Matlab Script for 3D Visualization of Missile and Air Target Trajectories

Witold Bużantowicz
Faculty of Mechatronics and Aerospace
Military University of Technology
Warsaw, Poland
Email: witold.buzantowicz [AT] wat.edu.pl

Abstract—**This paper presents a script for the 3-dimentional visualization of missile and air target trajectories. This minimalistic but easy-to-use package can help to produce attractive presentations for various scientific or public outreach purposes. The software requires only the basic module of MathWorks Matlab and an ordinary PC or notebook to use. It is freely available for scientific and educational use and can be downloaded from *http://www.wbint.pl/flypath3d/*.**

Keywords—visualization; Matlab; missile; air target; trajectory

I. INTRODUCTION

Vision is the most powerful communication channel that humans possess. Thus, in many fields of academic research, computer graphics and 3D visualization play an increasingly large role in the presentation of research findings. Such 3D representations of simulation results are both illustrative and informative. The aim of this article is to present a free Matlab package for 3D visualizations of missile and air target trajectories that provide a visual reference for the computer simulation. Because of its versatility and ease of use, the software can help to produce attractive presentations for various scientific or public outreach purposes.

The package is freely available for scientific and educational use at: *http://www.wbint.pl/flypath3d/*. It requires only the basic module of MathWorks Matlab and an ordinary PC or notebook to use. All the scripts were tested with Matlab R2014a under 32/64-bit Microsoft Windows 7 and 8.1 operating systems.

II. PROGRAM DESCRIPTION AND DATA SET ORGANIZATION

The package is composed of several M-code files that are installed in the Matlab environment using the *package_setup* command. It provides functions for constructing complex 3D scenes by placing consecutive objects that act, in effect, like building blocks within the layout. Objects can be created using the *new_object* function; the package allows kinematic data, 3D model geometry definitions, and display parameters to be combined into a single easy-to-use data set:

```
new_object(filename,matrix,varargin);
```

The function parameters are: output set file name (*filename*), array of kinematic data (*matrix*) and some optional arguments (*varargin*). Descriptions of the *new_object* function and other package functions and their parameters can be displayed using the standard Matlab *help* command; they will not be presented here for the sake of brevity. The array of kinematic data **K** contains three Cartesian coordinate vectors **x**, **y**, **z** and three angle vectors **θ**, **ψ**, **φ** completely describing the 3D object position in time:

$$\mathbf{K}\in\mathbf{R}^{j\times 6}:\quad \mathbf{K}=\begin{bmatrix}\mathbf{x} & \mathbf{y} & \mathbf{z} & \boldsymbol{\theta} & \boldsymbol{\psi} & \boldsymbol{\varphi}\end{bmatrix} = \begin{bmatrix} x_1 & y_1 & z_1 & \theta_1 & \psi_1 & \varphi_1 \\ x_2 & y_2 & z_2 & \theta_2 & \psi_2 & \varphi_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_j & y_j & z_j & \theta_j & \psi_j & \varphi_j \end{bmatrix}, \tag{1}$$

where x_i, y_i, z_i are the object gravity centre Cartesian coordinates and θ_i, ψ_i, φ_i are the pitch, yaw and roll angles in successive instants of time, $i\in\{1, 2, \ldots, j\}$.

The transformation matrix **M** is taken as follows:

$$\mathbf{M}\in\mathbf{R}^{3\times 3}:\quad \mathbf{M}=\begin{bmatrix} C\theta C\psi & -S\theta C\psi & -S\psi \\ \begin{pmatrix} C\theta S\psi S\varphi \\ +S\theta C\varphi \end{pmatrix} & \begin{pmatrix} -S\theta S\psi S\varphi \\ +C\theta C\varphi \end{pmatrix} & -C\psi S\varphi \\ \begin{pmatrix} -C\theta S\psi C\varphi \\ +S\theta S\varphi \end{pmatrix} & \begin{pmatrix} S\theta S\psi C\varphi \\ +C\theta S\varphi \end{pmatrix} & C\psi C\varphi \end{bmatrix}, \tag{2}$$

where θ, ψ and φ are the current values of pitch, yaw and roll angle, respectively, C stands for cosine, and S stands for sine function.

The relationship of these angles to the orientation of the visualized object in space is illustrated in Figure 1. A positive value of the pitch angle shows the object's climb, a positive value of the yaw angle shows the object's deviation to the right, and a positive value of the roll angle shows the tilting of

the object to the right relative to its longitudinal axis. All of the position and angle values refer to the global inertial clockwise coordinate system with a north-oriented *y* axis. This means that an object with initial coordinates $x_0 = 0$, $y_0 = 0$, $z_0 = 0$, angle values $\theta = 0$, $\psi = 0$, $\varphi = 0$, and constant speed $v = \text{const}$ will move along the *y* axis in the positive direction.

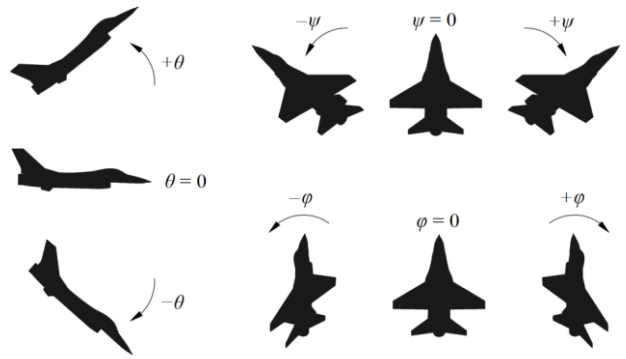

Figure 1. Interpretation of pitch, yaw and roll angle signs

The package supports 3D models collected in vertex and face list file formats (Figure 2). Additional models can be obtained from OBJ files created by popular 3D graphics and animation software, such as Blender or 3ds Max [1, 4, 5]. Please note that the package supports only triangular mesh objects; before prepared models can be used, appropriate conversions must be made. 3D models can be imported and tested using *model_import* and *model_show* functions.

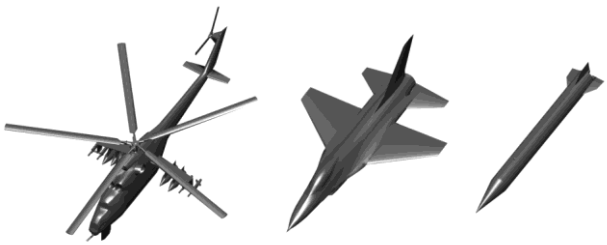

Figure 2. Examples of solid model collection included with the package

III. Numerical Example

In order to present the basic capabilities of the software, a numerical example will be considered. Assume that a simplified ballistic target trajectory is described by the following equations:

$$x = 0, \quad y = v_0 \cos\alpha, \quad z = v_0 \sin\alpha - \frac{gt^2}{2},$$
$$\theta = \tan^{-1}\left(\frac{v_0 \sin\alpha - gt}{v_0 \cos\alpha}\right), \quad \psi = 0, \quad \varphi = 0, \tag{3}$$

where $v_0 = 100$ m/s is the initial velocity, $\alpha = 45^o$ is the angle at which the target is launched, and $g = 9.81$ m/s2 is the gravitational acceleration. The numerical solution of (3), e.g. for 10 seconds of the flight, can be found using following Matlab code:

```
t = 0:0.01:10;
K = zeros(length(t),6);
K(:,2) = 100*t*cos(pi/4);
K(:,3) = 100*t*sin(pi/4)-9.81*t.*t/2;
K(:,4) = atan((100*sin(pi/4)-9.81*t)...
         /(100*cos(pi/4)));
```

The matrix **K** contains all the data required by the visualization script and is used as the input array for *new_object* function to prepare an air target simulation model:

```
new_object('target.mat',K,...
        'model','missile.mat',...
        'edge',[.2 .2 .2],...
        'face',[.3 .3 .3],...
        'scale',10.0,...
        'path','on',...
        'pathcolor',[.5 .5 .5],...
        'pathwidth',1);
```

Additional parameters such as *edge*, *face*, *scale*, *path* etc. are used to specify object properties in detail; these are clearly described in the user manual. The *model* parameter is used to determine the 3D solid model file for visualization purposes.

Finally, the *flypath* function is executed to generate the scene and save the result in PNG file format:

```
flypath('target.mat','step',200,...
        'view',[90 0],...
        'window',[1200 800],...
        'output','picture.png',...
        'dpi',600,...
        'xlim',[-100 100],...
        'ylim',[0 800],...
        'zlim',[0 300]);
```

The result of this simple simulation is presented in Figure 3.

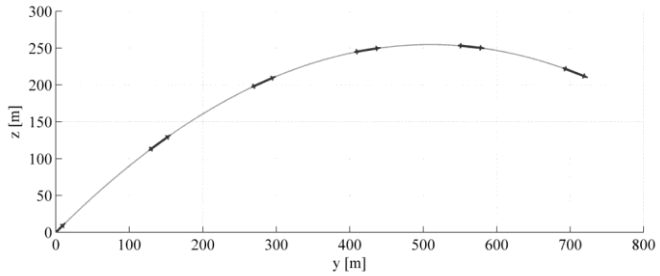

Figure 3. Flight trajectory of the ballistic target

IV. Examples of Application

The script described here can be applied in many fields. Several examples of possible package applications are presented in Figures 4-8. The software can generate both static images and animations for several purposes, including among others: missile and/or air target flight trajectories (Figure 4), missile-target engagement visualizations (Figure 5) and tactical scenes generation support (Figures 6-8). Static images

can be saved as PNG files with user-defined resolution. Animations can be made by including optional arguments for *flypath* function. Animations are saved to the current Matlab folder as animated GIF images.

It should be emphasized that extensive configuration options and the flexibility of this package enable the visualization software to be applied to any object whose position in time can be described *via* matrix **K**. The use of standard Matlab graphics commands [2, 3] is also possible.

Detailed instructions for use of the software are provided in a step-by-step manual included in the package. The website, which shows all the codes and data sets used for demonstration, supplements the information in this paper. Supplementary data associated with this article can be found at: *http://www.wbint.pl/flypath3d/*.

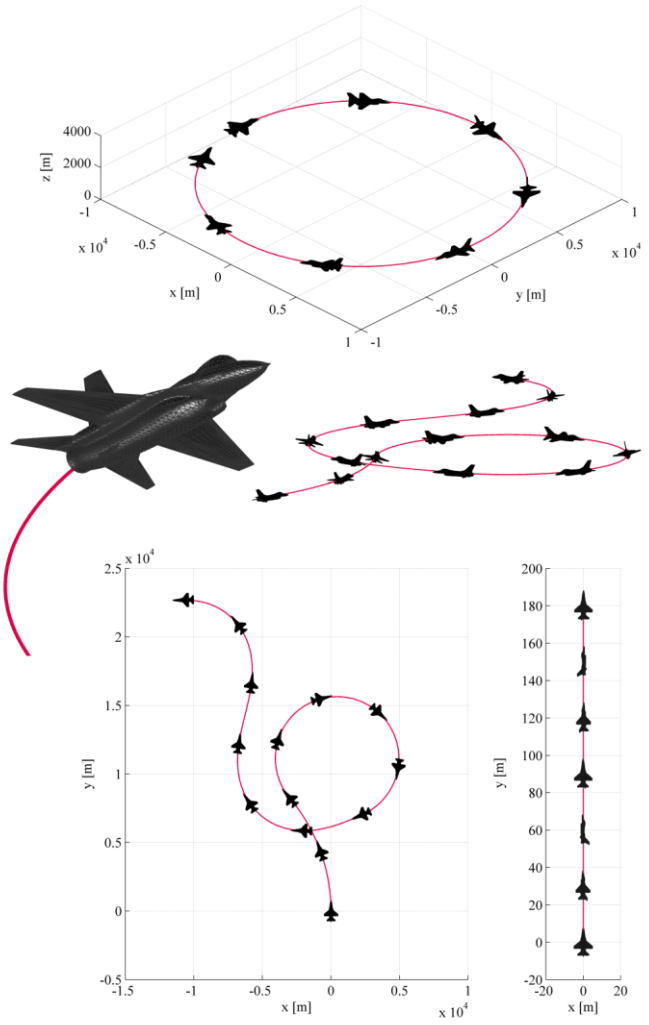

Figure 4. Trajectory visualization examples

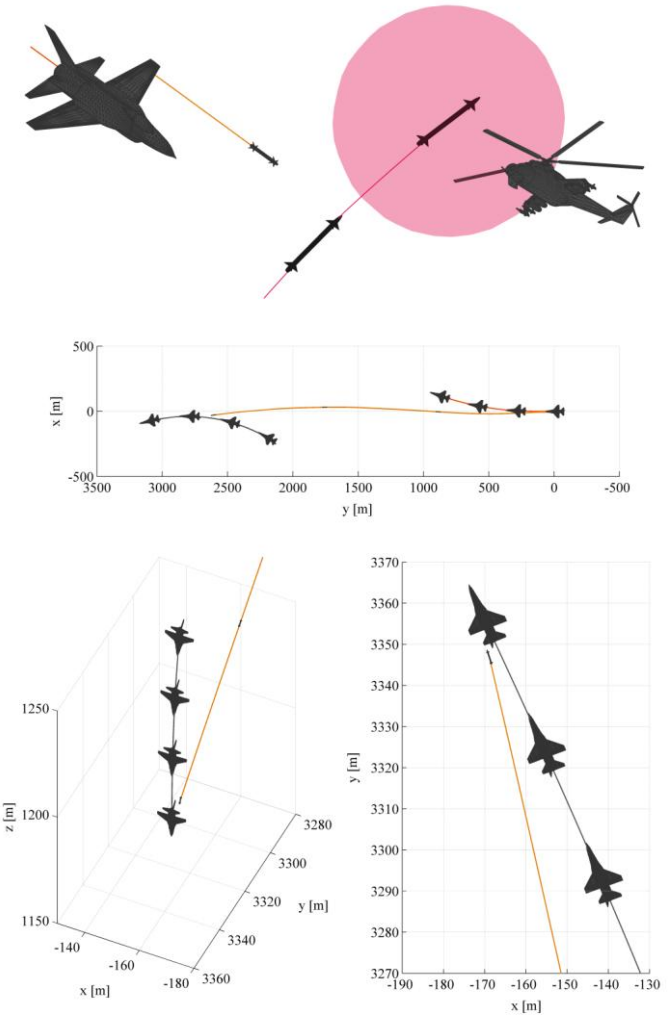

Figure 5. Visualizations of various missile-target engagements

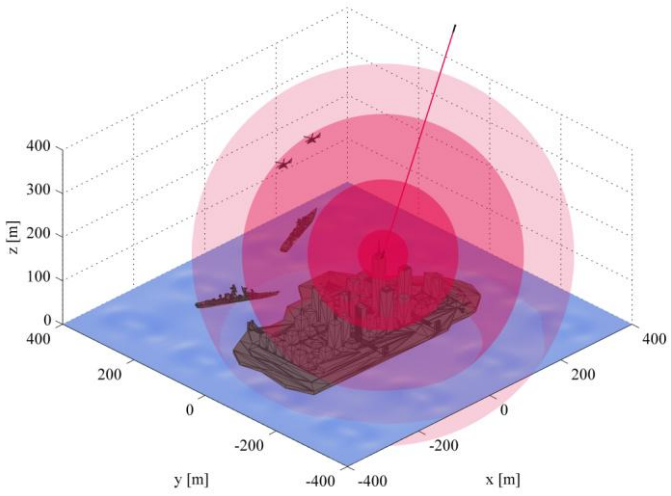

Figure 6. Visualization of fireball radius of nuclear airburst

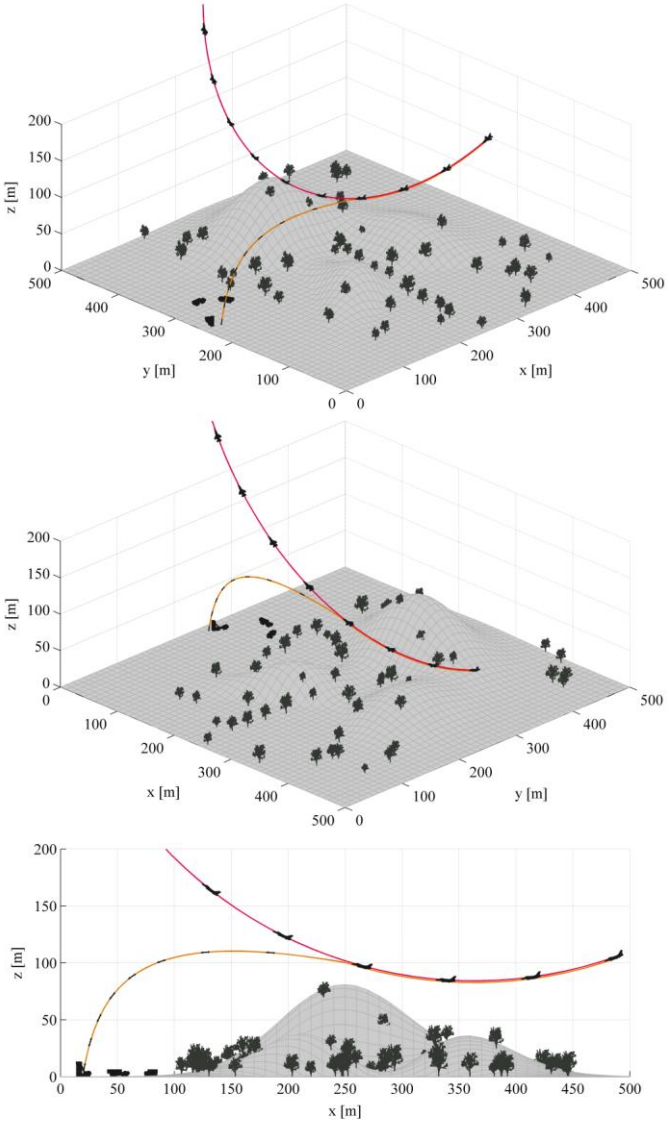

Figure 7. Visualization of tactical scene

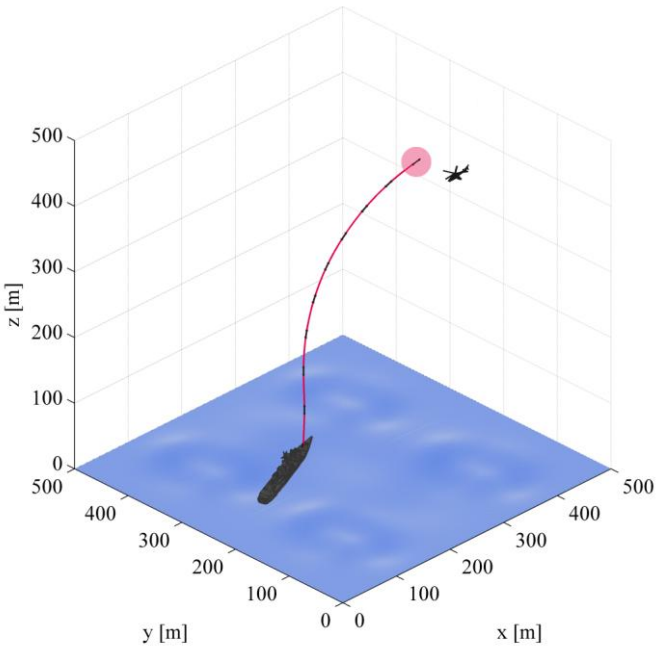

Figure 8. Visualization of naval tactical situation

V. SUMMARY REMARKS

This paper has presented a script for 3D visualizations of missile and air target trajectories. The package includes several easy-to-use functions that can help in the preparation of static or animated visualizations of scientific results for various scientific and public presentation purposes. The package is noteworthy for its additional functionality, including its ability to cooperate with 3D models and use standard Matlab commands.

The author hopes that this tool will help present visualizations of the results of research in an efficient and attractive way.

REFERENCES

[1] C. Kuhn, Blender 3D Incredible Machines, Birmingham: Packt Publishing, 2016.

[2] N. Majumdar, S. Banerjee S., Matlab Graphics and Data Visualization Cookbook, Birmingham: Packt Publishing, 2012.

[3] D. Redfern, C. Campbell, The Matlab® 5 Handbook, New York: Springer, 2012.

[4] S. Tickoo, Autodesk 3ds Max 2015: A Comprehensive Guide, Schererville: Cadcim Technologies, 2014.

[5] O. Villar, Learning Blender, Boston: Addison-Wesley Publishing, 2014.