ODYSC: A responsive educational web app for dynamics and control

Kevin Dekemele * **Amélie Chevalier Mia Loccufier**

* *Ghent University, Department of Electrical Engineering, Metals, Mechanical construction and Systems, Technologiepark 914, B-9000, Ghent, Belgium (e-mail: kevin.dekemele@ugent.be)*

Abstract: The Online Dynamical Systems and Control (ODYSC) web application has been developed for the introductory dynamics and control courses taught to engineers and business engineering of Ghent university, accessible through `http://www.odysc.ugent.be/`. Generally speaking, the advantage of web apps over conventional software is the compatibility between operating system (OS) and devices, as only a browser is required. Recently, students bring a mobile phone to lectures. While these are often used by the students themselves as a distraction, they are actually very powerful computers capable of performing simulations of dynamical systems. Because of this, ODYSC has been designed to be responsive, working equally well on both mobile phone as on a laptop or PC. While ODYSC can serve as a distance learning tool, it can also be employed during classical lectures. While the teacher explains new concepts in dynamics or control, these can immediately be made clear through simulations on the student's phone. Before, the amount of students attending these courses were a practical limitation for organizing computer labs. Even if such infrastructure exists, the software used requires the students to code in a new language, which they are often not motivated enough to do for a single lab, or the software is associated with costly or complicated licensing. ODYSC requires no special infrastructure but a device with a browser which students themselves bring anyway. ODYSC is still in full development and for now can perform open loop and closed loop simulations on LTI systems, and plot bode plots of the system.

Keywords: control education, computer software, web application, responsive, mobile

1. INTRODUCTION

Students attending introductory dynamics and control courses often feel overwhelmed by many different concepts and parameters even before the concept of feedback is introduced. Interactive computer simulation can aid understanding of the students, but require either an enormous amount of computers or so-called laptop lecture rooms. In the latter case, each students requires an installation of the software used. Often this software (e.g. MATLAB) is proprietary and requires complex licensing for either local installation or use through a central server which requires constant connection to this server. In Ghent university, the dynamic systems and control simulation tool of choice is MATLAB or Python. Although the researchers and teaching assistants are fluent in these languages, the engineering students attending the introductory dynamics course have diverse background and are often not willing or able to learn a new computer language for just a single computer lab. Even if they are, concepts as simulators, sampling time and time horizon are unknown to them. To solve problems with infrastructure, complex and costly licensing, and avoid extra required expertise from the students than just the introductory dynamics, it was opted to develop a responsive web application using the most recent web technologies HTML5 and Javascript ES6. It is dubbed the Online Dynamical Systems and Control (ODYSC) and can be found on `http://www.odysc.ugent.be/`.

The choice to develop a web application has several reasons; 1) Web applications are made to shield the user of the complexity behind the problem and aim to be intuitive for a layman 2) They are often single-purpose to not overwhelm the user, avoiding the feature-shock associated with classic software suites 3) Their inter-device compatibility and minimal installation requirements, as a web app only requires a device to have a (script-enabled) web browser 4) requires only a small initial download, and is run on the device itself (so-called client side) which implies no constant connection a server. These reason makes a web app the perfect platform for educational purposes as opposed to classic software suites as Matlab and Python, which are more suited for research purposes. A control web application has been developed as early as Schmid (1998). Although first of its kind, it was plagued by the limitations of the web technologies of that time (the security plagued Java applets), required a local installation of MATLAB and required either infrastructure as giant computer labs, or a PC owned by the students. In present time, the web scripting language Javascript has evolved from a mere script language to interact with forms to a full fledged programming language capable of doing scientific computing, with an overview of several libraries found in Qian (2016), lifting the need for specialized suites as MATLAB.

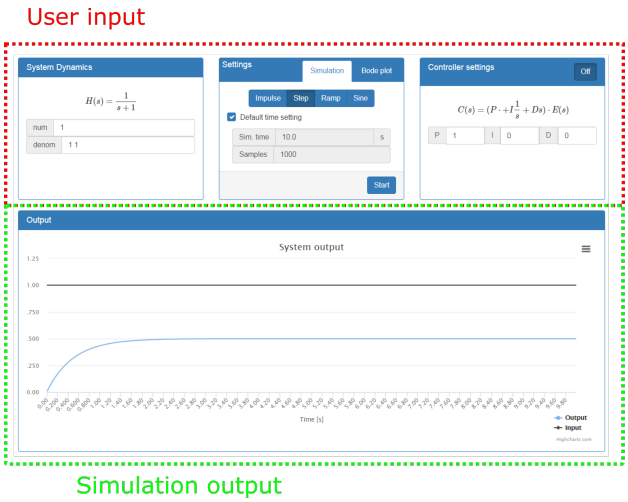
The advances made since 1998 in terms of laptop and mobile phones have resulted in the students carrying one or both devices at all times. So the inter-device compatibility provides many opportunities. If students themselves bring their mobile phone or laptop to courses, it leads to the frustration of the lecturers as these devices could be used to distract themselves or others. However; the author rather sees this as an opportunity to be seized, instead of a obstacle to be tackled.

A web app can made responsive, meaning that it is designed by work equally well on mobile phone as on a PC. This way, requirement for huge infrastructure, special devices or the need for a separate computer lecture can be lifted, as the students bring a compatible device with them anyway. In engineering, web education is often used separately from the class room, as a remote learning tool, Humar et al. (2005); Bourne et al. (2005); Long et al. (2014), where students could either prepare lectures or follow strictly on line courses, or through virtual or remote labs De Jong et al. (2013); Chevalier et al. (2017). In Boticki et al. (2013), a Android-based smartphone was developed to explain sorting algorithms to computer science students. The visualization and interaction with these algorithms helped, according to their survey, to increase understanding of these algorithms as well as increase the grades. Although the smartphone app was simple and required only a small installation, it could only run on Android smartphone, which the author claimed only 25 % of the attending students had. Therefore, it is better to opt for a inter-device compatible web app. With ODYSC, a lecture can now comprise of theory, while simultaneously be supplemented by interaction with the students by presenting numerical examples of theoretical concept. This simultaneously theoretical and computer lecture is a novel concept in web-based education. Although it is originally developed as a tool to use during lecture, it can perfectly be used in distance learning and massive open online courses (MOOCS)

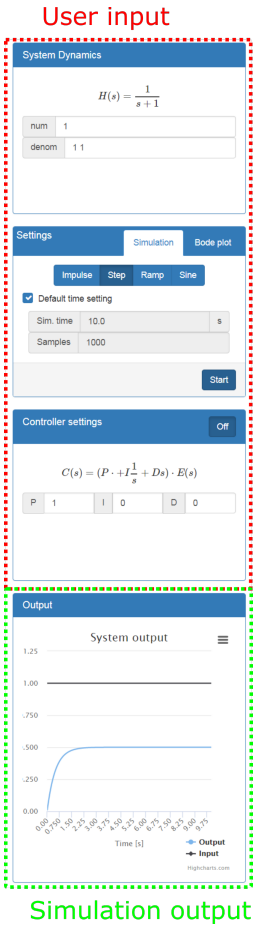
The paper is structured as follows, first the technologies behind the web app are explained, followed by an explanation of the lay-out. Then, several examples are presented using ODYSC, as a tutorial on how to use the web app in a lecture. Finally, the results of a first survey are presented.

2. WEB TECHNOLOGIES

The web layout has been designed using HTML5, stylized with CSS3 and scripted with Javascript ES6. Although all of these web technologies have been available since 1996, the ability to create a responsive, scientifically computing web app using only these languages has only been made possible by the recently developed libraries. Before, the first control web application (Schmid (1998)) required a local installation of MATLAB to overcome limitation of web technologies at the time. Remote labs on the other hand have the required software installed on the server (Chevalier et al. (2017)). This removes the software requirements on the client's (student) side, yet requires a constant connection between client and server. ODYSC doesn't have either requirements, only needing a web browser.



(a) The interface on big screen such as laptop, PC and high-end tablets.



(b) The interface on smaller screens as mobile phones.

Fig. 1. The layout of ODYSC on different devices.

A vital library is the CSS and HTML library called Bootstrap. This library allows to develop a single responsive layout, which present equally well on a mobile or a PC. This way, web developers don't have to develop different version of a website for different screen sizes. For simulating the linear systems, partial fractions are implemented by porting the MATLAB function *residue* to Javascript

by the author, assisted by many scientific libraries found on Qian (2016). A big advantage of HTML, CSS and Javascript is that they are client-side. This means, besides an initial download (for ODYSC 102 kb) when loading the page, there is no contact needed between server and client. The minimal size is dwarfed by conventional, feature-ridden software suits, which require downloads of several hundreds to thousands MB and long installation procedure with awkward licensing or registration.

3. LAYOUT

On PCs or other devices with high resolution screens the layout is divided in an upper and lower part, see Fig. 1a. On a mobile device the layout will collapse into 4 stacked compartments, Fig. 1b. Both layouts have a user input part, and a simulation output part. The user input compartments will be discussed one by one.

3.1 System Dynamics

First, the open loop dynamics have to be defined as a linear transfer function, a ratio of two polynomials in s:

$$H(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \ldots + b_1 s + b_0}{a_n s^n + a_{n-1} s^{n-1} + \ldots + a_1 s + a_0} \tag{1}$$

The dynamics need to be entered as a *vector* of the coefficients of the numerator under num and the denominator under denom, similar to MATLAB notation. Each coefficient is separated by a space, so in the num input form $b_m\ b_{m-1}\ \ldots\ b_1\ b_0$ and the denom $a_n\ a_{n-1}\ \ldots\ a_1\ a_0$. As a decimal a point should be used, not a comma.

As an example, the following transfer function is entered in Fig. 2a:

$$H(s) = \frac{s+5}{s^3 + s^2 + 6s} \tag{2}$$

A coefficient which is zero should also be typed explicitly.

3.2 Simulation Settings

The settings compartment has two tabs, one for (time) simulation and one for bode plot. Depending on which tab is active, the button 'start' will either perform the time simulation, or show the bode plot. Several option have a default value, for a lower threshold for beginner students.

Here, the settings for a time simulation are set, see Fig. 2b. The first set of buttons sets the input to apply to the system.

- impulse, the dirac delta $u(t) = \delta(t)$ or $U(s) = 1$
- step, $u(t) = 0$ for $t < 0$, $u(t) = 1$ otherwise, or $U(s) = \frac{1}{s}$
- ramp, $u(t) = 0$ for $t < 0$, $u(t) = t$ otherwise, or $U(s) = \frac{1}{s^2}$
- sine, $u(t) = 0$ for $t < 0$, $u(t) = sin(\omega t)$ otherwise, or $U(s) = \frac{\omega}{s^2+\omega^2}$

If the sine input is chosen, a window will be prompted, asking for the frequency of the sine, $f = \omega/2\pi$. Next, if 'start' is clicked, the resulting output will be shown in the graph.

To overcome lack of knowledge about simulation time and times, the sim.time, the time to simulate is chosen by default as the 1% settling time of the open loop system, calculated with the slowest poles. The amount of samples during this time is set to 1000. Experienced users can change these settings.

3.3 Bode Settings

Here, the settings for a bode plot are set, see Fig. 2c. A bode plot reveals the gain of the output $y(t) = A(f)sin(2\pi ft + \phi(f))$ for a sine input $u(t) = sin(2\pi ft)$. $A(f)$ is the gain and $\phi(f)$ the phase shift of the output compared to the input. The bode plot $H(j\omega)$ is derived from the transfer function by replacing $s = j\omega = 2\pi jf$. Its absolute value is $|H(j2\pi f)| = A(f)$ and is plotted when pressing 'start'. The user has a choice to plot the bode in logarithmic scale or linear scale.

Limits of the axis have a default setting, or can be chosen as manually. The default setting is from the lowest breakpoint/10 up until the final breakpoint $*10$.

3.4 Controller

This tab determines if the the simulation is performed in open loop or closed loop. In open loop, the button of the controller is not pressed and says on, see Figure 2e. The simulation is started, the following output will be shown (in time):

$$Y_{OL}(s) = H(s)U(s) \tag{3}$$

If the controller tab is open, Figure 2d, the parameters for a parallel PID controller reveal themselves:

$$C(s) = P + \frac{I}{s} + Ds \tag{4}$$

When compared to the classic PID implementation, $P = K_c$, $I = \frac{K_c}{T_i}$ and $D = K_c T_d$. If a simulation is started now, the following closed loop simulation will be performed:

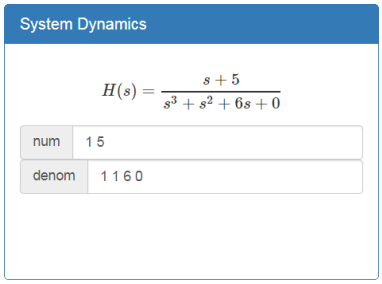
$$Y_{CL}(s) = \frac{C(s)H(s)}{1 + C(s)H(s)} U(s) \tag{5}$$

3.5 Output

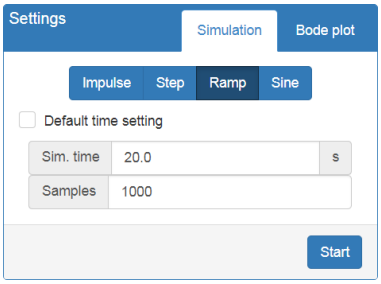
Depending on what settings tab is chosen, the output will either show a time simulation, Fig. 3a or a bode plot, Fig., 3b. By default, the applied input is not shown in the graph, but can be overlaid by pressing 'input' in the lower right corner, Fig. 3a.

4. EXAMPLE SCENARIOS

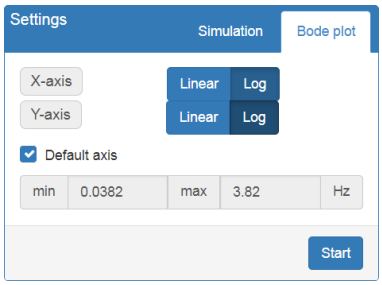
Two example scenarios are presented where together with a theoretical concept, a simulation can be performed to aid understanding. ODYSC is not limited to this use, it can also be used for distance learning, preparing lectures and MOOCS.



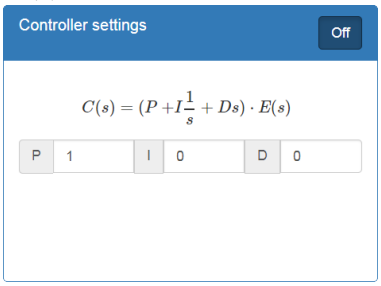
(a) The open loop system dynamics



(b) Settings for time simulations



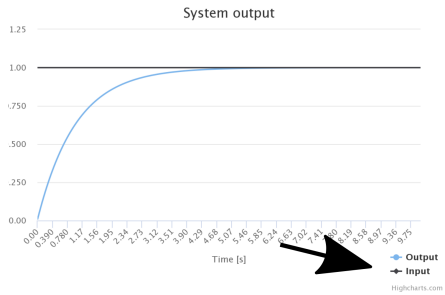
(c) Settings for time Bode plot



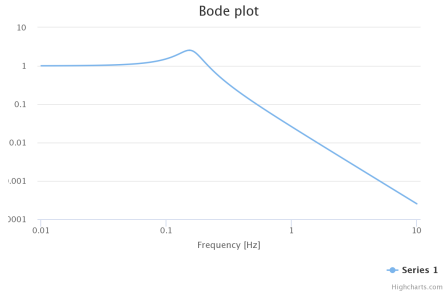
(d) Control tab is open, simulation are closed loop

(e) Control tab is closed, simulation will be open loop

Fig. 2. Each user input compartment



(a) The input can be overlaid by pressing 'input' in the lower right corner (pointed at with arrow)



(b) Bode on logarithmic scale, both for x and y axis

Fig. 3. Time simulations

4.1 Steady state error of feedback loop

The following open loop transfer function will be simulated first in open loop:

$$H(s) = \frac{2.5}{100s+1} \tag{6}$$

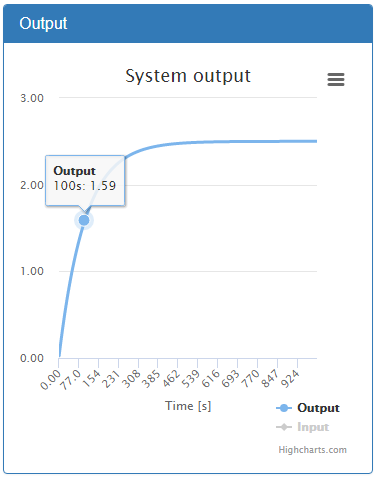
by typing num = 1 and denom = 100 1, and perform a step input time simulation. Students observe that 2.5 is the steady state value, and that at time equaling the time constant $\tau = 100$, the output is $2.5(1-e^{-1})$ or about 63% of the final value, Figure 4a. The students can now easily interpret the time constant and static gain of this first order system. If this system is controlled in feedback (put controller on), they can observe a much faster transient. Consider the P controller for $P = \{1.25, 2.5, 3.75, 25\}$ and a step input with Fig. 4b for $P = 2.5$. Although much faster dynamics can be observed, there always is a steady state error, so the output is always different from 1. It is known that a PI-controller will remove this error in case of a step input. Students can try different configurations, e.g. $P = 4$ and $I = \{0.04, 0.2, 0.4, 4\}$, see Fig. 4c for $P = 4$ and $I = 0.4$. Students observe no steady state error but now second order dynamics, although the open loop process is a first order! A shocking observation for them at first. If now a ramp input is applied, the output is not able to follow the input, revealing again a steady state error, even though a PI controller is applied.

4.2 Relation Bode and Sinusoid input

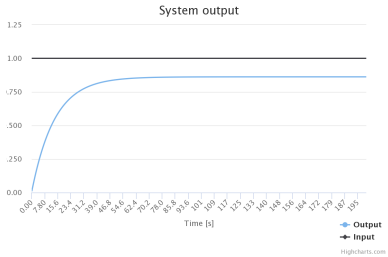
Although the bode plot can be easily derived from the transfer function $H(s)_{s\to j\omega}$, the relation of the bode plot and sinusoid input can immediately be made clear with ODYSC. Consider the follow transfer function

$$H(s) = \frac{1}{s^2+0.4s+1} \tag{7}$$

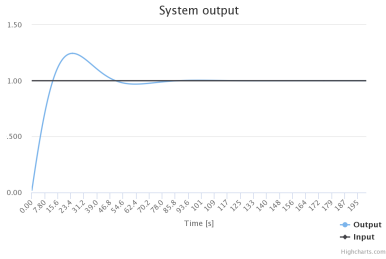
as num = 1 and denom = 1 0.4 1. Choose the bode settings instead of simulation settings, log both in x and y, and default axis. The result is Fig. 3b. Observe that for



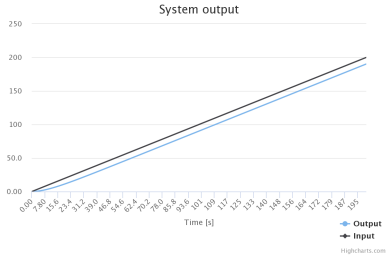
(a) Step response first order system



(b) A P controlled first order system.



(c) A PI controlled first order system.



(d) A PI controlled first order system with ramp input.

Fig. 4. A first scenario, a controlled first order system

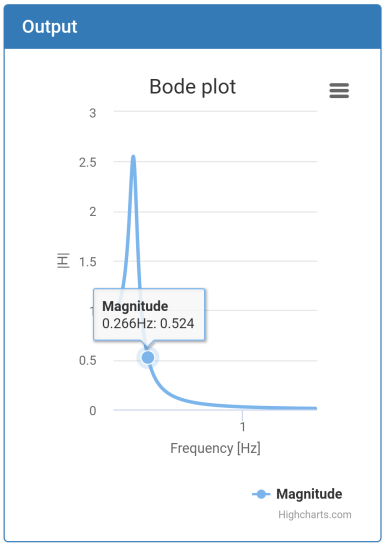
frequency $f = 0.26\ Hz$, the gain of the bode plot is about 0.5, Fig 5a. By definition of the bode plot, this means, if a sine input is applied of $0.26Hz$, the output has half of the amplitude of the input. Go back to simulation options, apply a sine as input in open loop and insert $0.26\ Hz$ as the frequency. An open loop simulation reveals that indeed, the output is half the amplitude of the input, Fig. 5b.

5. A FIRST SURVEY

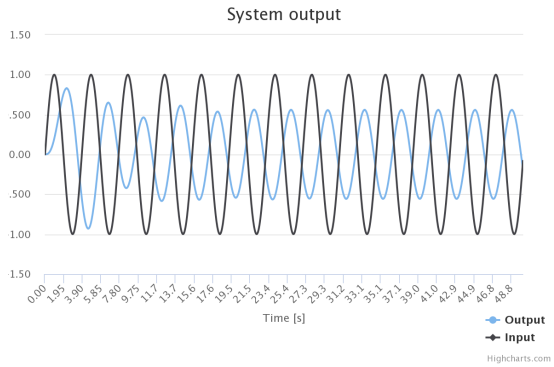
The ODYSC web app was presented to the Business Engineers in December 2017. Evaluation is obtained via a first survey with 56 participants, using a combination of a five-point Likert-type scale, multiple choice and open questions. In total 15 questions are asked divided into four sections: 1) Technical; 2) User interaction; 3) Educational value and 4) User feedback. For the first section, three multiple choice questions are posed. The results are presented in the three pie charts presented in Figure 6. Observe the diversity of the devices used, with no clear dominant device, confirming the need of inter-device compatibility. 17 out of 56 students were not aware dynamic systems could be simulated, indicating a significant need for the developed web app. Sections two and three of the survey, use a Likert-type scale with 1 indicating full disagreement with the statement and 5 indicating full agreement with the statement. For each question the mean value and the standard deviation are reported in Table 1. From this it is clear that in general user interaction is satisfactory. However, a lower mean value indicates a need for a user manual. The higher standard deviation indicates that some students need the user manual to work with the app, while other do not. For educational value, the survey clearly shows that students appreciate the web app to provide better understanding. However, the web app alone is not enough to fully grasp all concepts. Therefore, a combination of classical lectures with the web app would suite the students the most in order to grasp theoretical concepts. Section four is of the survey are two open questions about any problems that were encountered and possible suggestions for improvements. The results of these open questions showed again the need for a user manual and indicated some issues concerning the decimal point in the app which were resolved afterwards.

6. CONCLUSION

This paper present the ODYSC web app, suitable as a companion during dynamical systems lecture, or for distance learning. Because of its responsive design, students can access the app on any device with a browser while attending a lecture. This way, during theory or exercises, concepts and results can immediately be validated through a simulation, without the need of infrastructure associated with separate computer labs. For now, open loop and closed loop simulation can be performed, and a open loop Bode plot can be shown. ODYSC is still in full development and will be extended with other features to aid students in understanding basic concepts. A first survey was taken by business engineers and indicated an interest in ODYCS. Further feedback surveys will be taken in dynamics courses in business engineering and engineering technology in Ghent university.



(a) For frequency $0.26Hz$, the bode has a gain of about 0.5



(b) the output clearly is half the input, which can be explained with the bode plot

Fig. 5. Relation bode plot and sinusoid input

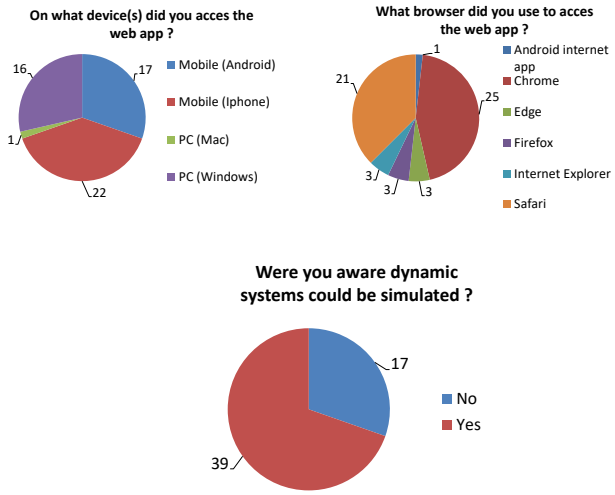


Fig. 6. Pie chart results of the technical section of the survey.

Table 1. Statistical analysis of the survey

User interaction		
Statement	Mean	Standard deviation
The web app is intuitive to use.	3.83	0.68
The speed of the web app is satisfactory.	4.41	0.68
The user interface is satisfactory.	3.82	0.73
There is a need for more information in order to fully understand how to work with the app.	3.57	1.08
Accessing and using the web app is easier than typical software used in courses.	3.70	0.78
Educational value		
Statement	Mean	Standard deviation
The web app improves your understanding of dynamic systems and feedback?	3.79	0.92
The level of the examples was adequate?	3.61	0.96
There is a need for more web apps to further understand theoretical concepts ?	3.41	0.88
There a need for an offline version of the web app?	3.18	1.36
I gained as much information as I would from a lecture explanation?	2.96	0.94

REFERENCES

Boticki, I., Barisic, A., Martin, S., and Drjevic, N. (2013). Teaching and learning computer science sorting algorithms with mobile devices: A case study. *Computer applications in Engineering Education*, 21, 41–50.

Bourne, J., Harris, D., and Mayadas, F. (2005). Online engineering education: Learning anywhere, anytime. *Journal of Engineering Education*, 94, 131–146.

Chevalier, A., Copot, C., Ionescu, C.M., and Keyser, R.D. (2017). A three-year feedback study of a remote laboratory used in control engineering studies. *IEEE Transactions on Education*, 60, 127–133.

De Jong, T., Linn, M.C., and Zacharia, Z.C. (2013). Physical and virtual laboratories in science and engineering education. *Science*, 340(6130), 305–308.

Humar, I., Sinigoj, A.R., Bester, J., and Hagler, M.O. (2005). Integrated component web-based interactive learning systems for engineering. *IEEE Transactions on Education*, 48, 664–675.

Long, J.M., Cavenett, S., Gordon, E., and Joordens, M. (2014). Enhancing learning for distance students in an undergraduate engineering course through real-time web-conferencing. In *Proceedings of the the 2014 American Society for Engineering Education International Forum*, 11024.

Qian, T. (2016). Scientific computation libraries written in javascript. URL `https://github.com/timqian/scientific-computation-libs-in-javascript`.

Schmid, C. (1998). The virtual control lab vclab for education on the web. *Proceedings of the American Control Conference*, 1314–1318.