PyMOL mControl: manipulating molecular visualization with mobile devices

Wendy WT Lam, Shirley WI Siu[§]

Department of Computer and Information Science, University of Macau

Avenida da Universidade, Taipa, Macau, China

[§]Corresponding author

Email addresses:

WWTL: lamwunteng@gmail.com

SWIS: shirleysiu@umac.mo

Abstract

Viewing and manipulating three-dimensional (3D) structures in molecular graphics software are essential tasks for researchers and students to understand functions of molecules. Currently, the way to manipulate a 3D molecular object is mainly based on mouse-and-keyboard control that are usually difficult and tedious to learn. While gesture-based and touch-based interactions are increasingly popular in interactive software systems, their suitability in handling molecular graphics has not yet been sufficiently explored.

Here, we designed the gesture-based and touch-based interaction methods to manipulate virtual objects in PyMOL utilizing the motion and touch sensors in a mobile device. Three fundamental viewing controls–zooming, translation and rotation–and frequently used functions were implemented. Results from a pilot user study reveal that task performances on viewing controls using a mobile device are slightly reduced as compared to mouse-and-keyboard method. However, it is considered to be more suitable for oral presentations and equally suitable for education scenarios such as school classes. Overall, PyMOL mControl provides an alternative way to manipulate objects in molecular graphic software with new user experiences. The software is freely available at http://cbbio.cis.umac.mo/mcontrol.html.

Running title

PyMOL mControl

Keywords

Gesture-based and touch-based interface; molecular visualization; mobile device; PyMOL

Background

Three-dimensional (3D) structures of molecules are crucial to the understanding of molecular functions in the study of life sciences [1, 2]. Molecular structures can be downloaded from the PDB database [3] or generated by structural bioinformatics methods. They can be visualized or edited subsequently using molecular graphics software. To date, many such software programs exist in the market, commercial or free, to help students and researchers to quickly display, animate, or render high-quality images for their needs in studies and publications.

Nevertheless, the way to manipulate a 3D molecular object in these software programs is still limited to mouse-and-keyboard control in a conventional computer system. The mouse is an excellent and inexpensive device for object manipulation in two-dimensional (2D) applications. However, since the mouse is moved on a planar surface, it is not intuitive to use it for manipulating virtual objects in 3D environment. To support more direct way to control, advanced input devices such as 3D mouse, data gloves, and gamepads are necessary [4]. Immersive virtual reality techniques, which allow one to fly through the microscopic worlds and directly interacting with the system, could be fascinating but the cost involved is high and the setup is too complicated [1]. In recent years, with the advance of sensor technologies, the study of gesture-based input methods for interactive software has received intense focus. Solutions using spatial recognition devices such as Leap Motion or Kinect for molecular visualizations have been reported [5-8]. While these non-standard devices are not always available (such as in the school laboratory), sensor-equipped mobile devices can be considered as alternatives. The advantage of using a mobile device is that almost all of us bring along with it wherever we go. Many mobile devices nowadays have powerful computing capabilities, built-in touch screens and sensors that can be exploited to capture user's motions and subsequently translated into object manipulation commands in the software. For example, Fiorella et al. used the multi-touch interface of a mobile phone to perform 3D scene navigation [9]. Such technique was later integrated into the popular desktop molecular visualization system VMD [10]. Regarding the use of motion sensors, Sasakura et al. made a preliminary study on the use of acceleration data from an iOS phone to detect the orientation-free direction of leaning or movement of the device [11]. The design was tested using an in-house developed molecular visualization system, but usability of the designed gestural interactions was not clear. Besides, the second motion sensor commonly equipped in mobile devices nowadays–the gyro sensor–was not studied.

Therefore, in this work we aim to investigate the effectiveness of gesture-based interactions in controlling 3D molecular objects using both motion and touch sensors in a mobile device. The goal is to design natural, intuitive, and easy-to-use gestural interactions with sufficient precision for beginners and frequent users. Based on the PyMOL framework [12], we developed a mobile controller called *PyMOL mControl* using a client-server architecture. For the purpose of platform compatibility, the client-side program was implemented as a web app, so the software requirement for the mobile device is simply a web browser that supports the use of sensor data. Since the original PyMOL user interface is difficult to use, a more user-friendly interface, which could shorten the learning curve and reduce user’s frustrations in the learning process, is desirable. For this purpose, frequently used operations such as selecting atoms, changing display style, coloring, saving scenes, etc. were also implemented. Generally, a molecular graphics software is only controlled by one person whom is sitting in front of the computer. With PyMOL mControl, everyone with a mobile device can participate as a "controller" to manipulate the displayed molecule. This feature is particularly suitable in group meetings for collaboratively adjusting the view of the molecule to facilitate group discussions.

In the following sections, we will present the design of PyMOL mControl and the usability study of it as a viable alternative to mouse-and-keyboard control of 3D molecular objects. In the pilot study, we briefly introduced to users about our designed mobile gestures for zooming, rotation and translation in PyMOL mControl, then we observed how users completed the assigned tasks by themselves. Experimental data was statistically analyzed to compare the efficiency of PyMOL mControl with the mouse-and-keyboard control.

Software Design

Client-server architecture

Figure 1 shows the client-server architecture of PyMOL mControl: The server module is loaded as a plugin in PyMOL. It can receive user events in real time from the connected mobile device; then it processes the data and maps it to PyMOL operations and commands such as zooming, rotation, translation, atom selection, and display update. On the other hand, the client module is a web app running on a mobile web browser. It utilizes hardware access APIs to sense user events. Collected data from user events is transferred to the server via WebSocket subsequently (See sections S1 and S2 in Supporting Information).

PyMOL mControl server

For the PyMOL mControl server, we used Tkinter module—Python's standard graphical user interface package—to build the user interface of the plugin. **Figure 2**(A) displayed the interface of the loaded plugin. Upon receipt of user event data from the mobile client, the data will be translated into corresponding PyMOL commands. For this, the *cmd* module in PyMOL is utilized to manipulate molecular objects by performing zooming, rotation, translation, and atom selection, etc. In addition, PyMOL *cmd* is also used to retrieve PyMOL internal data which is needed by the mobile client for user information.

Web app client

The mobile client was implemented using HTML5. A powerful feature of HTML5 is that it provides Application Programming Interfaces (APIs) to access in real time the sensors equipped in a mobile device. The three hardware APIs that are utilized in the web app client are the Touch Event API, the Device Motion Event API, and the Device Orientation Event API. The Touch Event API is for handling events caused by changes of the state of contact in the touch screen; the Device Motion Event API is for handling events caused by changes of the device's acceleration including and excluding effect of gravity; the Device Orientation Event API is for handling events caused by changes of the device's orientation.

Regarding the user interface design, the idea is to make it simple and intuitive to use. The amount of information displayed is limited to prevent cluttering and all user functions can be completed within a few steps. Snapshots of the web app client user interfaces and their descriptions are presented in **Figure 2** (B)-(H).

Compatibility issues

WebSocket is efficient for network communication in client-server model, however, it has not yet been supported by all mobile browsers. Users can refer to the web site in [13] to check for the compatibility of their browsers. Regarding access to sensor data in web app client, the current version of Android browser (version 4.4.4) and Chrome for Android (version 42) do not support Device Motion API based on our tests. Therefore, certain sensor-dependent functions are disabled when running the web app client in these browsers. See Table 1 for the summary.

Design of gesture-based and touch-based interactions

In the mouse-and-keyboard control of PyMOL, rotation and zooming of molecular objects are achieved by left-click and right-click on the mouse buttons respectively. If the mouse has three buttons, translation is also supported by holding the mouse middle button; otherwise, a combination of mouse and keyboard control is necessary (e.g. Ctrl+mouse left-click). The main disadvantage of such mouse-and-keyboard control is that the associations between commands and physical actions are not obvious. As a consequence, users may mix up the keyboard shortcuts and usage of mouse buttons, and eventually run into errors. From the perspective of human-computer interaction (HCI), the interaction design of PyMOL is considered low usability because the operations are difficult to learn and hard to memorize [1413].

To circumvent this problem, we have designed gesture-based and touch-based interactions for three basic viewing controls of molecular objects in PyMOL. They are zooming, rotation, and translation. As illustrated in **Figure 3**, the design idea of mobile gestures is to create an intuitive mapping of a viewing control to user’s motion. The gesture designed for object rotation is an exemplar: A molecular object in PyMOL is three dimensional just like the mobile device. The user can simply imagine that he or she is holding the molecule at hand and rotate it to a desired orientation by doing so to the mobile device. Ideally, user's motions around all three axes should be captured and translated to the molecular rotation. At the implementation level, we figured out that combining sensor data of all three axes can cause data confusion and lead to some imprecisions in the viewing control. To solve this, we separated the XY-rotation control from the Z-rotation control by requesting the user to notify the system before the corresponding gesture is used. As a consequence, orienting the mobile device along the X and Y axes results in XY-rotation of the molecule, and along the Z-axis results in Z-rotation of the molecule.

For the zooming control, in reality, *zoom in* and *zoom out* correspond to user actions that make the object to appear bigger or smaller. With a camera, the user will pull the zoom lever to adjust the view; however, if you are holding an object, you will draw it closer or move it farther away to obtain the *zoom in* and *zoom out* effects. For PyMOL mControl, we designed the tilt-and-wave gesture: The user shall tilt the mobile device slightly to his or her side and wave it towards himself or herself to perform the *zoom in* viewing control. Similarly, he or she shall tilt the mobile device slightly away from his or her side and wave it against himself or herself to perform the *zoom out* viewing control.

For the translation control, the most intuitive way is by *direct manipulation*, i.e. to pick up an object and to drag it to the desired position; this can be achieved by using the touch screen of the mobile device. In the main page, a touch panel is provided at the left-hand side of the interface. The user can simply slide his or her finger in the touch panel to move the molecule.

For the implementation details about how to convert the raw sensor data into PyMOL viewing controls, we refer the readers to sections S3 and S4 in Supporting Information.

Besides the three basic viewing controls discussed above, we have implemented the menu-based interfaces to allow the user to select atoms, chains or residues, to modify their display representations and colors. Also, the user can record displayed scenes for quickly retrieving them later.

Results

To test the effectiveness of the designed interactions in PyMOL mControl and to understand user's experience on molecular viewing via mobile controller, we conducted a pilot user study of both novices and experienced PyMOL users. The software evaluation procedure was like this: First, each participant was briefly introduced about the standard mouse-and-keyboard viewing controls in the PyMOL interface. Then, the participant was briefed about the gesture-based and touch-based controls using the mobile device. After practicing for a few minutes, the participant was given three tasks to complete in PyMOL: (1) To zoom and translate a given molecule to a specific view; (2) to rotate a given molecule to a specific orientation; (3) to select and change the representation and color of the molecule. The participant had to perform all tasks independently using firstly the mouse-and-keyboard control and then PyMOL mControl. The time to complete each task was recorded. Finally, the participant was asked to fill in a questionnaire to provide information about his or her proficiency in mobile devices, molecular graphics software, and the experience in gesture-based interactions. He or she was also asked to evaluate the suitability of PyMOL mControl in different usage scenarios.

All 15 participants were researchers or students from University of Macau. They are either bachelor or master students majoring in Computer Science or Biomedical Science. Majority of them took courses of bioinformatics, biology or chemistry previously, or performed research in bioinformatics, and thus they had some to moderate degree of knowledge about molecules. For each participant, an informed consent form was signed before the evaluation began. On average, the evaluation was completed in 10 minutes including the required tasks and the questionnaire.

Figure 4 presents the summary of participants' experiences on mobile devices, PyMOL and controller-based gesture recognition applications. Generally, our participants have the habit of using mobile devices. Although most of them do not habitually use PyMOL, almost half of them have some experiences on it during their studies (data not shown). In addition, they have little experiences on controller-based gesture recognition applications such as Wii.

Numerical results

Quantitative measurements of the overall usability and effectiveness of viewing controls were obtained by recording completion times of the three tasks. Based on

Wilcoxon signed-rank test, we used $p < 0.05$ as the significance level to determine the difference in user performance between mobile control and mouse-and-keyboard control in PyMOL. If the p-value is small, we can conclude that the difference is significant.

Figure 5 shows the time taken for users to complete each of the three tasks: Zoom + Translate, Rotate, Select + Styling. For the Zoom + Translate task, the median time to completion using PyMOL mControl was significantly longer than mouse-and-keyboard ($p = 0.02$). Similarly, for the Rotate task the median time to completion using PyMOL mControl was significantly longer than the mouse-and-keyboard ($p = 0.02$). For the Select + Styling task, there was no significant difference between the two approaches ($p = 0.691$). Precisely, for the first two tasks, users spent approximately 70% and 36% more time with mobile control than mouse-and-keyboard control to complete the tasks. These results are not surprising as users are generally more familiar with the use of mouse and keyboard than mobile device for virtual object manipulations; besides, the training time allowed for practicing PyMOL mControl was rather short. Reduced task performance in gesture-based interaction was also observed by Sabir et al. using Kinect [13]; the same reason was postulated.

Nevertheless, during the tests we recorded that about 25% of users made incorrect operations with the mouse-and-keyboard control and needed to try different combinations a few times before success. This confirms us that it is easy for users to mix up the keyboard shortcuts and usage of mouse buttons especially for infrequent users, whereas gesture-based interaction is more direct and intuitive although the precision of control in our program has still to be optimized.

Subjective results

Figure 6 shows how users evaluated the suitability of two different viewing controls in three usage scenarios. For personal usage, PyMOL mControl was deemed less suitable than mouse-and-keyboard control (p = 0.032). However, for giving presentations, mouse-and-keyboard control was judged to be significantly less suitable than PyMOL mControl (p = 0.029). For educational scenarios such as lab classes, the PyMOL mControl was judged to be equally suitable as mouse-and-keyboard control (p = 0.718).

Regarding the intuitiveness of the designed interactions for three viewing controls, 26.7% of the users selected “yes”, 60% selected “more yes than no”, and 13.3% selected “no more than yes”. Finally, 80% of users agreed that PyMOL mControl is useful.

We also obtained comments from users about how to improve the system. Since the translation control was supported by touch-based interaction, one suggestion was to include zooming control using multi-touch as a shortcut to the zooming gesture. Besides, some users preferred having the molecule displayed on their mobile phone and one user suggested the use of tablets instead of mobile phone to support more informative display on the mobile device.

Discussion

Overall, users were quite positive about PyMOL mControl as a supplementary input method to mouse-and-keyboard even though task performances might be slightly affected. Besides the training time problem as mentioned above, device sensitivity to user's motions might be another reason causing the reduced task performance. During the evaluations, we set the default values for step sizes of the zooming and translation controls according to our own experience. But different users might prefer different device sensitivity. Therefore, without optimizing these settings we believed that some users might feel uncomfortable. Among the three tasks tested, the task which was not affected by this problem was changing the display style of the molecule. In fact, users performed similarly in mouse-and-keyboard and mobile control. The corresponding operations in PyMOL mControl for this task were only selection-based rather than gesture-based, so the comparative performance can be attributed to the simple and familiar user interface design.

In the current implementation, translation control is touch-based which is different from zooming and rotation controls. The reason of this design choice was the limitation in accurately identifying user's translation motions from the sensor data. Nevertheless, we believe that supporting gesture-based translation would make the overall interaction design more intuitive and so user motions for viewing controls would become more coherent. This idea is probably feasible: As shown in the study of Sasakura and co-workers [11], the orientation free movement of the device can be differentiated from the device leaning by analysing signal patterns returned from the acceleration sensor. The problem remains to be solved is to determine the distance of movement, or positioning of the device, which is now being an active research in the area of mobile computing [15,16]. The main obstacle is the inherent measurement errors in the sensors and noise in the data coming from unintentional user motions such as small vibrations. In the study of human writing recognition using accelerometer in mobile phones, the authors proposed to reset velocity to zero between pauses of strokes to correct displacement inaccuracy [17]. Such technique may be applicable to measure precisely the translation motion and this will be further explored in our future work.

Our evaluation results show that users prefer the conventional mouse-and-keyboard control when performing single-user tasks. However, for giving presentations, PyMOL mControl is considered a better choice as the speaker is not physically restricted to the location of the computer. He or she can move freely and at the same time control the molecular visualization as desired. In addition, interactions between the speaker and audiences can be more coherent and natural. A special feature in PyMOL mControl is its ability to handle multiple client connections and allows switching control among them. This is particularly suitable for activities involving group of people such as meetings and school classes. In these scenarios, any attendee who has a mobile device can participate and adjust the molecular view with minimal interruption to the on-going discussion. Switching control between teacher and students in school classes can enrich interactions, foster attention and enhance learning.

Currently, although multiple clients are allowed in PyMOL mControl, only one client can be active at one time. We believe that simultaneous and collaborative control would be even more beneficial in different group activity scenarios and this would be a direction for our future work.

Conclusions

This paper presents a new software design of PyMOL mControl for viewing control of 3D objects in the molecular graphics software. Sensor technologies equipped in a mobile device including the touch screen, gyroscope and accelerometer are used to detect user motions to realize three viewing controls – zooming, translation and rotation. Commonly used functions in PyMOL such as atom selection, changing the molecular display style and coloring are also provided to enrich software utility.

Evaluation of PyMOL mControl was done by gathering both numerical measurements and subjective feedbacks from 15 users, novice or experienced, from University of Macau. Results were statistically analyzed to measure the effectiveness and intuitiveness of using PyMOL mControl as compared to the conventional mouse-and-keyboard control. Overall, task performances are slightly affected when using gesture-based and touch-based control, but the use of mobile device as remote controller is welcomed especially for presentations and educational scenarios. Based on these encouraging results, our future work will focus on improving the precision of user motion detection, exploring different user gestures, and enhance functionality for collaborative viewing control.

Acknowledgements

The authors are thankful to the time and valuable feedback of all participants in the experiments. The authors declare that no competing interests exist.

References

1. S. I. O'Donoghue, D. S. Goodsell, A. S. Frangakis, F. Jossinet F, R. A. Laskowski, M. Nilges, H. R. Saibil, A. Schafferhans, R. C. Wade, E. Westhof , A. J. Olson (2010) Visualization of Macromolecular Structures, Nature Methods 7, S42–S55.
2. P. A. Craig, L.V. Michel, R. C. Bateman (2013) A Survey of Educational Uses of Molecular Visualization Freeware, Biochem. Mol. Biol. Educ. 41, 193-205.
3. F. C. Bernstein, T. F. Koetzle, G. J. Williams, E. E. Meyer Jr., M. D. Brice, J. R. Rodgers, O. Kennard, T. Shimanouchi, M. Tasumi (1977) The Protein Data Bank: A Computer-based Archival File For Macromolecular Structures, *J. Mol. Biol.* 112, 535-542.
4. E. Moritz, T. Wischgoll, J. Meyer (2007) Comparison of input devices and displays for protein visualization, *The ACM Student Magazine*, 1-14.
5. I. Jamie, C. McRae C (2011) Manipulating Molecules: Using Kinect for Immersive Learning in Chemistry. Proceedings of the Australian Conference on Science and Mathematics Education.
6. K. Sabir, C. Stolte, B. Tabor, S. O’Donoghue (2013) The Molecular Control Toolkit: Controlling 3D Molecular Graphics via Gesture and Voice. Proceedings of the IEEE Symposium on Biological Data Visualization (BioVis 2013), pp.49-56.

7. S. M. Waldon, P. M. Thompson, P. J. Hahn, R. M. Taylor II (2014) SketchBio: a Scientist's 3D Interface for Molecular Modeling and Simulation, BMC Bioinformatics 15, 334.
8. W. Chinthammit, S. Yoo, C. Parker, S. Turland, S. Pedersen, W.-T. Fu (2015) MolyPoly: a 3D Immersive Gesture Controlled Approach to Visuo-Spatial Learning of Organic Chemistry. Proceedings of the 25th Australian Computer-Human Interaction Conference (OzCHI 2013), LNCS 8433: pp.153-170.
9. D. Fiorella, A. Sanna, F. Lamberti (2010) Multi-Touch User Interface Evaluation for 3D Object Manipulation on Mobile Devices, J. Multimodal User Interfaces 4, 3-10.
10. VMD Remote: http://www.ks.uiuc.edu/Research/vmd/plugins/remote/.
11. M. Sasakura, A. Kotaki, J. Inada (2011) A 3D Molecular Visualization System with Mobile Devices. Proceedings of the15th International Conference on Information Visualisation (IV), pp. 429-433.
12. The PyMOL Molecular Graphics System, Version 1.7.4 Schrödinger, LLC.
13. Web Sockets. http://caniuse.com/#search=websocket.
14. L. Grell, C. Parkin, L. Slatest, P. A. Craig (2006) EZ-Vis, a Tool for Simplifying Molecular Viewing in PyMOL, Biochem. Mol. Biol. Educ. 34, 402-407.
15. M. Liu M (2013) A Study of Mobile Sensing Using Smartphones, International Journal of Distributed Sensor Networks, Article ID 272916.
16. E. Macias, A. Suarez, J. Lloret (2013) Mobile Sensing Systems, Sensors 13 (12), 17292-17321.

17. S. Agrawal, I. Constandache, S. Gaonkar S (2009) PhonePoint pen: using mobile phones to write in air. Proceedings of the 1st ACM Workshop on Networking, Systems, and Applications for Mobile Handhelds, pp. 1-6.

Figures

Figure 1 The client-server architecture of PyMOL mControl.

The server (lower box) is implemented as a PyMOL plugin utilizing PyMOL commands to manipulate the visualization of molecules. The client (upper box) is running as a web app, utilizing the WebSocket API to communicate with the server and other event APIs to capture system events and user motions.

Figure 2 User interfaces of the PyMOL mControl server and the mobile client.

A) Main interface of the server displaying the IP address, the port number, and the number of connecting clients. B) Web app client login. C) Main page of the client after login: The displayed molecule is shown in the title bar; translation control of the molecule is done via the touch panel and rotation control can be activated via the Rotation button. The user can save the current view as a scene for fast re-display and switch between perspective and orthoscopic view. D) The general setting page is to adjust the step sizes for the motion controls and mode of collaborative control if there are multiple clients. E) Usage instruction page. F) Rotation control page: the user has to first specify the current motion, either XY-rotation or Z-rotation, then he/she can rotate the device to perform the actual rotation on the displayed molecule. G) Atom selection and display style setting page. H) Color setting for selected atoms.

Figure 3 Interaction design for three viewing controls.

A) XY-rotation: Rotate the mobile device about its X and Y axes. B) Z-rotation: Rotate the mobile device about its Z-axis. C) Zooming: Tilt the mobile device slightly to the user and then wave it towards himself or herself to zoom in; tilt it away from the user and then wave it against himself or herself to zoom out. D) Translation: Slide the finger on the screen along the desired direction. All user actions are performed with the mobile device in the landscape mode.

Figure 4 Summary of the participant's experience in A) mobile device, B) PyMOL, C) gesture-based applications (5:excellent, 1: poor).

Figure 5 User performances in the three tasks. A) Zoom and translate, B) Rotate, C) Selection and changing display style. The thick line in the shaded box represents the median of the distribution of values whereas the top and bottom boundaries of the box indicate the 75th percentile and 25th percentile of the distribution. Outliners are represented as star or dot (with the sample ID) whose values are 1.5 times greater or smaller than the interquartile range. The upper and lower whiskers indicate the extreme values excluding the outliners.

Figure 6 Subjective assessments of how different controls are suitable for three usage scenarios (see caption in Figure 5 for the explanation of the plots).

Tables

Table 1 - Compatibility table of existing mobile platforms in supporting 3D object viewing controls in PyMOL mControl web app client.

	iOS	Android	Chrome for Android
Zooming	✓	✗	✗
XY-Rotation	✓	✓	✓

Z-Rotation	✓	✗	✗
Translation	✓	✓	✓