An Abstract of

**Three－Dimensional Bio-Graphic Website for Biophysicists**

by

Yadi Wen

Submitted to the Graduate Faculty as partial fulfillment of the requirement for the

Master Degree in Computer Science

The University of Toledo

August 2016

This 3D graphic website is designed for biophysicists. This 3D graphic website custom JavaScript has been implemented for building the interface, importing the geometry, moving rotation protein molecules as well as zooming in and out protein molecules. In my opinion, the website I intend to design has to be easy to use (user-friendly) as the target user (biophysicists) may not be a computer scientist. Thus, this website must be simple and remain its essential elements, and it can be applicable for both user interface and for user experience. With this 3D graphic website, biophysicist could easily process protein and other macromolecules in 3D space. It also allows biophysicists to build a reasonable sequence of movement for proteins. For non-professional people, it will be easier for them to understand the protein structure at the nanoscale by visual perception in the real world.

**Content**

Introduction 1

Motivation behind the Project 3

Requirement and Specification 4

Project Design 7

Project Implementation 14

Conclusions 16

**Chapter 1**

# Introduction

The Bio-Graphic website is intended for biophysicists who are creating biological models. Today there are many 3-D graphic websites and software such as Tinkercad, 3DTin and FreeCad. However, major disadvantage of those websites and software is that they have too many functions unnecessary for biophysicists. Therefore, by setting basic functions necessary for biophysicists, I have built a website to help them process proteins in the 3D space.

I have been developing this website for almost one year but there is still a lot of work to do. At first, I discussed with the student from Biological Science Department of University of Toledo. Upon his request, I needed to make their own 3D version and clear user interface for biophysicists, so my website should have the functions of sphere and cylindrical geometries as well as moving function which is intended for moving the geometries (up and down, left and right) in 3D space. Currently this website has had all those basic features and run smoothly. This report is written according to the website approach and takes every detail into consideration based on their requirements.

Firstly, I will present an overview of this project, design motivations and goals to the reader. Next, I will go into details about how it motivated the development of the website. Then, I will present how this website works. After that, I will describe which technology I have decided to use for the development of my project. Lastly, I will serve to cover what may come next in the development of this project. In fact, I hope this project which is still a work in progress (if not discontinued) will be carried on also by other students and researchers who are interested in it. The conclusions hereinafter will explain why this project hasn’t been completed, and also include the work that I have done.

**Chapter 2**

# Motivation behind the Project

Creating a high-quality graphics on desktop computers has always been my dream. In recent years, with the development of the 3-D printing technology, it is promoting major innovations in many areas such as engineering, manufacturing, art, education and medicine. Recent advances have enabled 3D printing of biocompatible materials, cells and supporting components to develop into complex 3D functional living tissues. 3D bio-printing is applied to regenerative medicine to address the need for tissues and organs suitable for transplantation. So bio-printing technology has attracted a large number of biophysicists’ attention. Accordingly, the idea that a 3-D graphic website for biophysicists could be developed is formed.

According to the investigation, there are many 3D graphic websites including Tinkercad, and 3DTin etc. at present. But so far, there is no 3D graphic website designed specifically for biophysicists. For most biophysicists, the 3D graphic websites have too many unnecessary functions and the user interface is too complicated. For my website, it must be easy to use (user-friendly) as the target user may not be a computer scientist. Thus, it must be simple and clear, and it can be applicable both for user interface and user experience.

**Chapter 3**

# Requirement and Specification

In this chapter I will describe the requirements for this website project, and what I would expect to implement in the final version of this website.

Like I mentioned before, the website I want to design must be easy to use (user-friendly) as the target user may not be a computer scientist. Thus, it must be simple and remain essential elements, and it can be applicable both for user interface and user experience. I will try my best to give the users a clear interface.

Now I will introduce the system requirements for the website I want to design. I rely on the subdivision based on functional and non-functional requirement. I will use the MoSCow method and a tabular representation.

3.1 Functional Requirements

The functional requirements must be taken into consideration when designing the website because they highlight functions and performance of the website. During the design, I must think about these core operations. And the following tables have included some of these functional requirements.

Table 1: Functional Requirements

|  |  |  |
| --- | --- | --- |
| **Requirement** | | **MoSCow** |
| Basic sharps | Cube | Should have |
| Rectangle prism | Should have |
| Cylinder | Must have |
| Sphere | Must have |
| Cone | Should have |
| Oval | Should have |
| Moving (up and down, left and right) structures | | Must have |
| Zoom in and zoom out | | Must have |
| Functions of group and ungroup | | Should have |
| Functions of copy and paste | | Should have |

3.2 Non-Functional Requirements

On the other hand, non-functional requirements describe what the website should have when it is in service. These non-functional requirements are not concerned with the functions of the website. Instead, they typically describe the performance of the website.

Table 2: Non-Functional Requirements

|  |  |
| --- | --- |
| Requirement | MoSCow |
| User-friendly UI | Must have |
| Fast loading speed | Must have |
| Security(Sign up, Login) | Should have |

3.3 Use Case

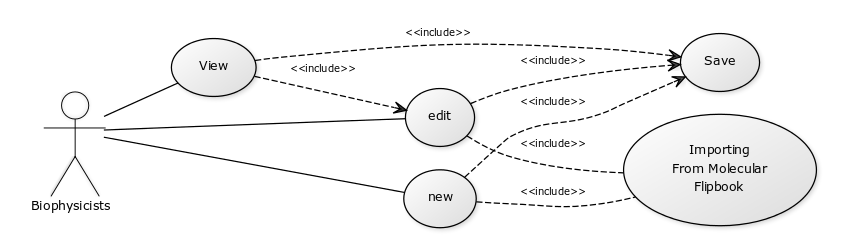
Use case Diagram is a tool to describe what service the website can provide. Those services are provided for users who operate the website. As shown in figure, the main users for my website are biophysicists. When they visit the website, they could view and edit the project. They are also allowed to create new blank project. When they create a new project or edit the project, the importing features of the website will make it possible for biophysicists to import molecular from Molecular Flipbook. Once all work has been done, users could save their files directly. When users run the website, they are allowed to move around the screen at any angle, zoom in and out the screen, as well as rotate the object at any degree. 

Figure 3.1: User Case Diagram

**Chapter 4**

# Project Design

In this chapter I will describe the process of designing various aspects of the website. I use some UML diagrams that I will also describe to show how the software is supposed to evolve in the next design stages. And I will also describe what technologies I have chosen for the design of the website.

4.1 UML Diagram

UML diagram means unified modeling language, a standardized modeling language, enabling developers to specify, visualize, construct and document artifacts of a system. The UML architecture is based on Meta Object Facility, which defines the foundation for creating modeling language. UML diagrams are divided into two categories including structure diagrams and behavior diagrams. Now I will use activity diagrams schematically to show the performance of a given procedure of the website.

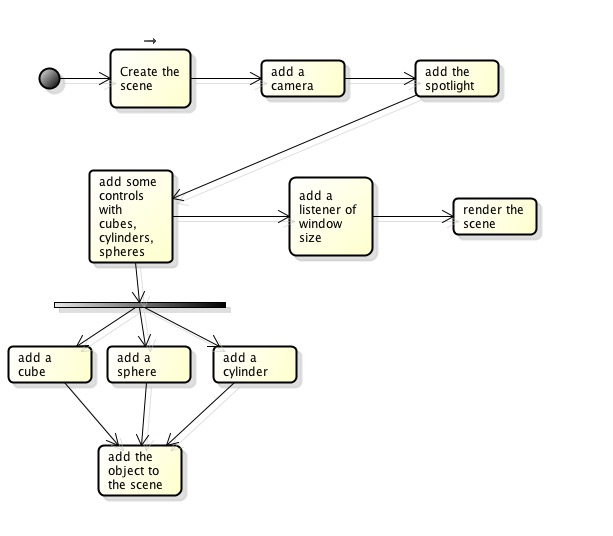


Figure 4.1: Website Activity Diagram

The activity diagram shown in Figure 4.1 represents the procedure for loading website. At the beginning, the user needs to visit the website. Then the website will create the scene. Once the scene is in place, I could create a camera. Next step is to add the spotlight. After all of these steps, the website will add control functions such as cube, cylinders and spheres. Until website gets the control functions, then it will add the object to the scene. Now the website runs well. Next step is to add an event listener. After all of these steps, I need to resize width and height of the browser. Then I need to render the scene through the camera I have added before.

4.2 Design Technologies

This 3D graphic website uses three.js downloaded from <http://threejs.org/>. Three.js is an object-oriented JavaScript library for 3D graphics. It is also an open-source project created by [Mr.doob](https://github.com/mrdoob), and it seems to be the most popular JavaScript library for 3D because JavaScript has a security feature called the same origin policy. I need set up the local environment firstly. Once the local environment has been set up, I need to create an index page by using HTML, which is an easy part.



Figure 4.2 index.html

In figure 4.2, it shows the JavaScript that I have stored in a folder called “js”. First, I need to create three files: index.html, main.js and style.css. Now download Three.js source. After this, edit the index.html page. And there is nothing special for the index page.

Three.js uses the concept of display list. That’s means all objects stored in the list and then drawn to the screen. So firstly I start to write the initialization function by setting up a scene.

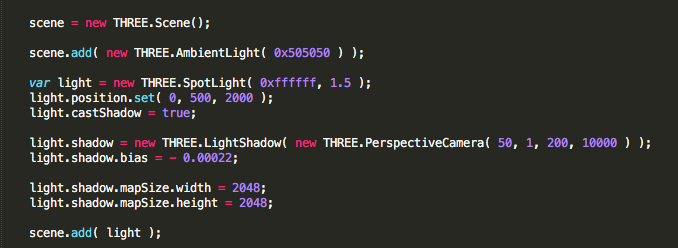


Figure 3.3 Set up the scene

Next, I will set up a three.js renderer which can draw simple things. I could use the SVG or canvas renderers, but I don’t want to choose them. I focus on WebGL renderer. That’s because it’s able to take advantage of the GPU. After creating the renderer, I append it to the DOM via the body element. This will make three.js create a canvas inside the body element that will be used to render the scene. In Figure 3.4, the first line defines the WebGL renderer. I can pass the renderer’s options in the first argument as a map. Here, I set true *antialias*, because I want the edge of objects to be smooth, not japed.

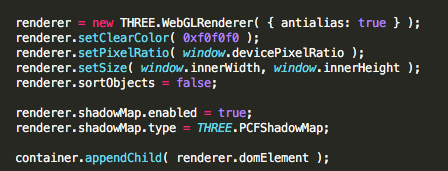


Figure 3.4 Create a renderer and add it to the DOM

Now, the scene and renderer are in place. For the purpose of rendering, I need to create a camera in the scene, so the renderer will identify where it should render the scene. There are a few types of cameras in Three.js. The reason why I chose the *PerspectiveCamera* is that it could take few parameters including FOV and Aspect etc. After the camera has been created, I set the position just by using coordinates. This type of camera is presenting the scene as we see the real world. Creating the camera is a bit more complicated.

Screen%20Shot%202016-08-04%20at%207.04.56%20PM.png

Figure 3.5 Create a camera

I have already added the camera to scene. Next, I need to add a light in order to see the 3D object, and I use a *PointLight* which gives out light like a bulb. I have also added a SpotLight to cast a shadow in one direction within a falloff cone. Shadows are an efficient tool when you make the scene more realistic.

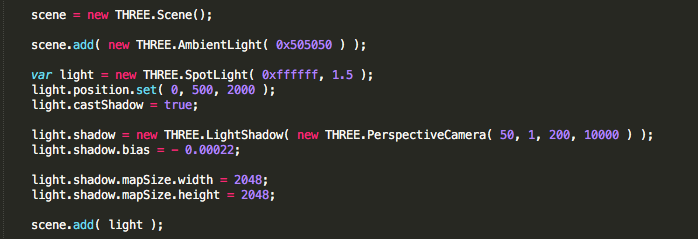


Figure 3.6 Create a light, set its position and add it to the scene.

Next step is the most important step that is to load geometry. In Three.js, the objects that are being drawn in the screen are called meshes. Each mesh must have its own geometry and material. Geometry is a set of points that need to be connected in order to create an object. Material is simply the paint that will cover the object. Upon the users’ requirements, I have added cube, sphere and cylinder to the scene.

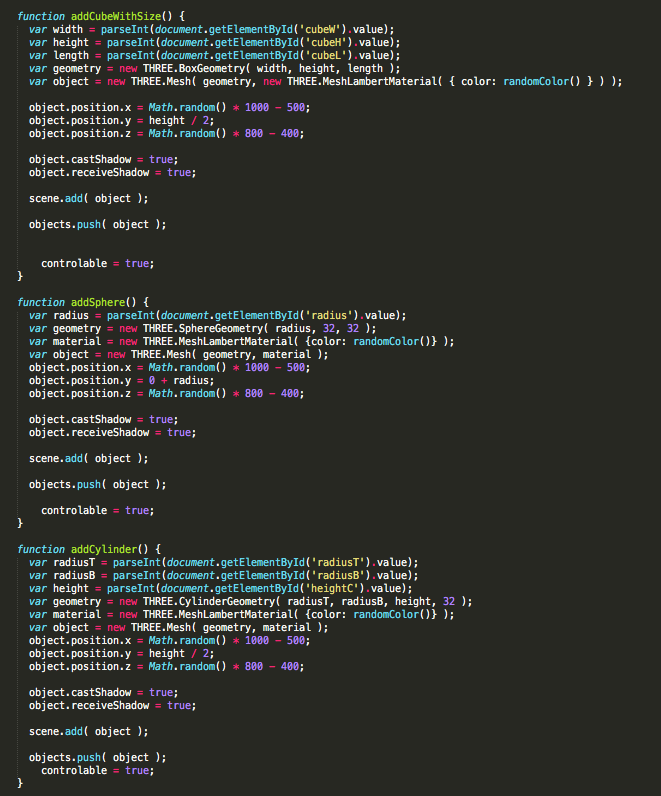


Figure 3.7 Load Geometry to scene

The last step is to add movement to the scene. By using *OrbitControls.js*, I rotate the camera around an object. It also allows the user using mouse wheel to zoom in and out the mesh.

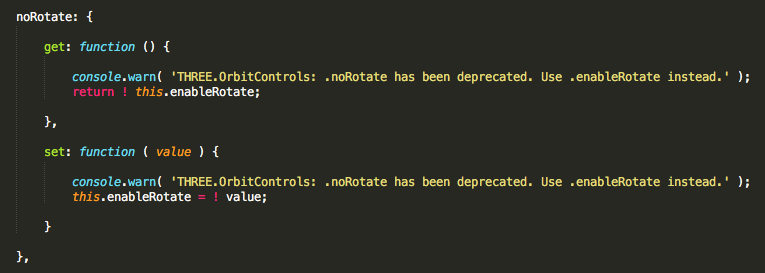


Figure 3.8 Add Orbit Controls

After I set up all functions, I need to finish up with animation function. Then, I need to render the scene through the camera that I have added earlier and then update the orbit controls.



Figure 3.9 Render the scene and update the renderer

Now, I have described all key functions in my project.

# Chapter 5

# Project Implementation

In this chapter I will show what the current version of this website looks like and what it can do. Now I have already uploaded this page to the web server, so users could visit the website through <http://utbiologists.noip.me>.

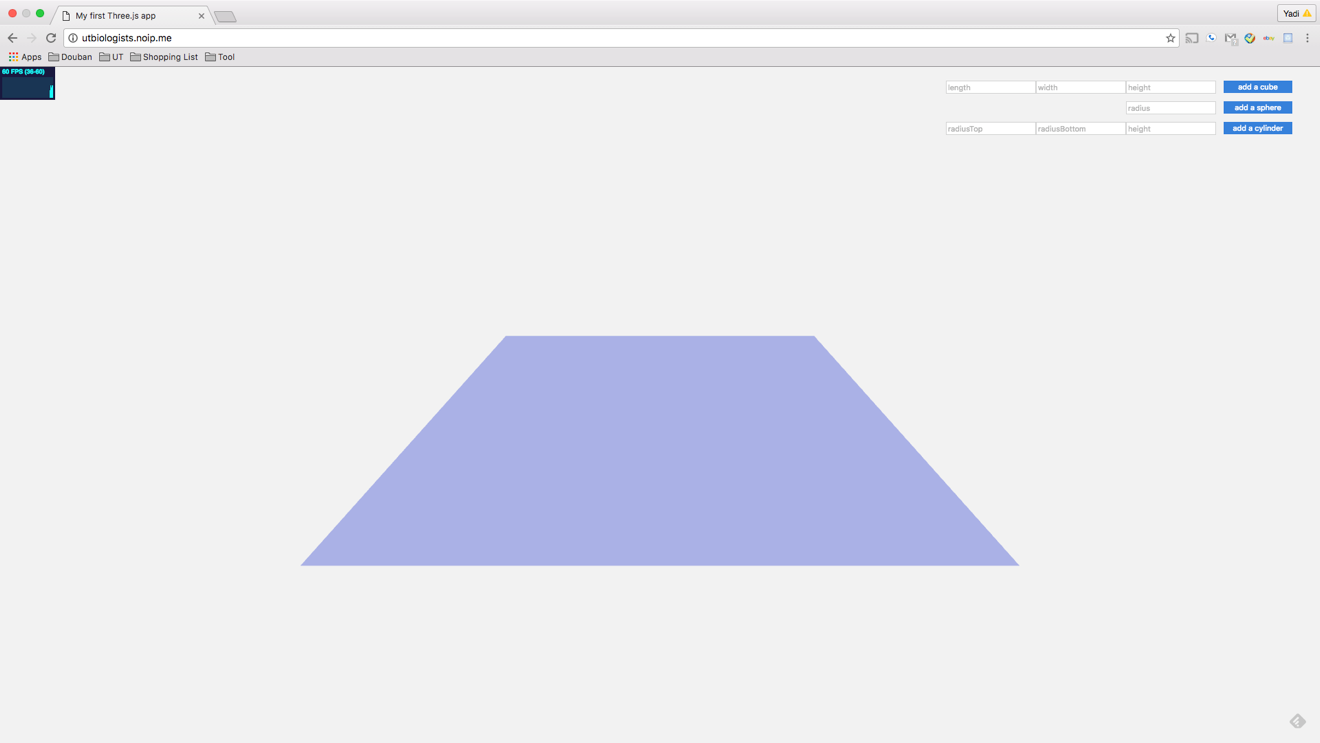


Figure 5.1 Home page

Users could click the input box in the upper right and enter the number with pixel unit. After that you could add geometries you want to. You can click the upper left corner to rotate the objects. If you click the objects and hold left click button, you can move the objects. You also can use mouse wheel to zoom in and out geometries. If you click the object which you want to delete, then press backspace key. I upload the demo video to my server, you can check from <http://utbiologists.noip.me/Video.mp4>.

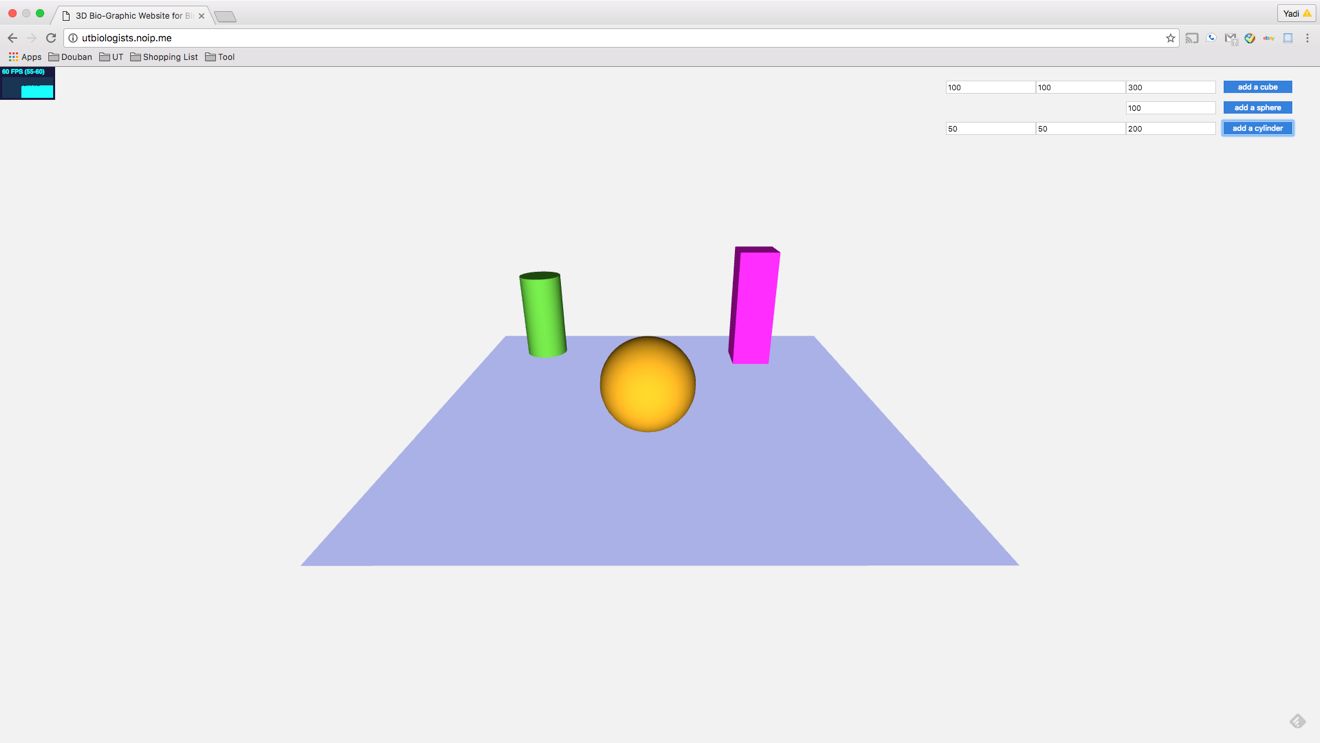


Figure 5.2 Add Geometries

**Chapter 6**

# Conclusions

In this chapter, I will conclude what I have discussed and described throughout this report. Then I will give some details about what may come next in the future development of this project.

I stated that I will give reader an overview of the problems the users wanted to address. They want to have a 3D graphic website which is interface friendly and clear to non-computer major biophysicists.

6.1 Future work

This website is still a work in progress and it is far from completion. I hope to continue to work on it, enhancing already existing features and adding new ones. In the remaining part of the report, I will try to focus on what we would like to add to the website and what may come next.

6.1.1 Import molecular models from Molecular Flipbook

For what concerns the website there are an important feature is I would like to add a feature which could import molecular models from Molecular Flipbook directly. Even if this a step further from the job done. I did not want add too many complicated functions at first, but sooner or later this will be necessary to users.

6.1.2 Improve the user interaction with the view

I would like to collect feedback on the UI from the viewer as many as possible and try to figure out a way of giving the maximum control over the user interface elements in the 3D scene.

**Reference**

1. Molecular Flipbook Website-home of Molecular Flipbook (Accessed: 2016-8-3). <https://www.molecularflipbook.org/>.
2. Three.js home page(Accessed:2016-2-9) http://threejs.org/
3. K. Wiesent, K. Barth, N. Navab, P. Durlak, T. Brunner, O. Schuetz, and W. Seissler, “Enhanced 3-D-reconstruction algorithm for C-arm systems suitable for interventional procedures,”   IEEE Trans. Med. Imaging **19**(5), 391–403 (2000).<http://dx.doi.org.proxy.ohiolink.edu:9099/10.1109/42.870250>
4. O. Faugeras, Three-Dimensional Computer Vision (Artificial Intelligence), (MIT Press, Cambridge, 1993).
5. A. C. Kak and M. Slaney, Principles of Computerized Tomographic Imaging (IEEE, New York, 1988).
6. Geroimenko, V.: Augmented Reality Technology and Art: The Analysis and Visualization of Evolving Conceptual Models. In: 16th International Conference on Information Visualisation, p. 1 (2012)
7. Borchers, J.O.: A Pattern Approach to Interaction Design. John Wiley & Sons (2001)
8. Brown, B., Bell, M.: CSCW at play: ‘There’ as a collaborative virtual environment. Proc. CSCW’04, Chicago (2004)
9. Taivalsaari A, Mikkonen T (2011) The web as an application platform: the saga continues. In: 37th EUROMICRO conference on software engineering and advanced applications, pp 170–174
10. Fiser A, Do RK, Šali A (2000) Modeling of loops in protein structures. Protein Sci 9:1753–1773Dong, K., Nan, K., Tilak, S., Zheng, C., Xu, D., Schulze, J., Arzberger, P., Li, W.: Real time biomedical data streamingplatform (rimes): A data-intensive virtual environment 2, 342–346 (2010)
11. Sowizral HA, Deering MF (1999) The java 3D API and virtual reality. IEEE Comput Graph Appl 19(3):12–15
12. Roccatello, E., & Rumor, M. (2009). Design and development of a visualization tool for 3D geospatial data in CityGML format (UDMS Annual). Amsterdam: CRC Press.
13. Eswar N, Madhusudhan MS, Marti-Renom MA, Šali A (2005) PROFILE\_SCAN: A module for fold-assignment using profile-profile scanning in MODELLER
14. Madhusudhan MS, Marti-Renom MA, Sanchez R, Šali A (2006) Variable gap penalty for protein sequence-structure alignment. Protein Eng Des Sel 19:129–133
15. Herráez A (2006) Biomolecules in the computer: Jmol to the rescue. Biochem Mol Biol Edu 34:255–261
16. Kuntal BK, Aparoy P, Reddanna P (2010) EasyModeller: A graphical interface to MODELLER. BMC Res Notes 3:226–330