**AQ3: Soccer Ball over a Wall and Net**

**Miguel A. Morales**

**113111388**

**Objective:** Create a program that calculates the trajectory of a free kick going over a wall, into a soccer net. This program will take into account air resistance and the swerve of the ball.

**Challenging objective:** Incorporate a goalkeeper that defends the soccer net, and that reacts intelligently to the progressive movement of the ball (taking into account his/her reaction time). Create an intelligent wall that appropriately places itself with regards to the placement of the free kick.

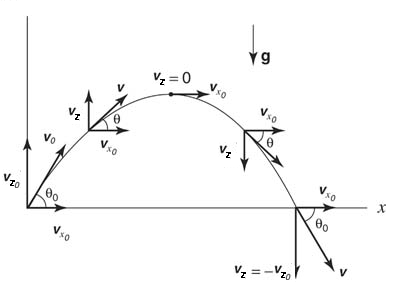
**Variables**

|  |  |  |  |
| --- | --- | --- | --- |
| **t\_step** | time step between each calculation | **r** | radius of the soccer ball |
| **num\_step** | the number of steps | **p** | average density of the soccer ball |
| **theta\_deg** | angle of the kick in degrees | **C** | drag coefficient |
| **theta\_rad** | angle of the kick in radians | **A** | silhouette area of soccer ball |
| **w** | angular velocity on ball | **D** | Constant (p\*C\*A) |
| **s0** | average drag coeff | **m** | mass of soccer ball |
| **a\_x** | initial acceleration in the x direction | **t\_vals** | Time array |
| **a\_y** | initial acceleration in the y direction | **ax\_vals** | x-acceleration array |
| **a\_z** | initial acceleration in the z direction | **ay\_vals** | y-acceleration array |
| **v\_0** | initial magnitude of the velocity | **az\_vals** | z-acceleration array |
| **v\_0x** | initial velocity in the x direction | **vx\_vals** | x-velocity array |
| **v\_0y** | initial velocity in the y direction | **vy\_vals** | y-velocity array |
| **v\_0z** | initial velocity in the z direction | **vz\_vals** | z-velocity array |
| **d\_0x** | initial displacement in the x direction | **dx\_vals** | x-position array |
| **d\_0y** | initial displacement in the y direction | **dy\_vals** | y-position array |
| **d\_0z** | initial displacement in the z direction | **dz\_vals** | z-position array |
| **check** | check if the ball has gone in the net | **final** | Final array value |
| **check\_wall** | check to see if the ball has hit the wall | **v\_final** | Final velocity |

**2-D Trajectory with air resistance**

The Newtonian kinematic equations will be used to calculate the displacement and velocity of the soccer ball. The x-direction will be horizontal displacement of the ball, and the z-direction will the vertical displacement of the ball. We will be calculating the trajectory of a ball with initial velocity and a rise angle of. The free kick will take place 30m from the soccer net.

[Figure 1]



The trajectory of the ball will be created using an Euler loop that will incorporate the following velocity equations:

The Initial values for and will be given by and as can be seen by [Figure 1]. The total velocity will also be calculated within the loop in the form . Note that and in the case without air resistance.

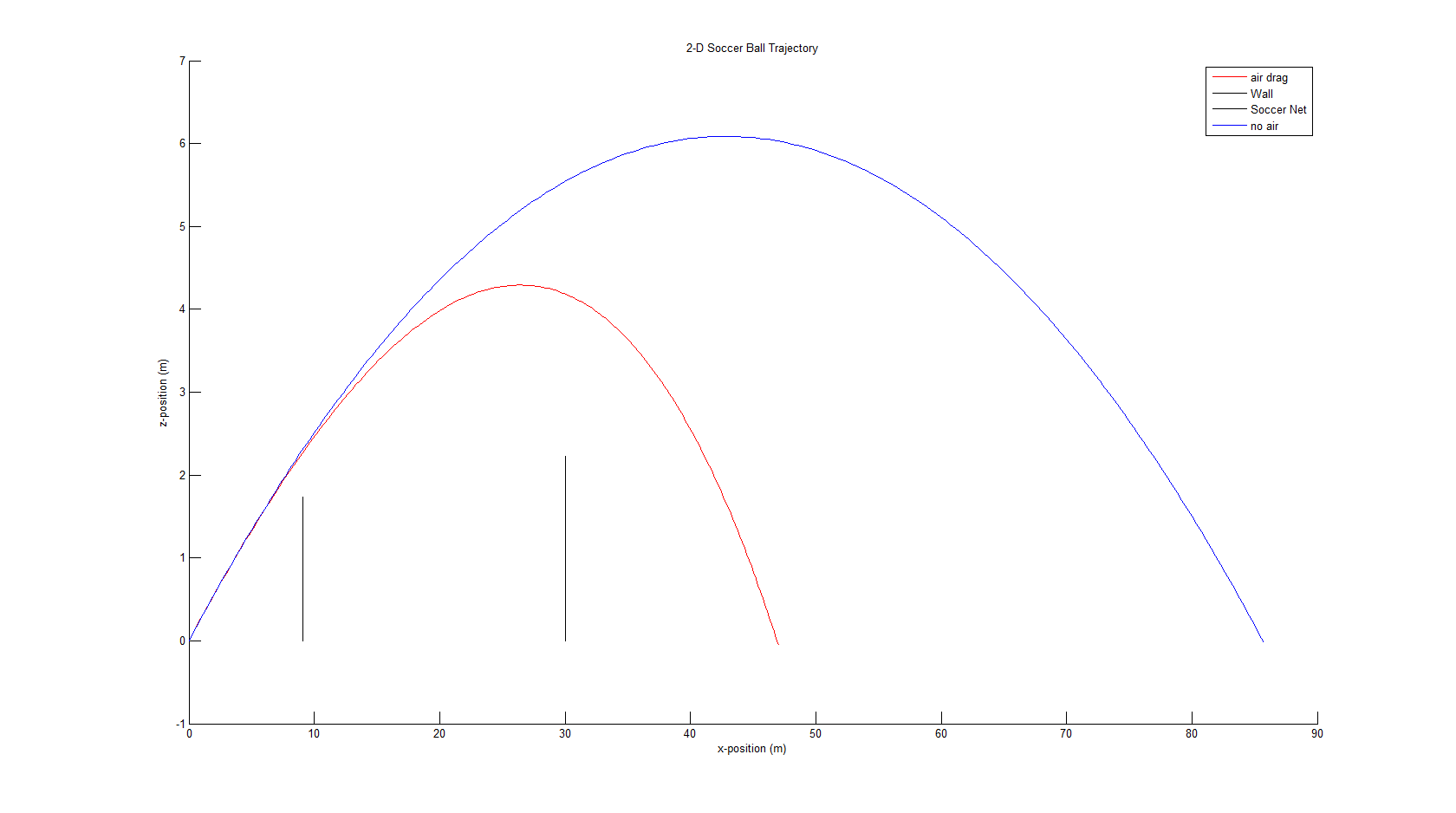
Air resistance will be taken into account by including the drag factor (shown in red) into the acceleration arrays in the following way:

Where is the drag coefficient and is the mass of the soccer ball.

These velocity and acceleration arrays will be used to calculate the displacement of the ball, which will be included in the same Euler loop. The displacement will be calculated by:

Finally, we will add two vertical lines using the *line()* function: one being the “wall” and the other being the soccer net. The wall will be placed according to regulations (10 yards from the kick) and the net will be placed 30 meters from the kick. The trajectory of the soccer ball with – and without – air drag can be seen in the [Figure 2]

[Figure 2]



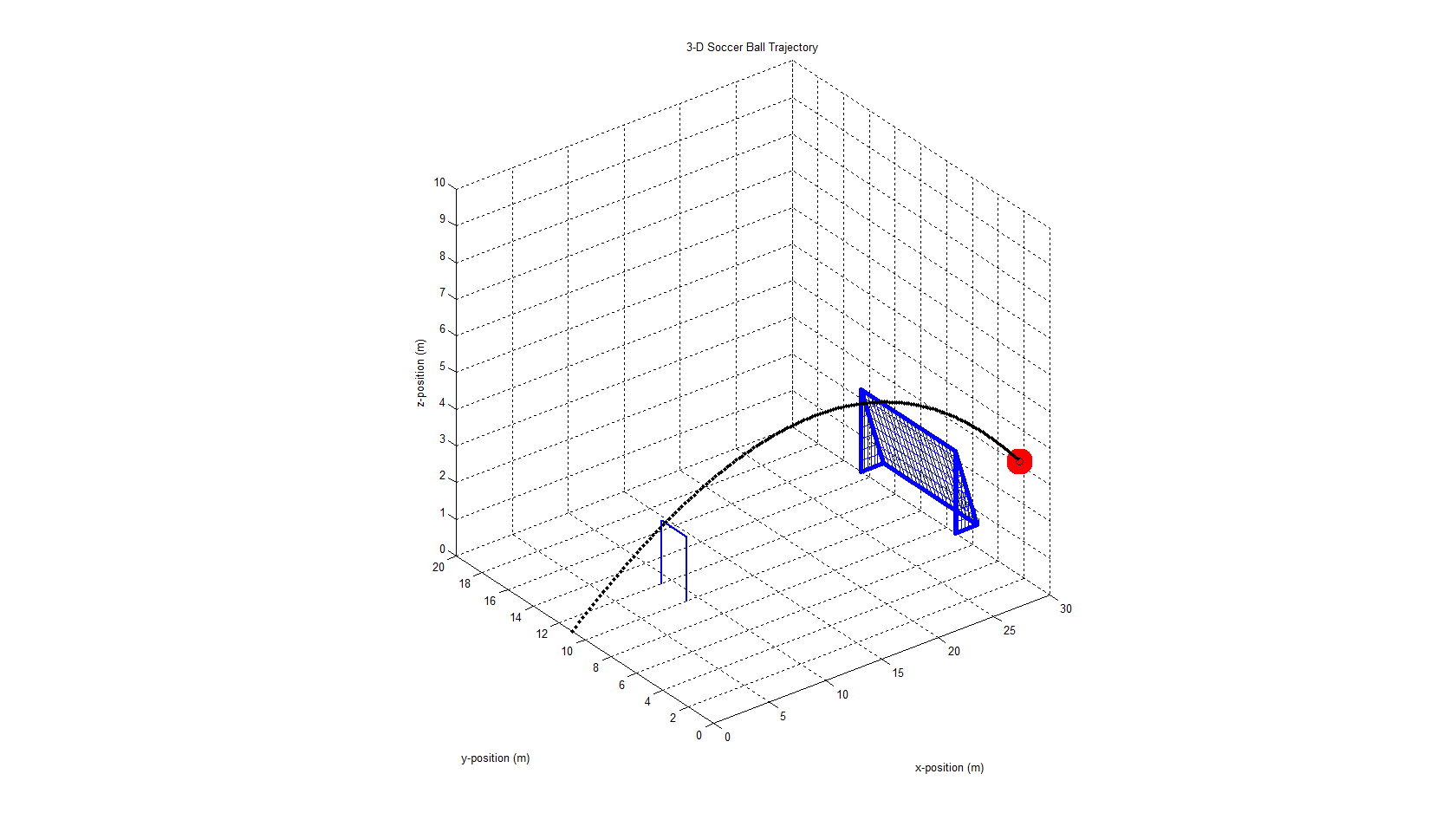
*A soccer ball of radius 0.101 m [1], with a density of 1.6 kg/m3 [4], a mass of 0.43 kg[5], and a coefficient of drag of 0.25 [3] was used for the simulations.*

**3-D Trajectory (with swerve)**

In order to create a three-dimensional plot, three additional arrays must be created:

Additional lines are also created in order to render a three-dimensional wall and soccer net. The three-dimensional plot can be observed in [Figure 3].

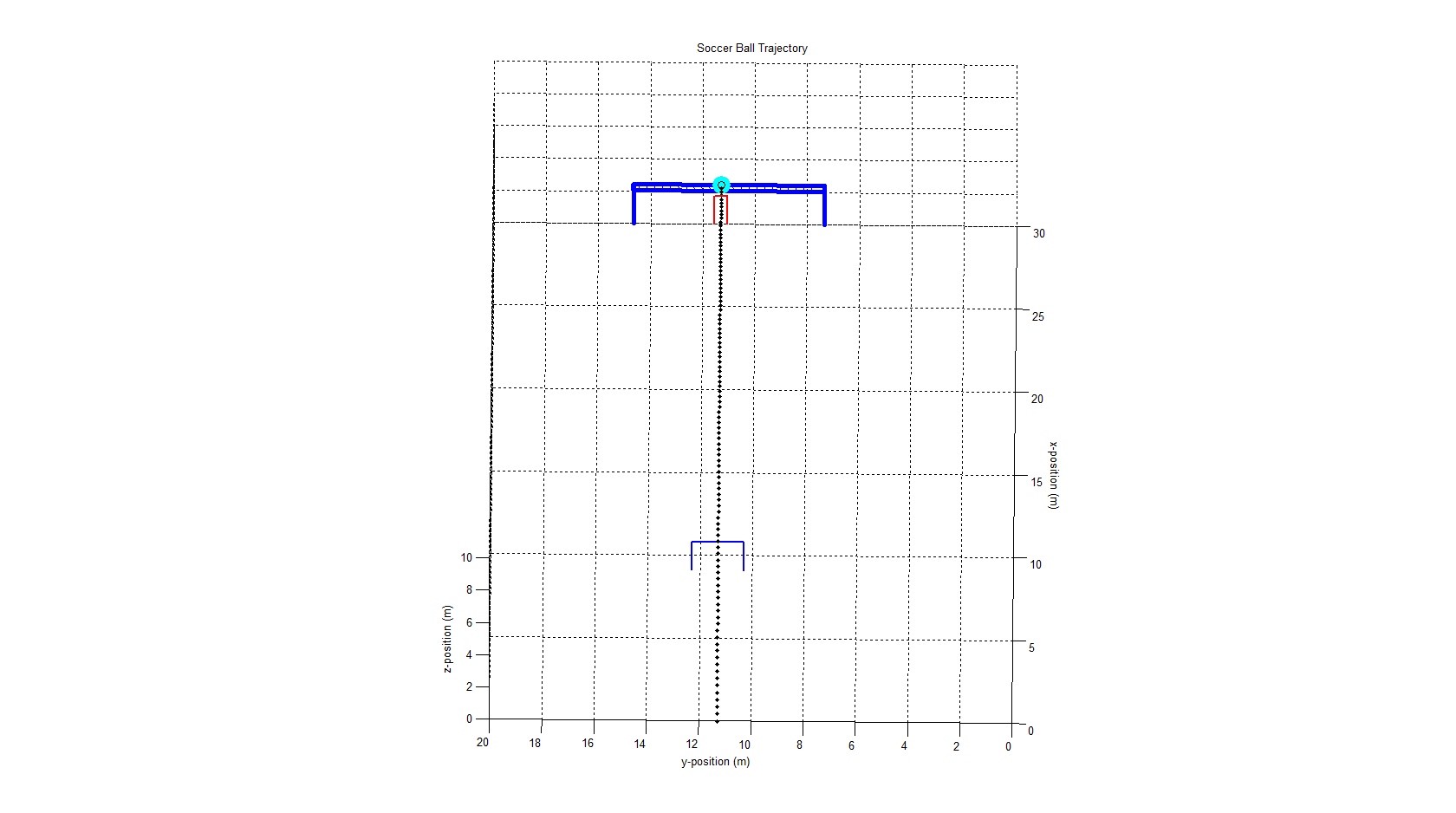
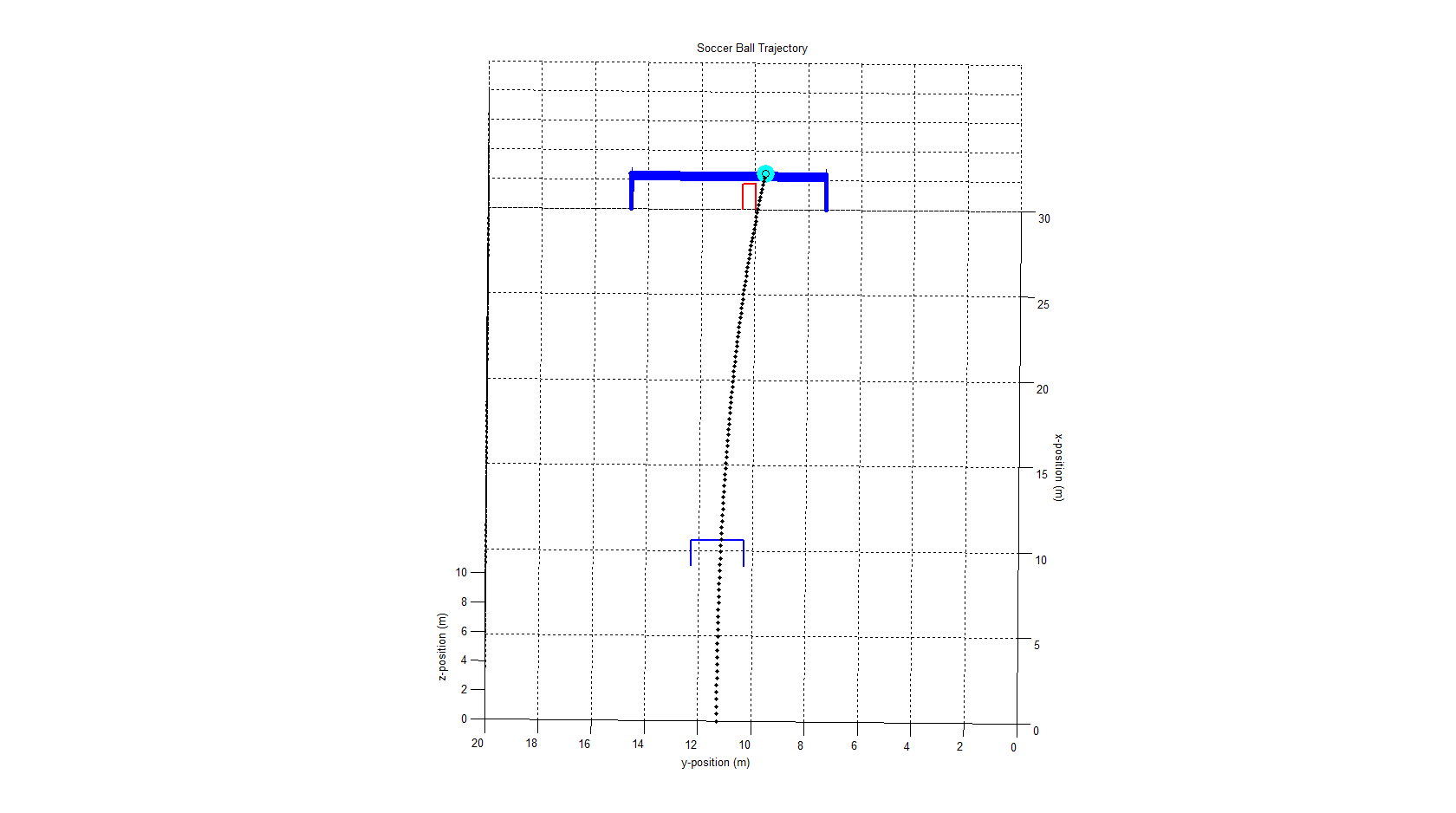
[Figure 3]



We then account for the swerve of the ball by including the contribution of the Magnus Force towards the acceleration in the y-direction in the following manner:

A bird’s-eye view of the soccer ball without swerve can be observed in [Figure 4] and the trajectory that *includes* swerve can be observed in [Figure 5].

[Figure 4] [Figure 5]

*Without swerve With swerve (20 rotations/s)*

*A soccer net of regulation size 2.44m x 7.32m [6], as well as a four man wall (assuming standard height of 1.74m and width of 0.5m per person) was used for the simulations.*

**Goalkeeper and Wall AI**

In order to create a realistic simulation of a free-kick, it is important to include a goalkeeper! For the sake of simplicity we will use lines to create a vertical rectangle in front of the soccer net; this will be the goalkeeper. In order to create a “smart” and realistic goalkeeper, certain conditions will be put into place: the goalkeeper will not step past the boundaries of the soccer net, he/she will react to the progressive direction and velocity of the ball, and his/her reaction time will be taken into account. The simplistic nature of this AI will also bring forward some limitations: he/she will not be able to dive or jump upwards to intercept the ball.

In order for this goalkeeper to work (and move) accordingly we will use a conditional if statement:

if (right\_lim > 7.35 && left\_lim < 14.65 && i > react\_step )

where the right\_lim and left\_lim are variables that define the right and left side of the goalkeeper and react\_step is the number of time steps that make up the goalkeeper’s reaction time. Once these conditions have been met, the goalkeeper is finally allowed to move by redefining right\_lim and left\_lim .

left\_lim=d\_0z+0.25-d\_0z+d\_z(i-react\_step);

right\_lim =d\_0z-0.25-d\_0z+d\_z(i-react\_step);

Note that the right and left limits are moving relative to the ball’s direction, thus in a situation where the ball is first moving outwards then back in (strong swerve) the goalkeeper will react accordingly. Once the limits have been redefined, the lines that make up the rectangle are redrawn accordingly. The loop then continues.

Implementing an intelligent wall is a significantly easier due to the fact that it must remain static during the free-kick (assuming that the players do not jump). The wall will be made to position itself within regulation distance from the free kick regardless of its position on pitch. If the ball is placed within the penalty area, no wall will be implemented (we will ignore that penalty kicks must be taken from 12 yards or 15m).

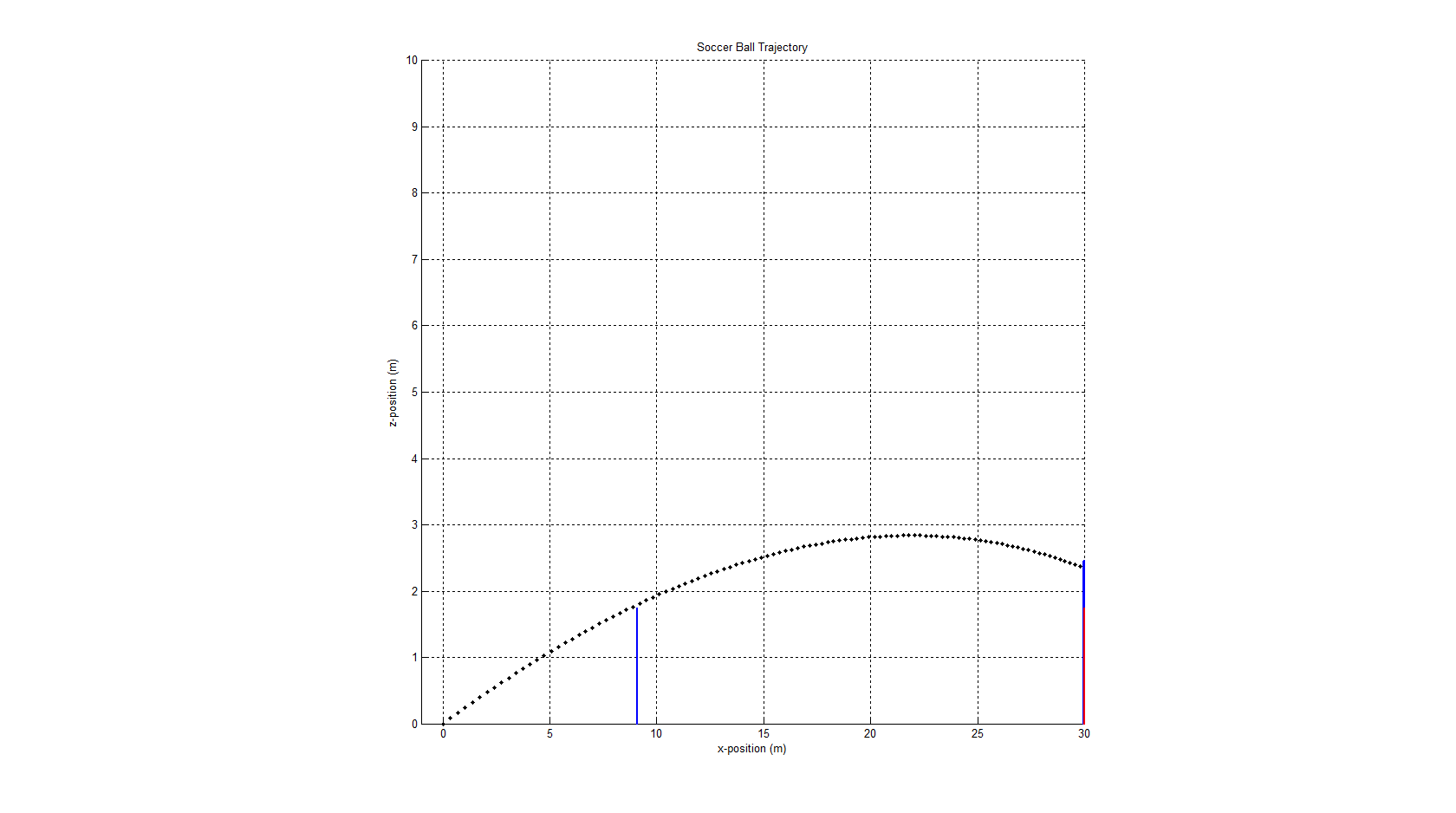
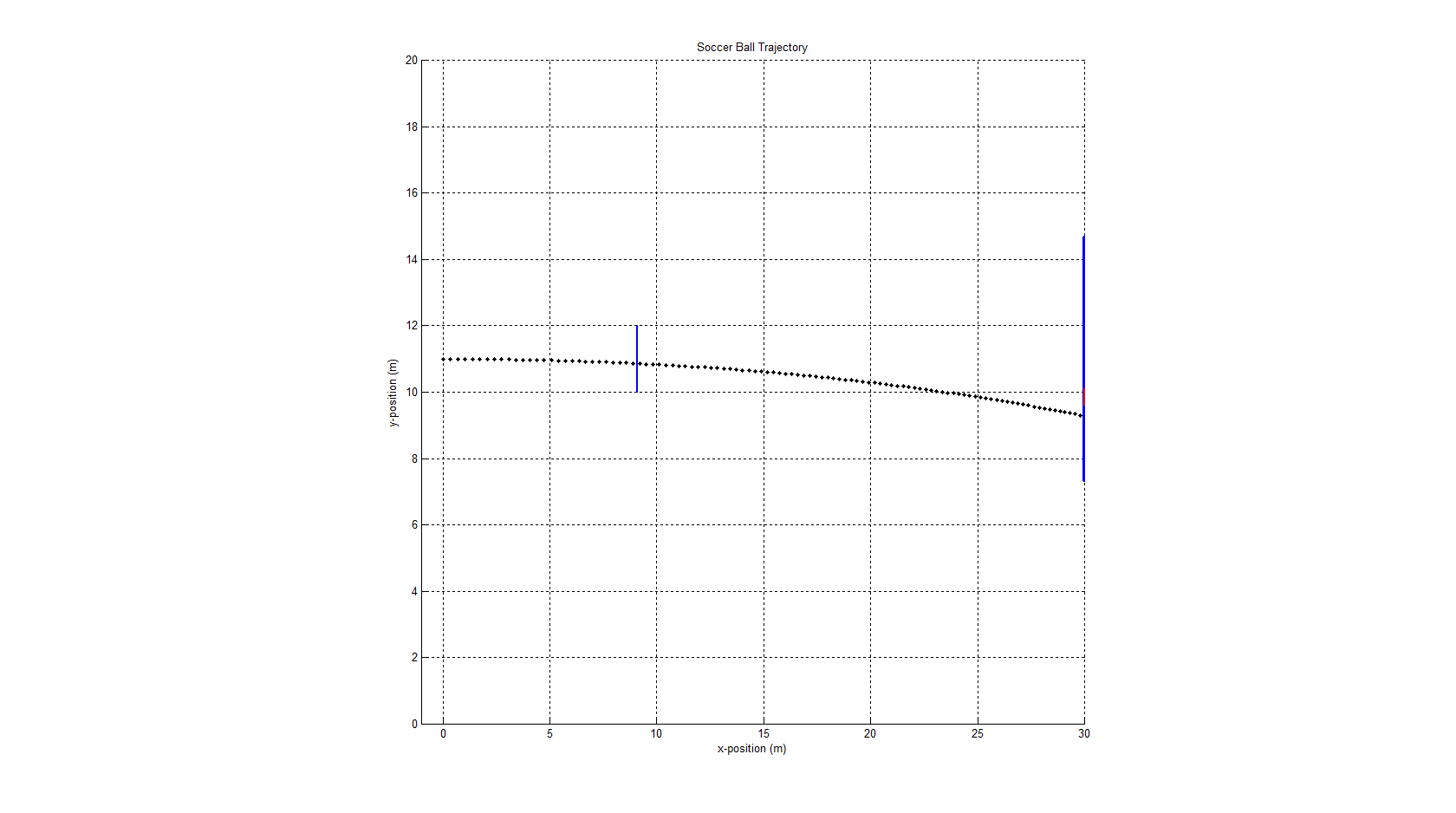
**Implementation of the Program**

This program can be used to simulate a multitude of different possible free kicks or penalty shots.

We will show the versatility of the program by displaying some of the simulated shots. In the following demonstrations, 3 different point-of-views will be shown as well as he corresponding print out from the program and the initial variables. Note that a cyan circle will be shown if a goal is scored and a red circle will be shown if the shot is unsuccessful.

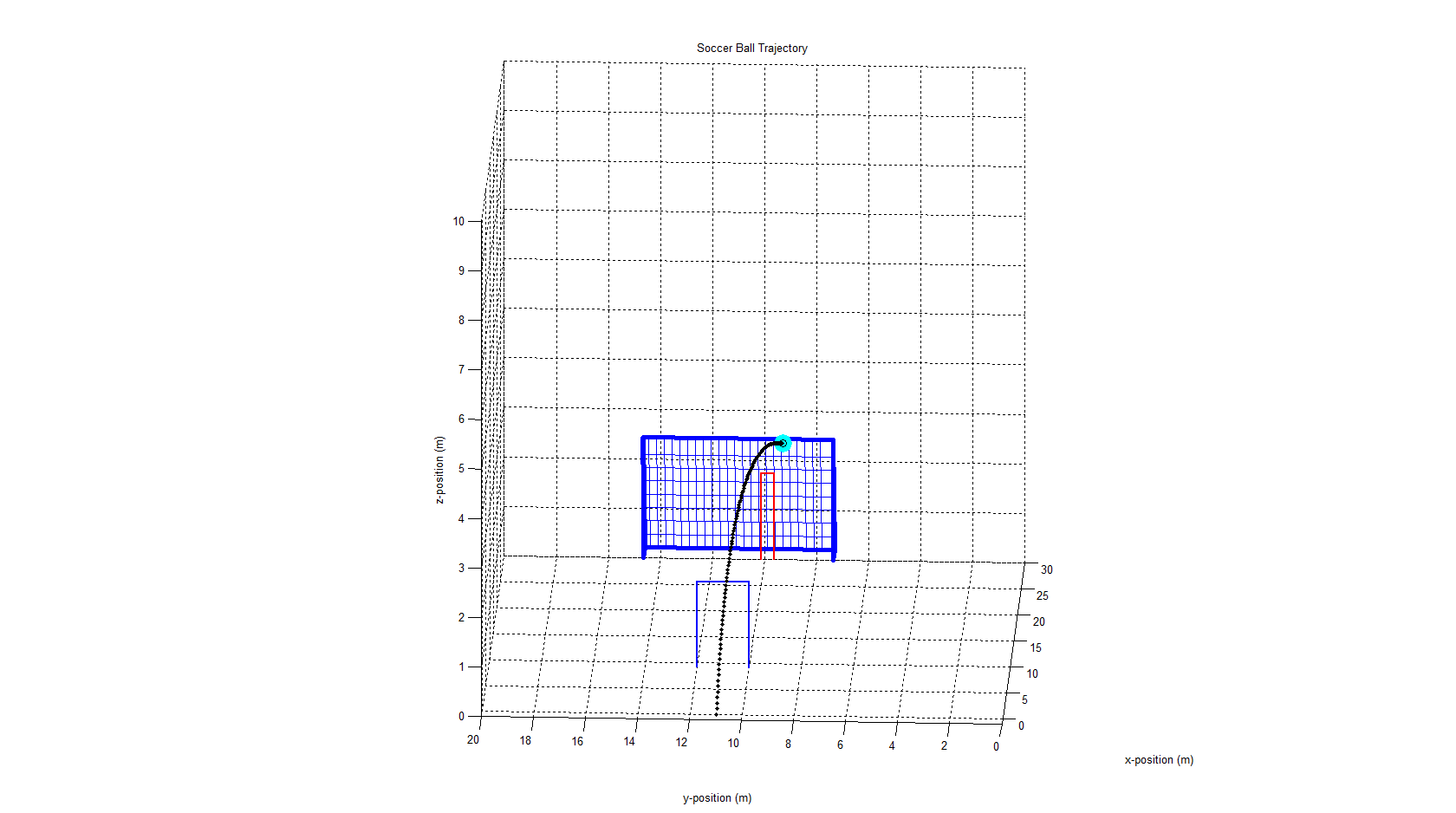
1. **Free kick from 30 meters**

[Figure 6] [Figure 7]

 ****

*x-z plane view x-y plane view*

[Figure 8]



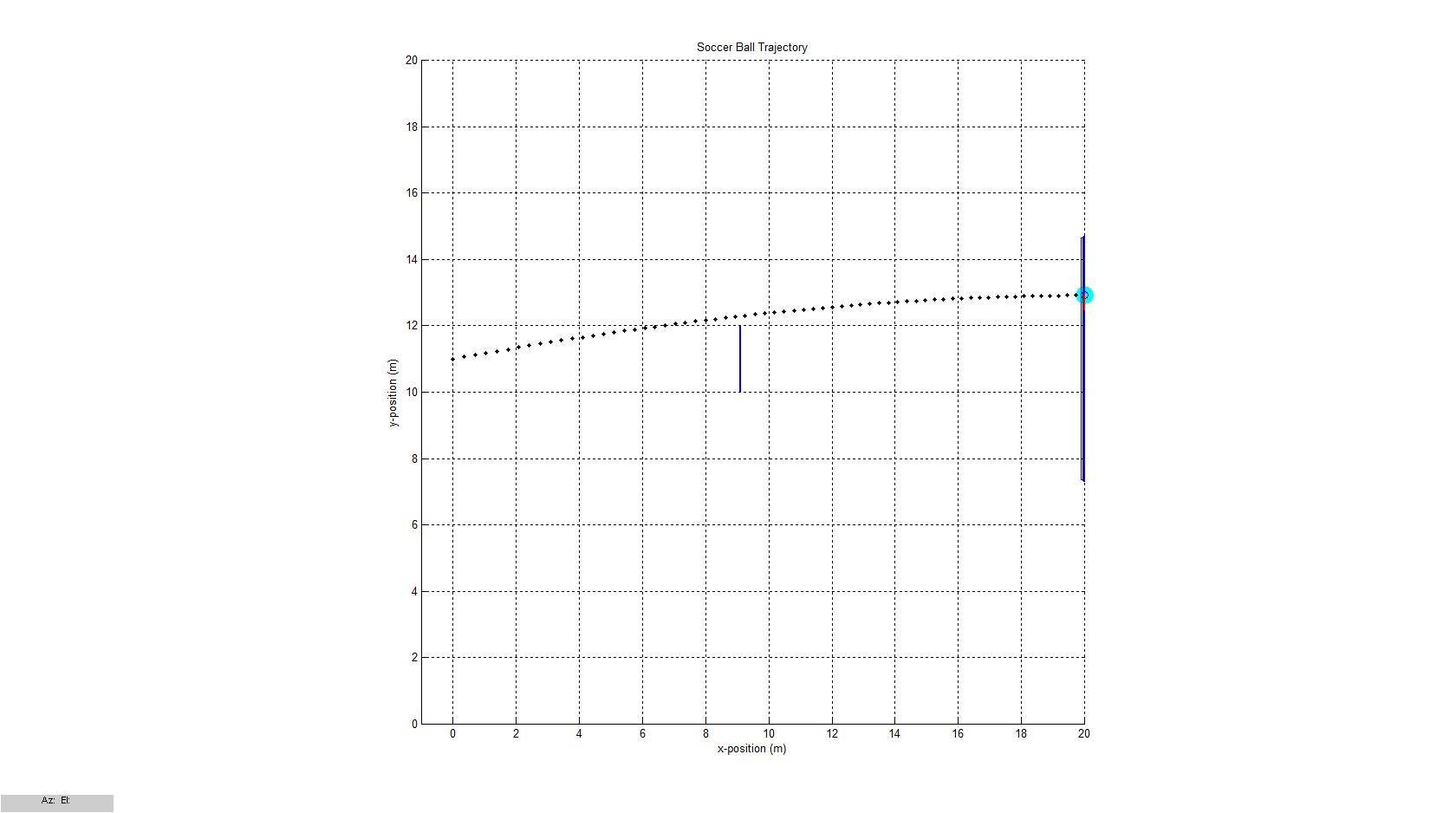
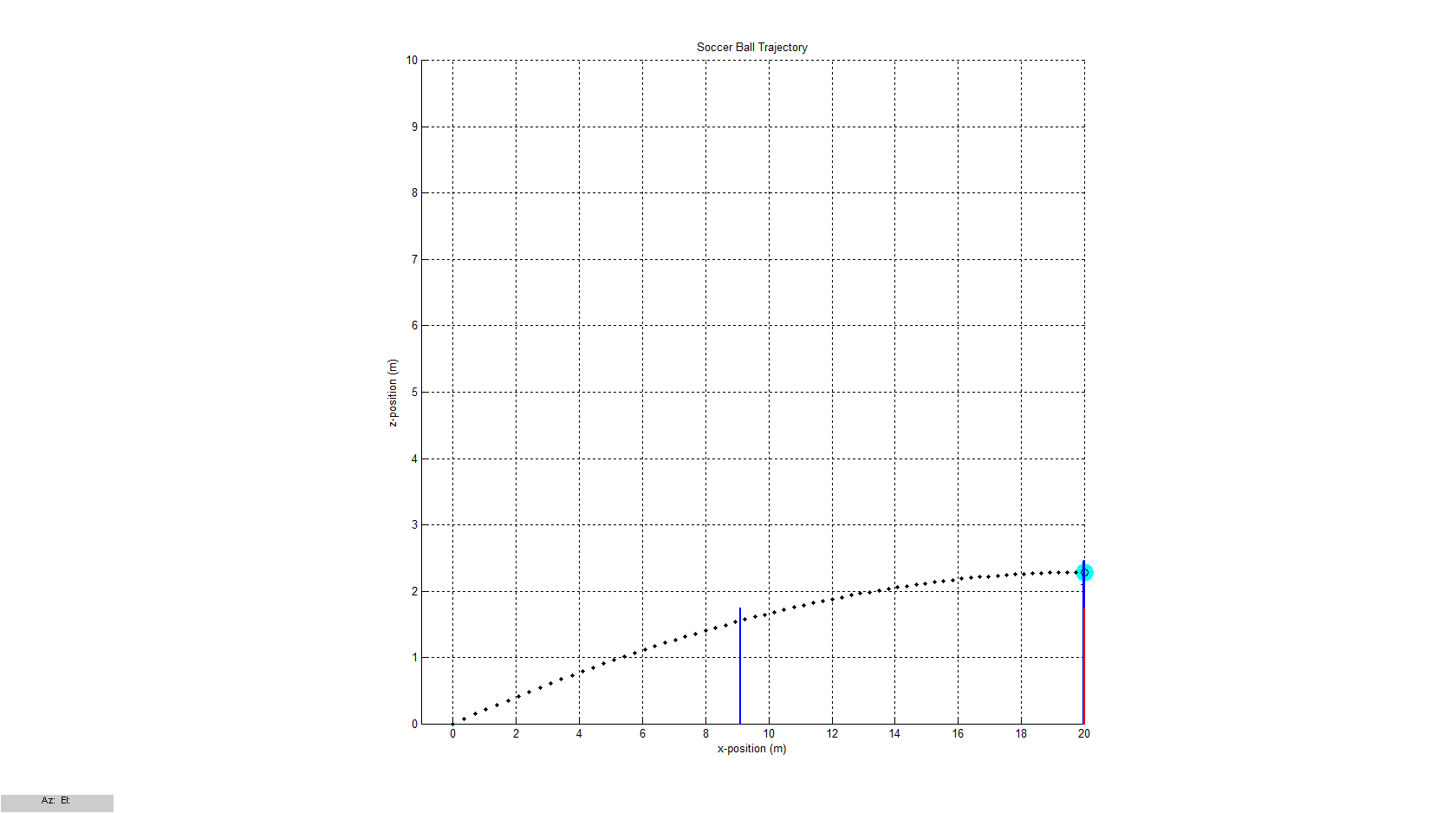
*3-D view*

*“GOAL! The final velocity is 33.7378. The final position at time 1.05s is [30.045,9.2732,2.3362].”*

Distance from the net (m): 30; Rise angle (deg): 13.5; Initial velocity (m/s): 36; Swerve (rotations/s): 20; Shot angle: 0;

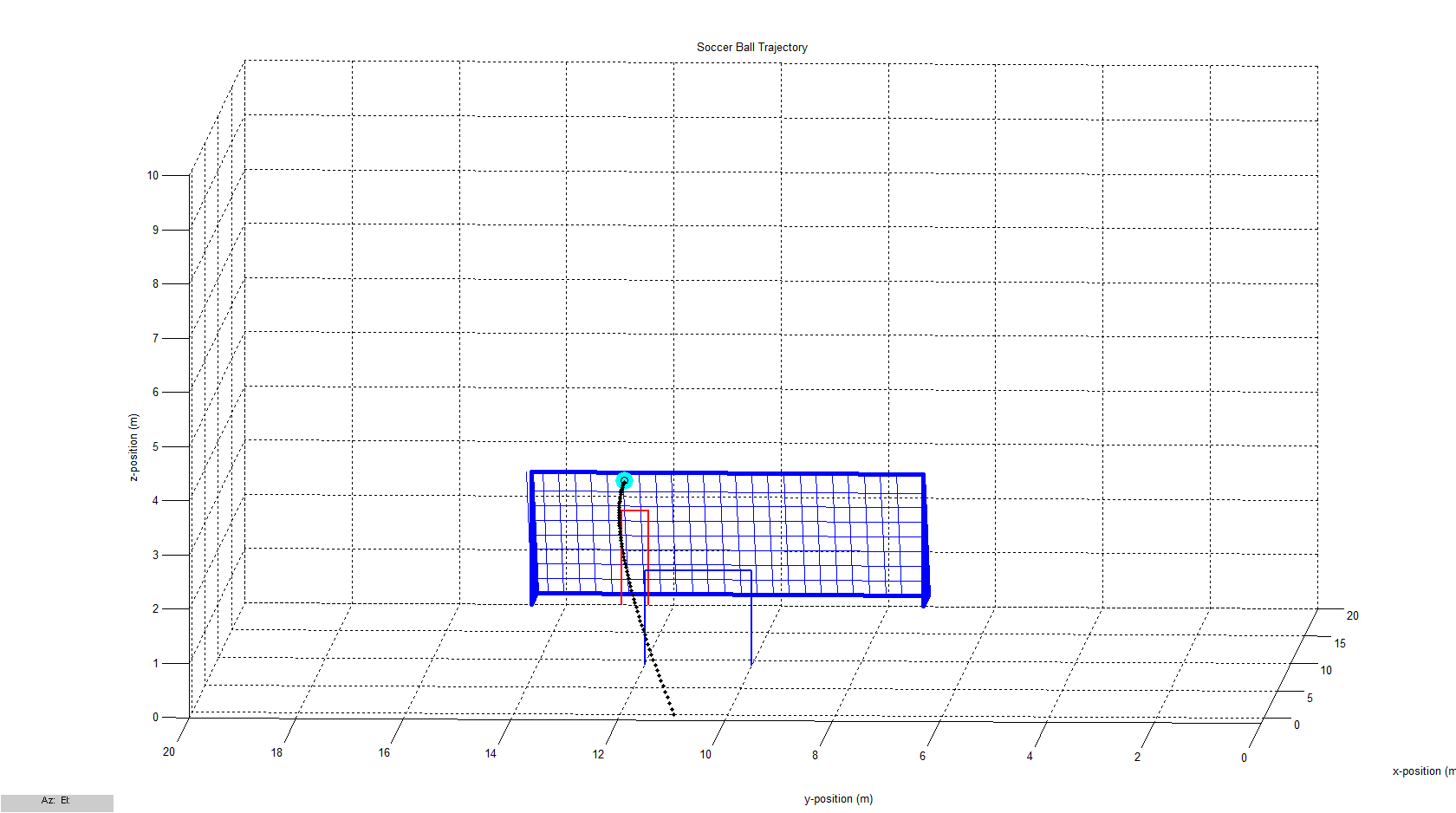
1. **Curve around the wall from 20 meters**

[Figure 9] [Figure 10]

****

*x-z plane view x-y plane view*

[Figure 11]

****

*3-D view*

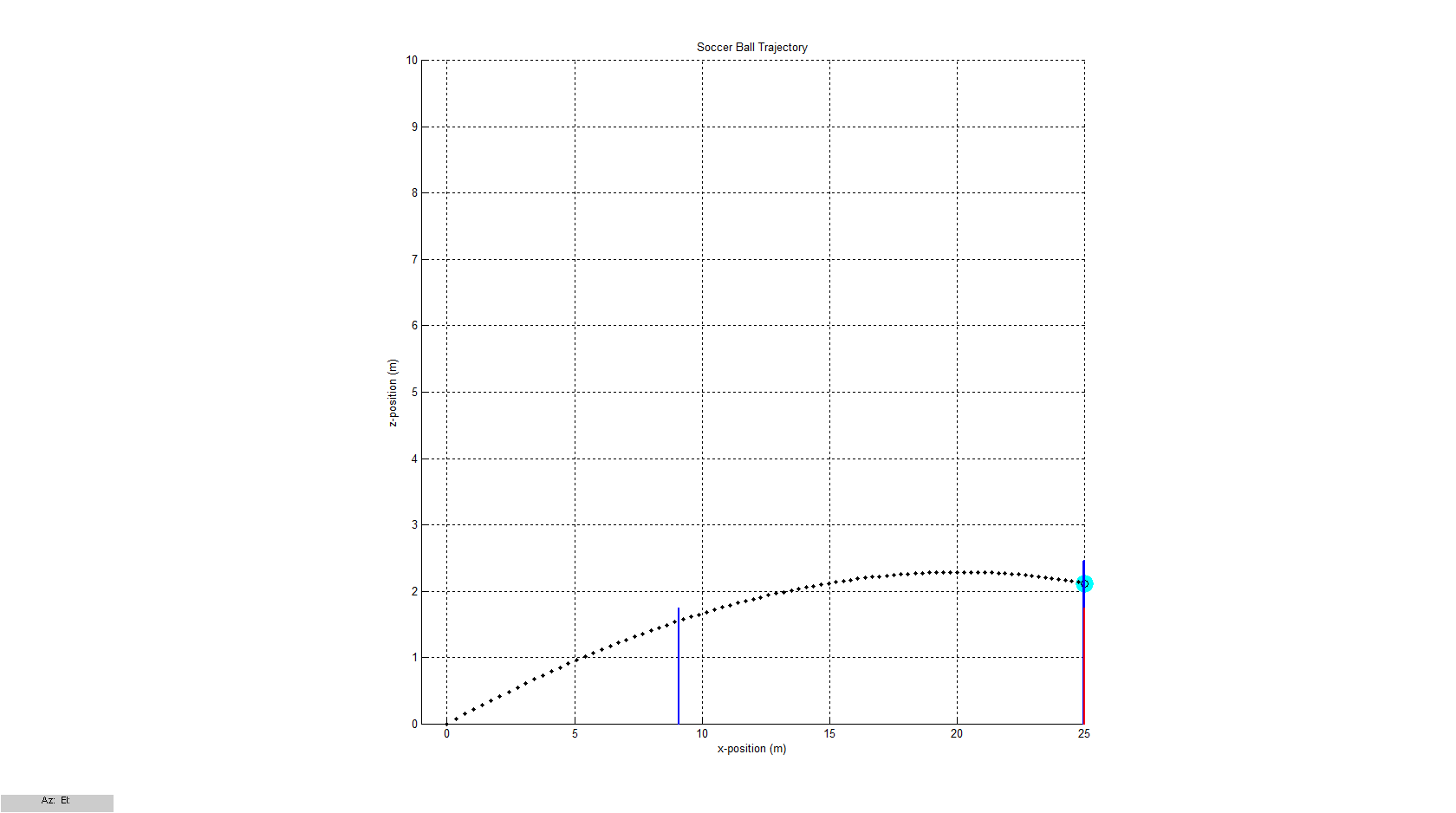
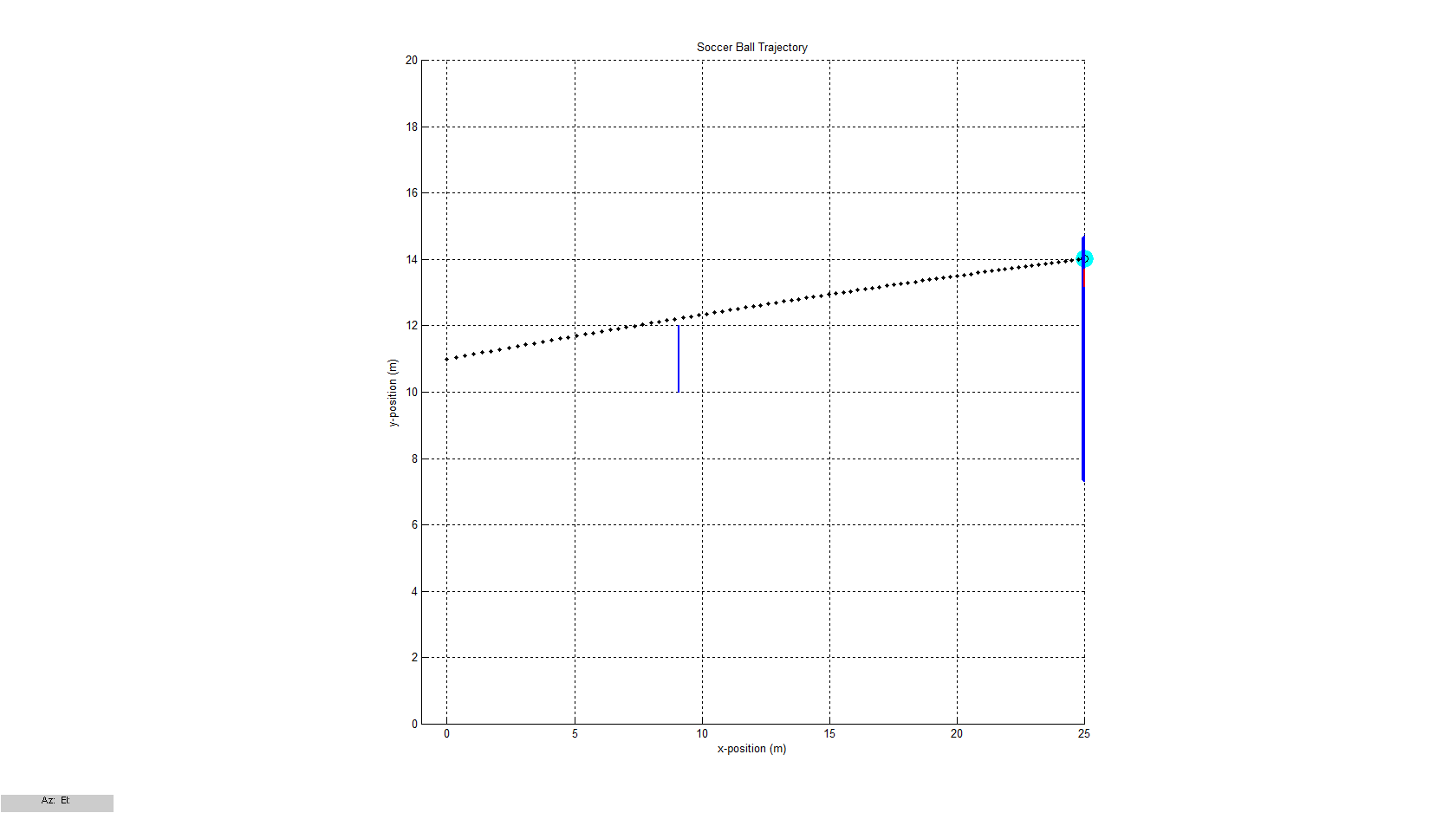
*“GOAL!*

*The final velocity is 38.5708. The final position at time 0.65s is [20.0092,12.9182,2.2856].”*

Distance from the net (m): 20; Rise angle (deg): 12; Initial velocity (m/s): 36; Swerve (rotations/s): 60; Shot angle: 10;

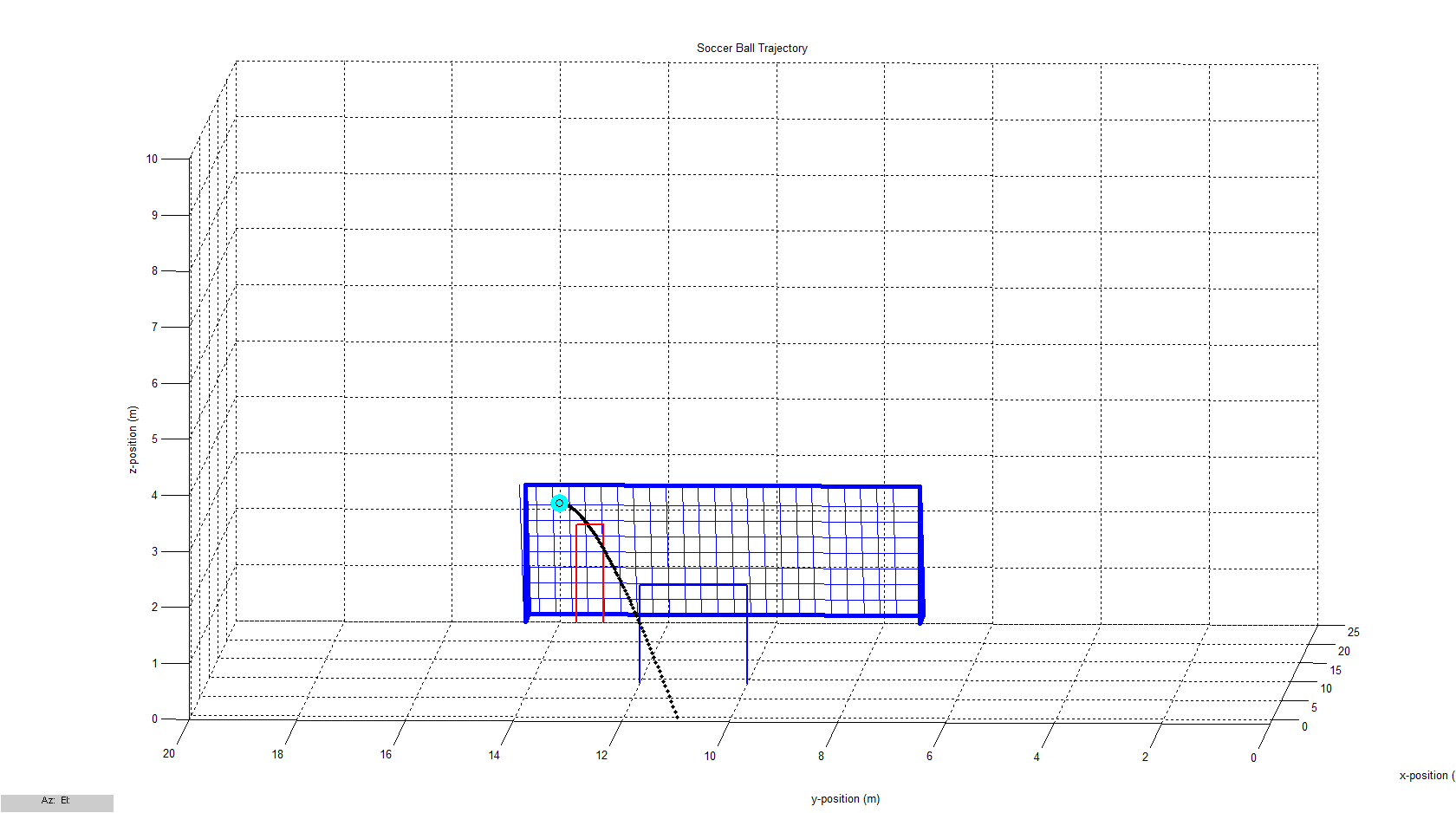
1. **Top corner shot from 25 meters**

[Figure 12] [Figure 13]

****

*x-z plane view x-y plane view*

[Figure 14]

****

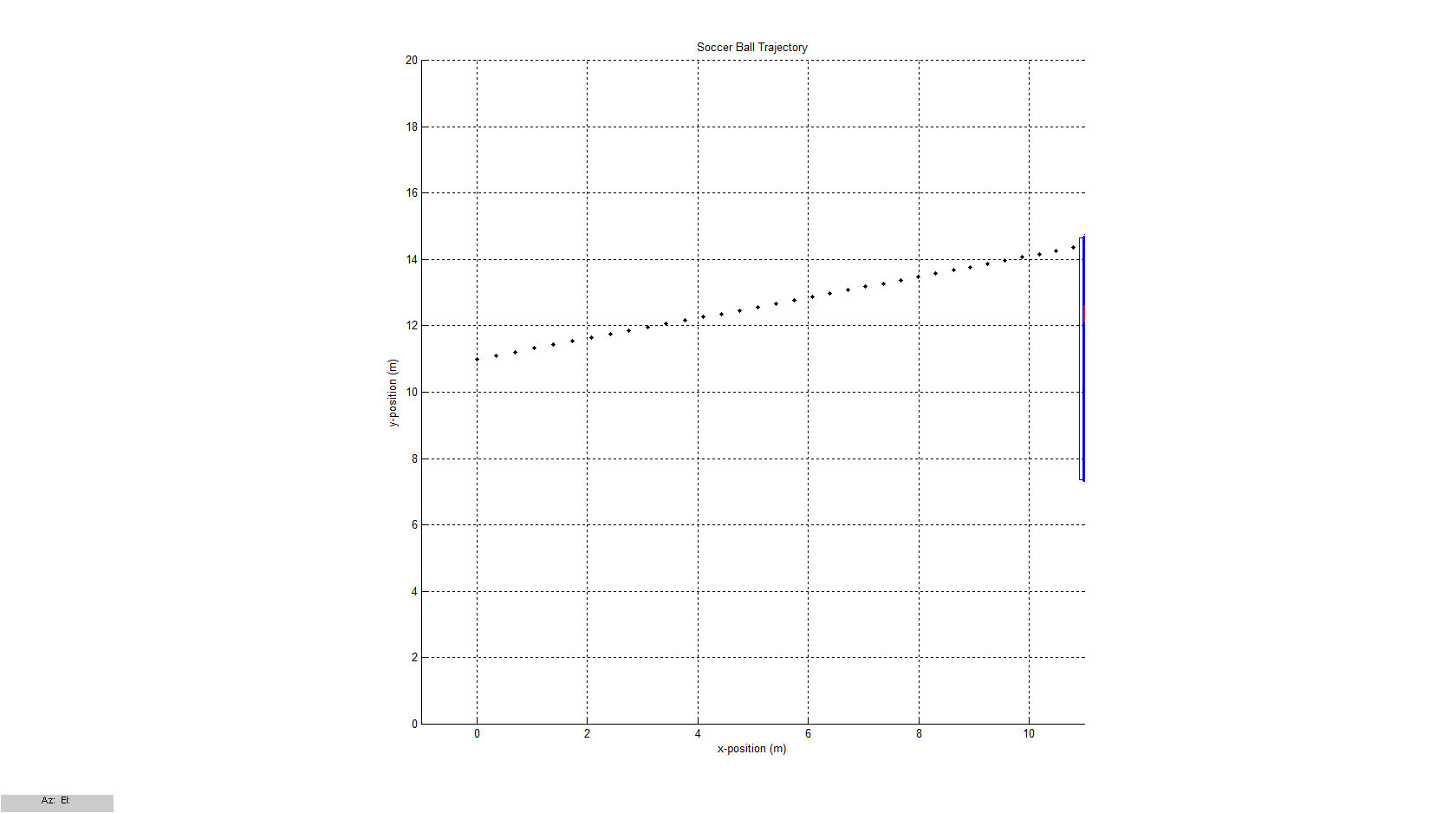
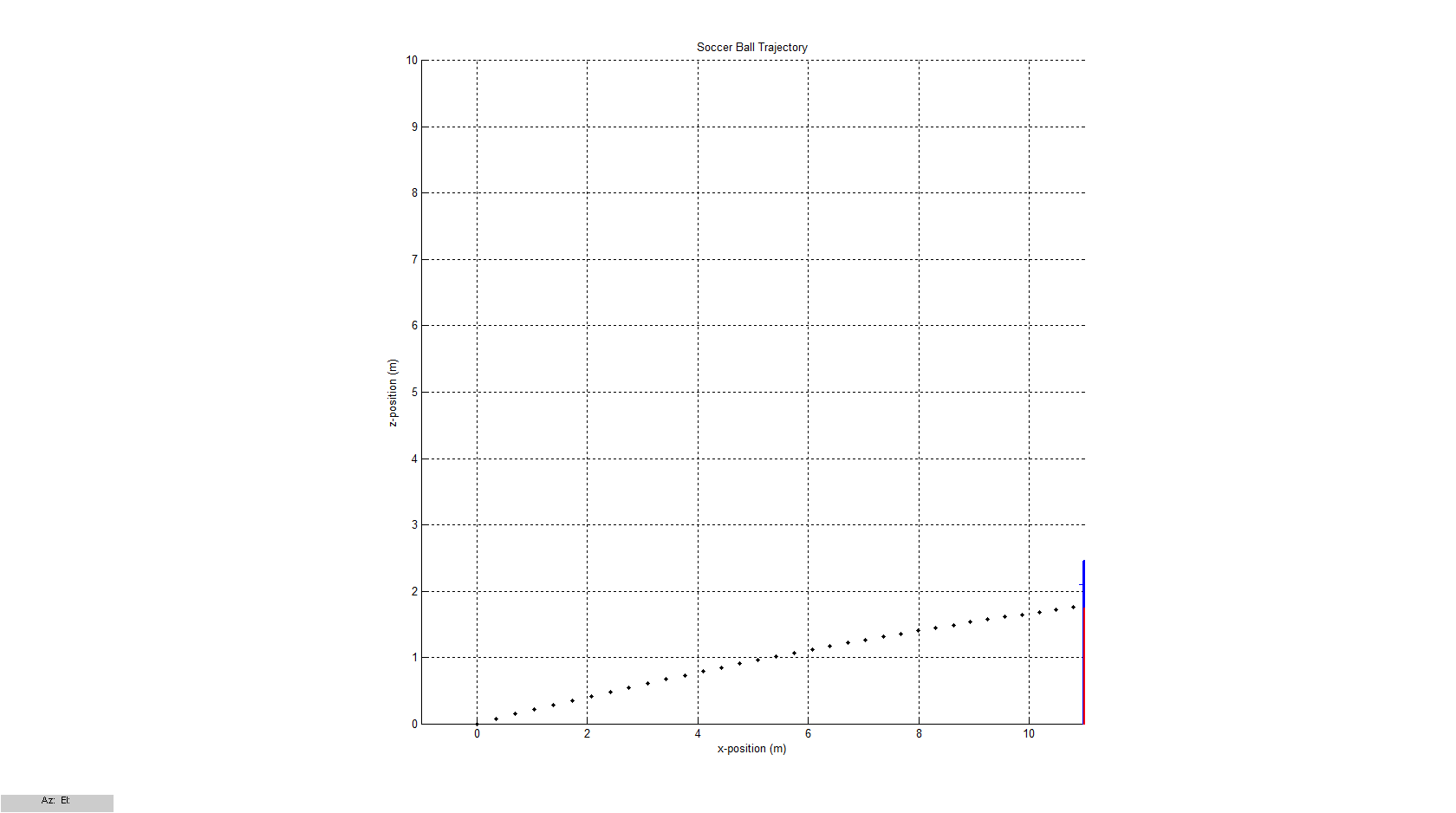
*3-D view*

*“GOAL! The final velocity is 36.1872. The final position at time 0.84s is [25.0108,14.0131,2.1174].”*

Distance from the net (m): 25; Rise angle (deg): 12.5; Initial velocity (m/s): 36; Swerve (rotations/s): 20; Shot angle: 8;

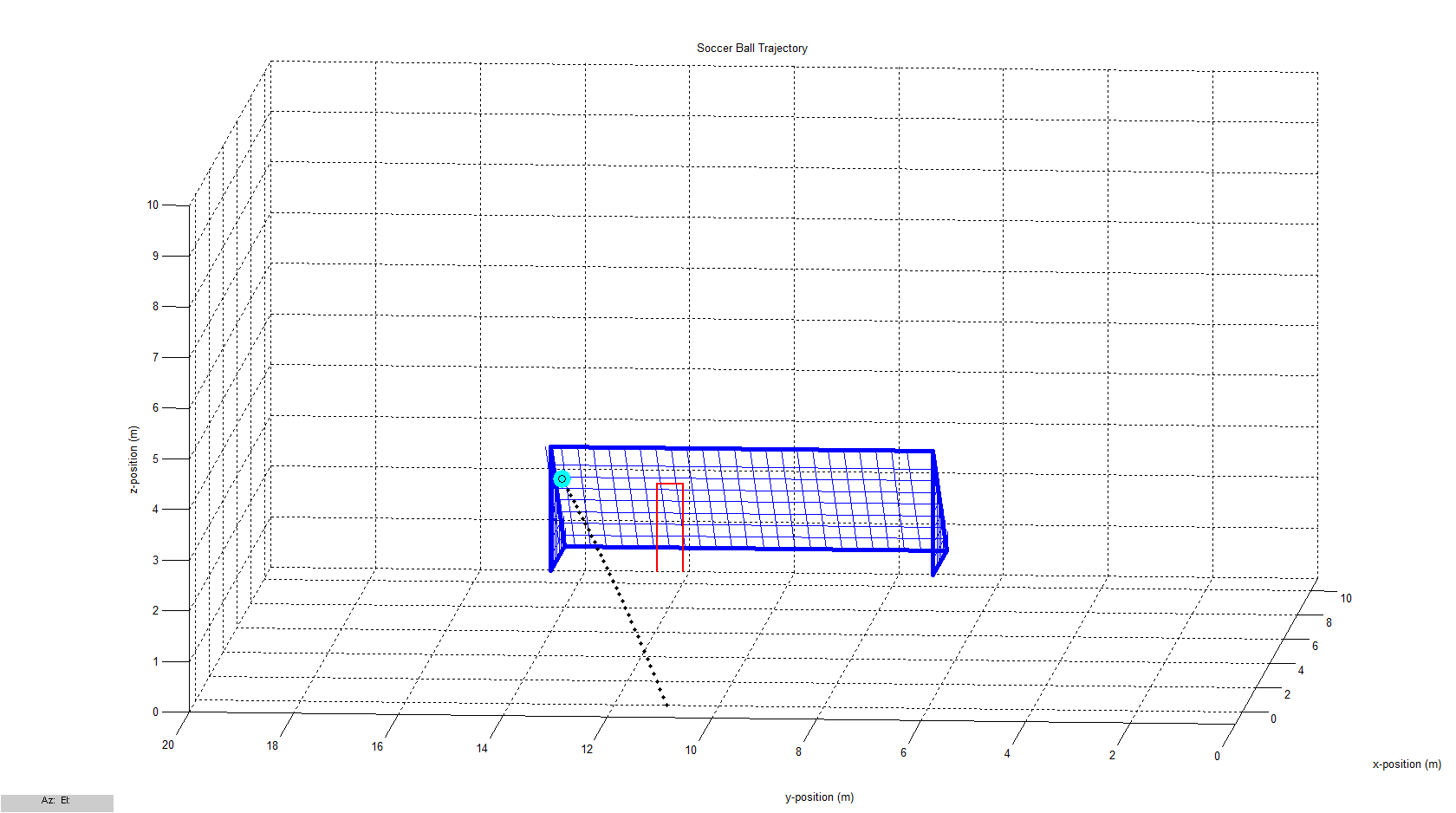
1. ***Penalty Kick (11 meters)***

[Figure 15] [Figure 16]

*** ***

*x-z plane view x-y plane view*

[Figure 17]

******

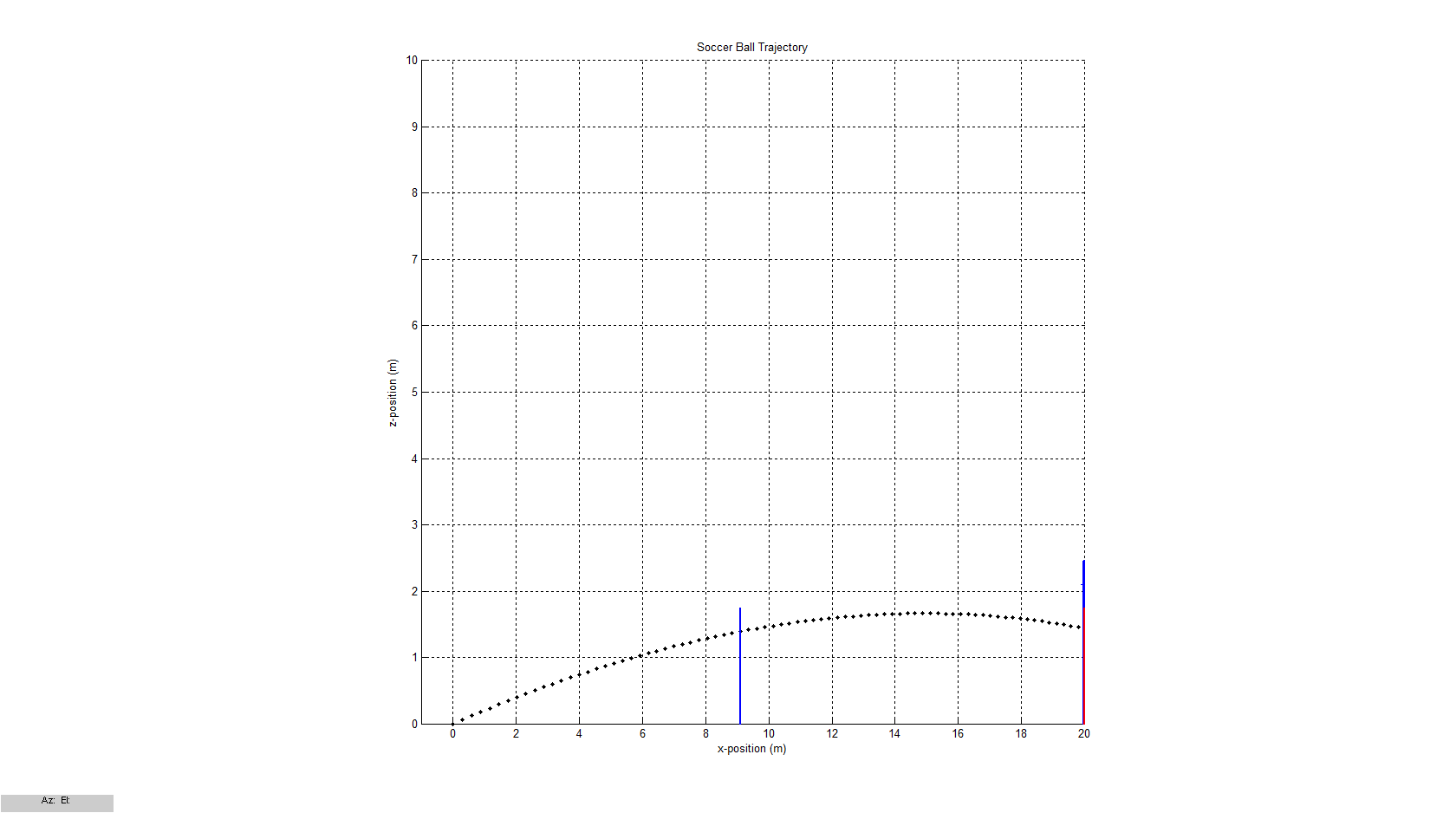
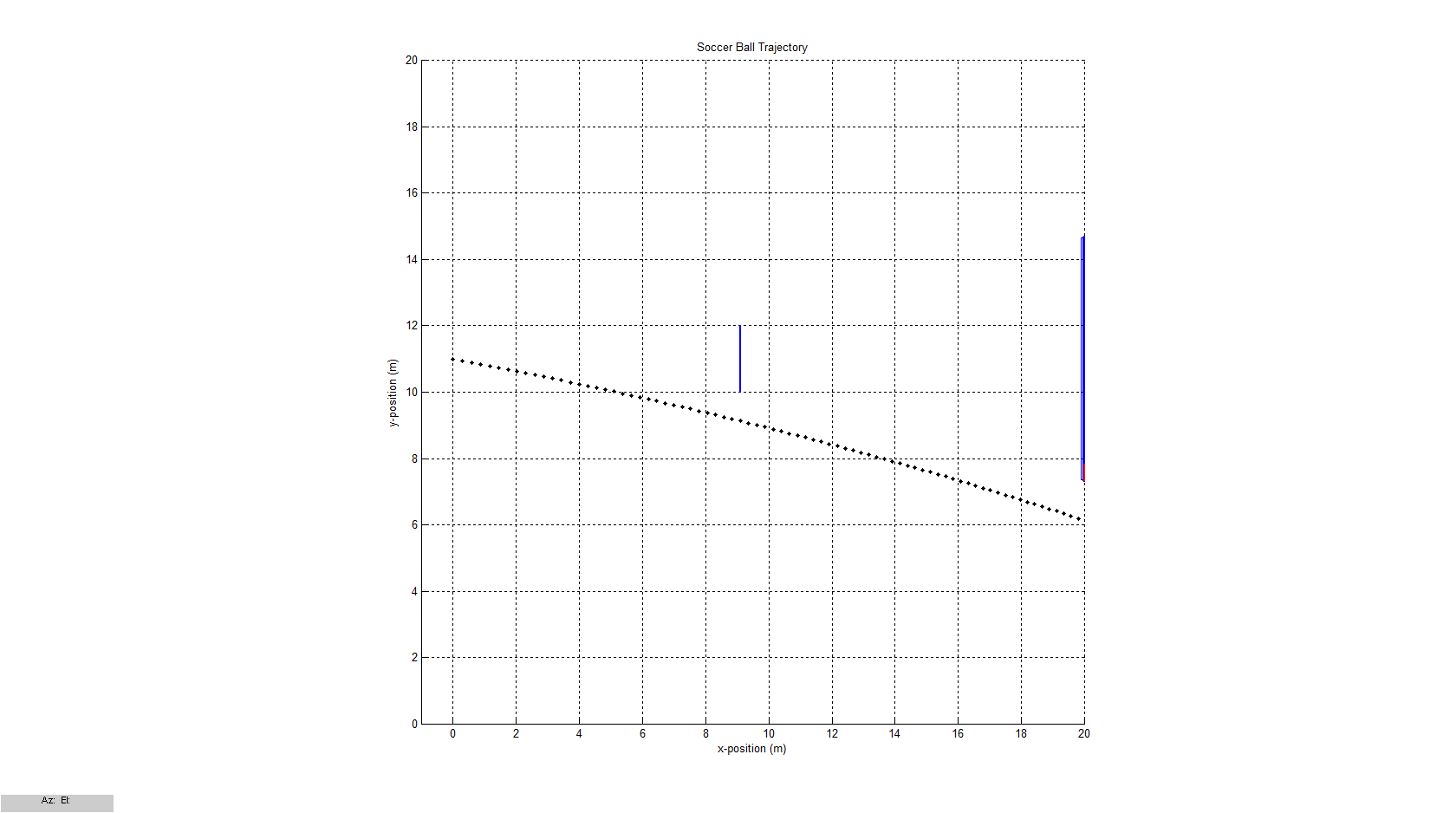
*3-D view*

*“GOAL! The final velocity is 43.2168. The final position at time 0.34s is [11.0973,14.4538,1.7853].”*

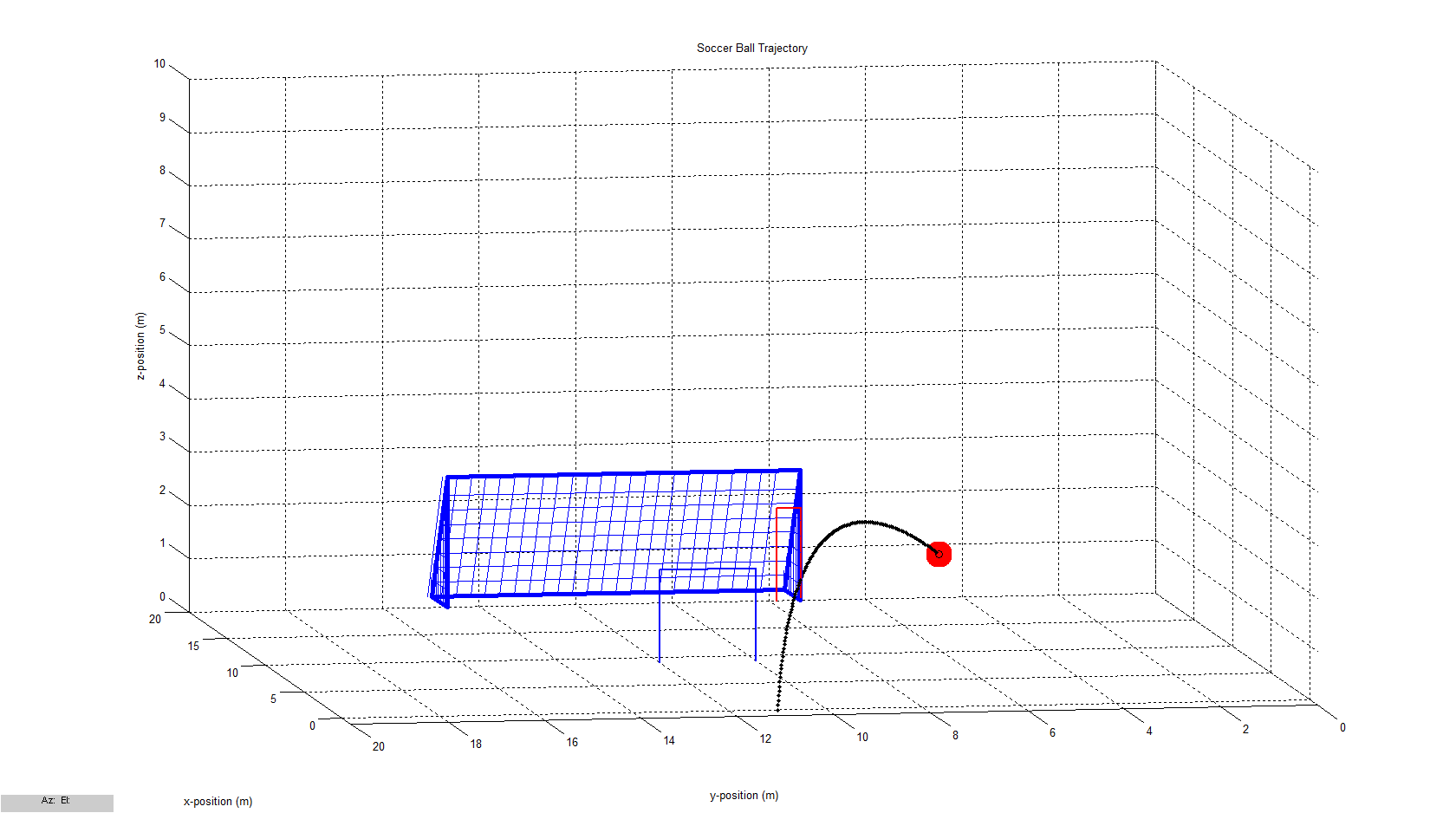
Distance from the net (m): 11; Rise angle (deg): 12; Initial velocity (m/s): 36; Swerve (rotations/s): 20; Shot angle: 17;

1. ***Missing the Net (20 meters)***

[Figure 18] [Figure 19]

**** *x-z plane view x-y plane view*

[Figure 20]

****

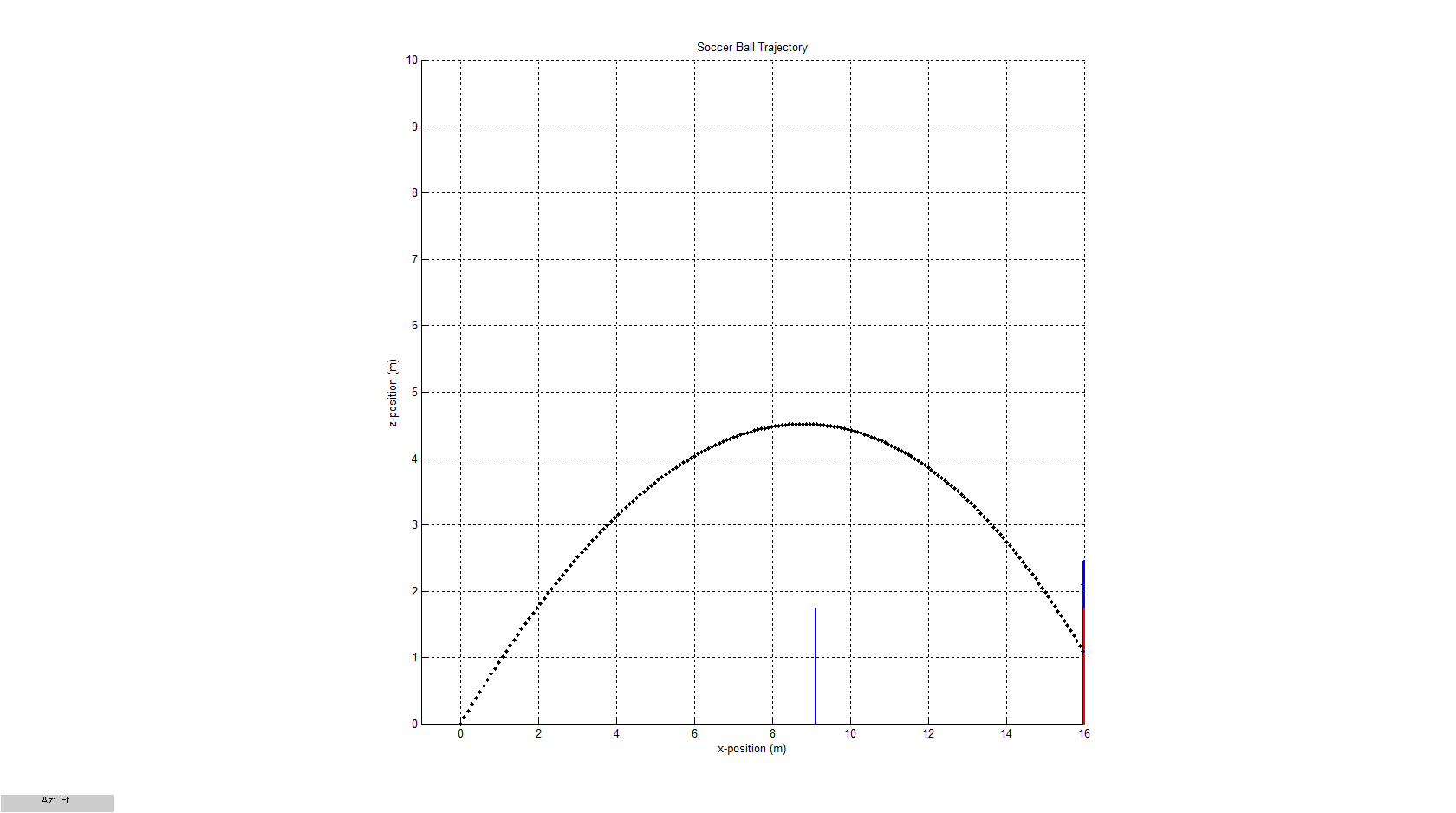
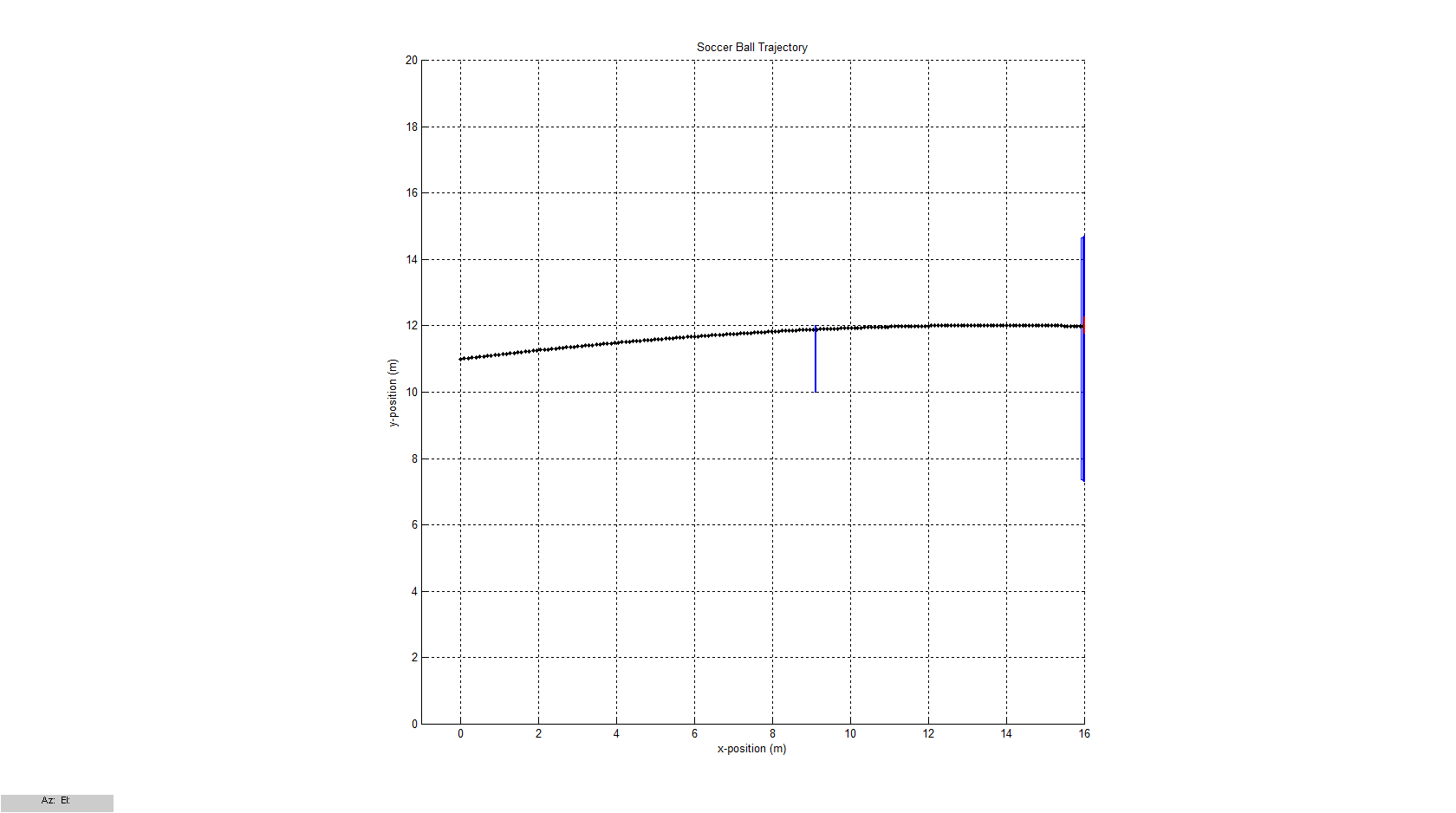
*3-D view*

*“The ball has gone over the net and hit the ground. The final velocity is 28.8932. The final position at time 1.17s is [28.4017,3.1381,-0.045818].”*

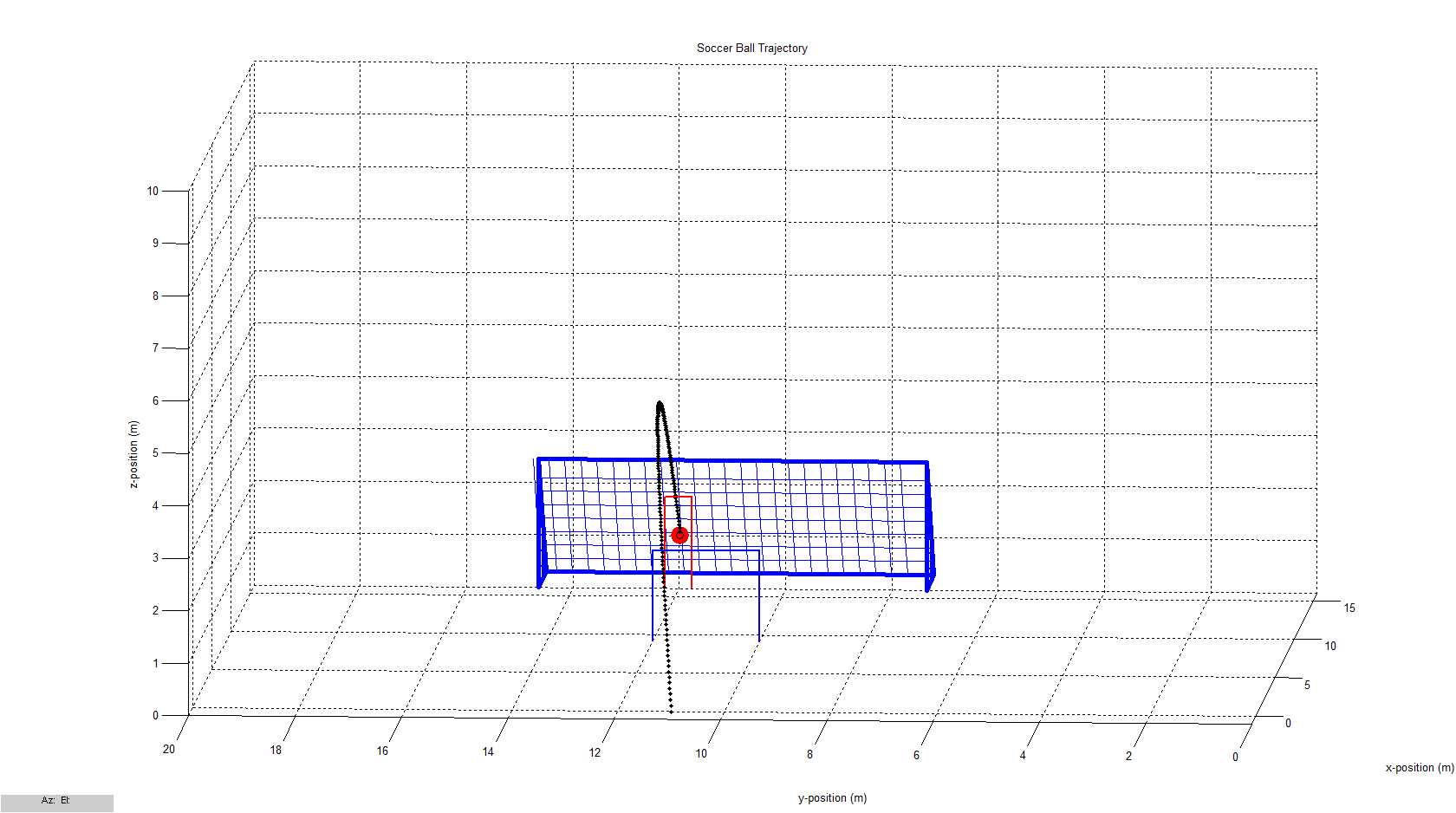
Distance from the net (m): 20; Rise angle (deg): 13.5; Initial velocity (m/s): 36; Swerve (rotations/s): 20; Shot angle: -13;

1. **The Goalkeeper makes a save (16m)**

[Figure 21] [Figure 22]

*x-z plane view x-y plane view*

[Figure 23]



*3-D view*

*“The ball has been caught by the goalkeeper. The final velocity is 11.4193. The final position at time 1.8s is [16.047,11.9844,1.0109].”*

Distance from the net (m): 16; Rise angle (deg): 25; Initial velocity (m/s): 30; Swerve (rotations/s): 20; Shot angle: 0;

**References**

1. http://en.wikipedia.org/wiki/Football\_%28ball%29
2. http://www.guinnessworldrecords.com/records-11000/fastest-football-kick/
3. http://www.grc.nasa.gov/WWW/k-12/airplane/socdrag.html
4. http://en.wikipedia.org/wiki/Density\_of\_air
5. http://footballs.fifa.com/Football-Tests/Weight
6. http://www.livestrong.com/article/99687-size-regulation-soccer-net/
7. Nicholas J. Giordano and Hisao Nakanishi (2006). *Computational Physics*. 2nd ed. New Jersey: Pearson Education. p18-46.