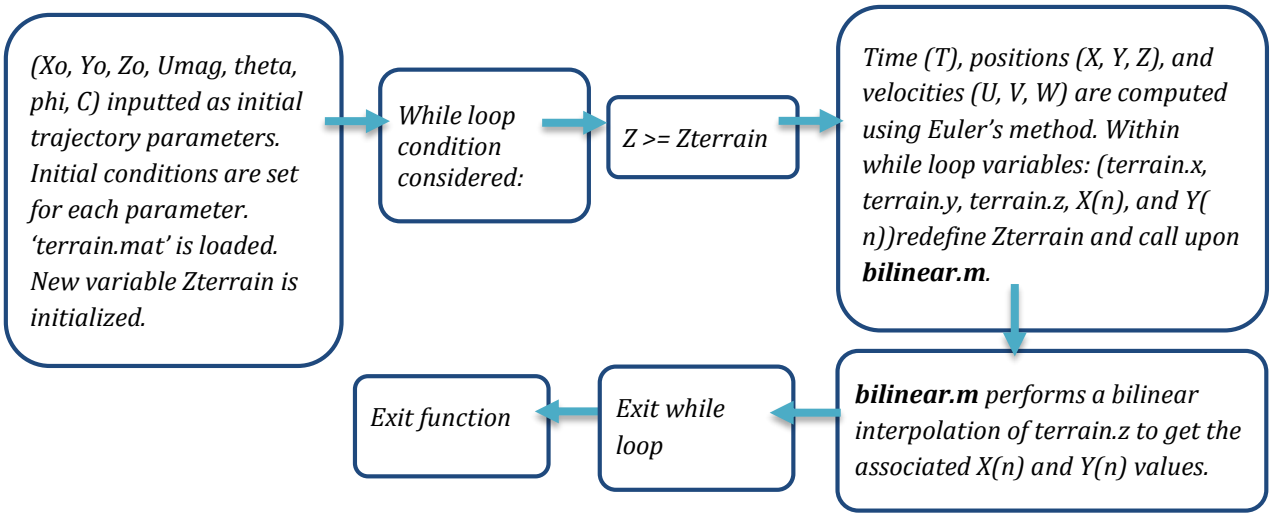
```
[t1, x1, y1, z1, u1, v1, w1] = projectile3d_2(0,0,0,40,45,45,0.0);
[t2, x2, y2, z2, u2, v2, w2] = projectile3d_2(0,0,0,40,45,45,0.2);
[t3, x3, y3, z3, u3, v3, w3] = projectile3d_2(0,0,0,40,45,45,0.4);
[t4, x4, y4, z4, u4, v4, w4] = projectile3d_2(0,0,0,40,45,45,0.6);
```

table2 is a data structured table which utilizes the MATLAB function struct. table2 is a 4-element with the following field names: trajectory, time, max_height_position, and final_position. A variable *i* is defined in order to find the index at which z*n* is at the maximum, using this index the corresponding *xn* and *yn* positions as well as velocities (*u, v, w)* are found. Below is the code for table2(1) associated with the first trajectory:

```
i1 = find(z1 == max(z1(:)));
table2(1) = struct('Trajectory', 1, 'total_time',
t1(end),'max_height_position', [max(x1(i1))],'final_position', [x1(end)]);
table2(1).max_height_position = [max(x1(i1)), max(y1(i1)), max(z1),(u1(i1)),
(v1(i1)),(w1(i1))];
table2(1).final_position = [x1(end), y1(end), z1(end), u1(end), v1(end),
w1(end)];
```

figure2 uses the position outputs *(xn, yn, zn)* from **projectile3d** for each trajectory and plots the projectiles for each trajectory using the MATLAB function plot3. Each trajectory is distinguished by a different color and the title, legend, and axes (Xrange: [0, 120], Yrange: [0, 120], and Zrange: [0, 40]) are labeled.

Task 3:
trackprojectile.m



Like the previous functions, this function uses Euler's method in order to compute the time, position, and velocities of the trajectories given initial position (Xo, Yo, Zo), initial velocity, (Umag), launch direction (Θ, ϕ), and coefficient of friction *(C)*:

```
function [T, X, Y, Z, U, V, W] = trackprojectile(Xo, Yo, Zo, Umag, theta, phi, C)
```

projectile3d.m and **projectile3d_2.m** assumed that the tennis ball was landed on an even, flat ground, while **trackprojectile.m** takes into account an uneven terrain onto which the ball lands. Initial conditions are established with time interval $\Delta t = 0.001$ second. The variable *Zterrain* is defined and initialized. The initial inputs enter a while loop if the condition *Z >= Zterrain* is satisfied, then the trajectory outputs are computed. Within the while loop, *Zterrain* is redefined by calling upon a separate function called **bilinear.m**:

```
Zterrain = bilinear(terrain.x, terrain.y, terrain.z, X(n), Y(n));
```

bilinear.m
The standard input format for **bilinear.m**:

```
function [Tp] = bilinear(x,y,T,Px,Py)
```

This function performs bilinear interpolation of *T(x,y)* to get *Tp (Px,Py).* The **trackprojectile.m** while loop inputs the values of *terrain.x* and *terrain.y* from the loaded 'terrain.mat' file and *X(n+1)* and *Y(n+1)* from the while loop to interpolate the *terrain.z* value and compute a new value for *Zterrain* using **bilinear.m**.

The following is reported in **report.m**:
The following initial parameters for four different trajectories are inputted into **trackprojectile.m**. The corresponding outputs of *[Tn, Xn, Yn, Un, Vn, Wn]* are computed with n corresponding to the trajectory number and used to create table3 and figure 3.

```
[T1, X1, Y1, Z1, U1, V1, W1] = trackprojectile(0,0,0,40,45,45,0.2);
[T2, X2, Y2, Z2, U2, V2, W2] = trackprojectile(1,2,3,40,10,60,0.2);
[T3, X3, Y3, Z3, U3, V3, W3] = trackprojectile(2,3,4,40,60,55,0.2);
```

table3 is a data structured table which utilizes the MATLAB function struct. table3 is a 3-element with the following field names: trajectory, time, and final_position. Below is the code for table3(1) associated with the first trajectory:

```
table3(1) = struct('Trajectory', 1, 'total_time', T1(end),'final_position',
[X1(end)]);
table3(1).final_position = [X1(end), Y1(end), Z1(end), U1(end), V1(end),
W1(end)];
```

figure3 uses the position outputs *(Xn, Yn, Zn)* from **trackprojectile.m** for each trajectory and plots the projectiles and end points (distinguished by a large marked circle) for each trajectory using the MATLAB function plot3. Each trajectory is distinguished by a different color and the title, legend, and axes (Xrange: [0, 100], Yrange: [0, 100], and Zrange: [0, 40]) are labeled. Also, the 'terrain.mat' file is loaded and plotted on the same graph using the following commands:

```
hold on;
b = surf(terrain.x, terrain.y, terrain.z);
set (b, 'edgecolor', 'none');
```

figure3 shows that the position outputs for each inputted trajectory of **trackprojectile.m** does not cross below the surface of the terrain.