INTERACTIVE APPLICATIONS IN A MANDATORY CONTROL COURSE

Yves Piguet [*] **Roland Longchamp** [**]

[*] *Calerga Sàrl, av. de la Chablière 35, 1004 Lausanne, Switzerland. E-mail: yves.piguet@calerga.com*
[**] *Laboratoire d'automatique, EPFL, 1015 Lausanne, Switzerland. E-mail: roland.longchamp@epfl.ch*

Abstract: This paper describes how a mandatory control course is enhanced with interactive applications. Basic concepts of automatic control, different ways to assess performance and robustness of a controlled system, and limits of closed-loop systems are illustrated with graphics which can be manipulated directly with the mouse, letting the student acquire a deeper and broader understanding of theory. The applications, implemented with Sysquake as stand-alone programs, offer a simple user interface while supporting a wide range of problems.

Keywords: Interactive programs, computer graphics, teaching, automatic control.

1. INTRODUCTION

At the Swiss Federal Intitute of Technology in Lausanne (EPFL), like in many other engineering universities, undergraduate students of several cursus have a mandatory course in automatic control. Following a course in dynamical systems and modelling with Laplace transfer functions and state-space models, it provides an introduction to automatic control, sampling, z transform, frequency response, stability, discretization, direct synthesis in discrete time, and fuzzy controllers. It is followed by optional courses on two degrees of freedom polynomial controllers, identification, autotuning, and adaptive control.

Experience shows the importance of the course support material, such as a required textbook which closely matches the contents of the course, and software which illustrates the examples. The second edition of the textbook Longchamp (2006) gave the opportunity to review and extend the applications used for 12 years, and to include them with the textbook on a CD-ROM for Windows and Macintosh.

Classical automatic control has introduced different kinds of graphics, such as time-domain simulations, frequency responses, Nyquist diagrams, root locus, and others. Computers have been a mean of generating them since they exist. With the introduction of personal computers and software like Matlab, it became possible to change parameters and generate new graphics relatively easily and quickly, even during ex cathedra courses.

A further step, thanks to the constantly increasing computation power of computers, was interactive graphics, where the user can change parameters with the mouse by manipulating graphical elements such as sliders or representation of the poles in the complex plane. This idea was implemented in the early ninetees in different places; implementations were limited by the underlying software tools which were used. At EPFL, in 1994, an application developed in C, initially for research on robust pole placement (see Piguet et al. (1997)), was quickly brought in the classrooms for the mandatory control course (see Piguet et al. (1999)). The potential of interactive graphics led to the development of Sysquake (see Calerga Sàrl), software with a programming language which extends its scope to many fields, for teaching, research and development (see Dormido (2003); Dimmler and Piguet (2000)).

The remainder of the paper is structured as follows. Section 2 details the requirements for effective applications. Section 3 illustrates these requirements with the description of three applications. Section 4 details why and how Sysquake was used for their development. Conclusions and perspectives are given in Section 5.

2. PURPOSE OF APPLICATIONS

The mandatory control course is given to undergraduate students before they choose specialization fields. Applications must be simple to use, so that students can focus on the topics which are illustrated, not software details. At the same time, applications should not constrain the students to a predefined learning path, but let them explore further the theory.

Experience shows that applications are best used in three phases: to reproduce textbook examples, to explore them in depth, and to solve other problems.

2.1 Reproduce textbook examples

In a first phase, the student will run the application as an enhanced version of the example described in the book. Instead of static figures which represent a predetermined, fixed value for each parameter, (s)he can observe the effect of each parameter change by manipulating it with the mouse. This expands the example representation beyond the two dimensions of planar printed graphics.

In a classical textbook, a way to show the effect of a parameter change is to have separate figures for different values of the parameter, or to represent multiple traces in the same figure, leading to graphical complexity. An interactive application shows continuously the effect of parameter changes and the relationship between different figures.

Parameters which can be manipulated with the mouse are represented by sliders when they are scalar, or by symbols in the complex plane when they represent complex values such as the zeros or poles of a transfer function. In some cases, graphics can be manipulated directly, for instance to change a gain by moving a response up or down.

2.2 Explore examples in depth

The description of textbook examples cannot cover all aspects of a problem. A lot can be learnt by exploring the problem for parameters out of the range of what could be considered as “sensible”.

Displaying optional plots helps the student to understand the relationship between different aspects of the same problem. This is especially important in the field of automatic control where many kinds of diagrams and performance criteria are used to describe the same setup of system and controller. Interactivity gives insight of which quantities are closely related and how controller parameters can be used. The student can acquire a broader understanding of the whole picture.

2.3 Solve other problems

Most applications are useful not only for specific examples, but also for solving other problems, such as homework exercises or larger projects. The controlled system models and the controller structure can be changed.

In the case of digital PID or two degrees of freedom controllers, the model can be specified as a continuous-time transfer function, which is converted to a discrete-time transfer function. Intersample behavior of the continuous-time system output can be studied.

3. EXAMPLES

This section presents three examples and show what interactivity adds to the learning process compared to static graphics.

3.1 Poles and time signals

The main part of the control course covers discrete-time systems, whose transfer functions are obtained through z transform. A prerequisite of the course is a continuous-time signals and systems course. Therefore, after an intuitive overview of automatic control, the course deals with the description of the z transform and properties of discrete-time systems.

The z transform maps a time signal to a function of the complex variable z. Signals involved in linear time-invariant systems with a finite number of states have usually z transforms which are rational functions of z. Two of the simplest rational functions have one real pole or a pair of complex conjugate poles. It is fundamental to have an intuitive knowledge of the relationship between the location of these poles in the z complex

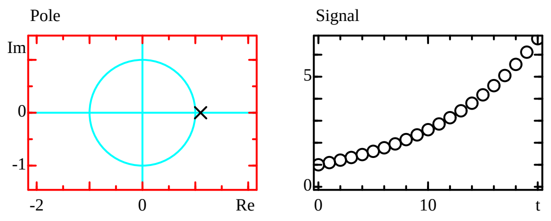

Fig. 1. Relationship between poles and time response (one unstable real pole).

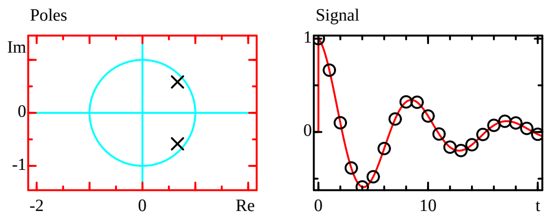

Fig. 2. Relationship between poles and time response (two stable complex conjugate poles).

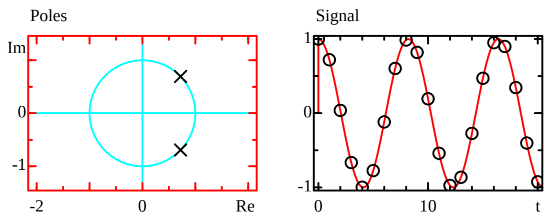

Fig. 3. Relationship between poles and time response (two stable complex conjugate poles on the unit circle).

plane and the corresponding time signal; this is related to important properties of systems, such as damping, oscillations, frequency responses, and stability, among others.

An interactive application illustrates this relationship. Figure 1 shows the case of a real pole outside the unit circle, which corresponds to a signal which diverges exponentially. The student can drag the pole with the mouse and see that the rate of divergence depends on the location of the pole. If the pole is inside the unit circle, the signal converges exponentially to zero.

The pole can also be brought out of the real axis, and transformed into a pair of complex conjugate poles (for real signals, poles are the zeros of polynomials with real coefficients, hence they are either real or make up pairs of complex conjugate numbers); see Figure 1. In the damped complex conjugate case, a wave is superimposed to the samples for enhanced clarity.

There are two particular cases which are interesting: poles on the unit circle, which is the frontier between converging and diverging signals (see Figure 3), and poles on the imaginary axis, where every second sample is zero. These cases are easily obtained with the mouse, because poles stick to the unit circle or the imaginary axis when the mouse is sufficiently close to them.

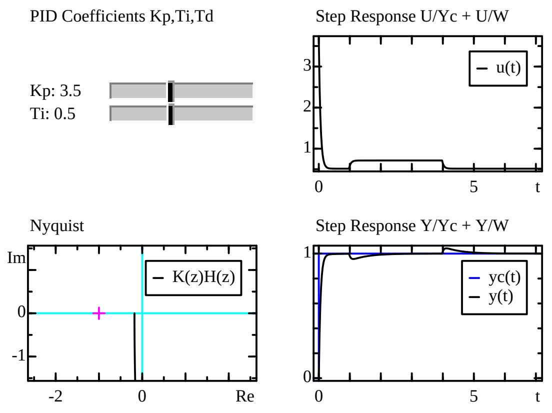

Fig. 4. Interactive application for the effect of the integrator term on set-point tracking and disturbance rejection in PI control.

3.2 Digital PID controller

The most popular controller structure is the PID controller, named after its three terms: the control signal (the input of the system) is the sum of a term “P” proportional to the error between the set-point and the measured system output, a term “I” proportional to the integrated error, and a term “D” proportional to the output derivative (many variants exist). Each term has a specific effect on performance.

An interactive application illustrates an example which shows in an extremely clear and intuitive way the effect of each term on different step responses and on the Nyquist diagram (Figure 4). Time responses at right show the responses to a unit step of the set-point at $t = 0$, and a -0.2 step of the input disturbance between $t = 1\,\mathrm{s}$ and $t = 4\,\mathrm{s}$. The time response at top right is the system input, and the time response at bottom right the system output. The effect of the integral term is considered here. One can demonstrate that an integrator causes the steady-state error to vanish. But what is the effect of the integral term gain? What is the transition between a purely proportional controller (without “I” term) and a PI controller? Moving the slider corresponding to the integral term gain (the integration time constant T_i is decreased, see Figure 5), the steady-state error between the set-point $y_c(t)$ and the system output $y(t)$, which stays constant when $T_i = \infty$, is reduced faster and faster. With a small value of T_i, the step response exhibits a large overshoot. The Nyquist diagram (omitted in the figure) shows that the frequency response comes close to -1, which means there is a reasonance peak.

3.3 Waterbed effect

The design of a good controller must take into account different objectives. Among them, the

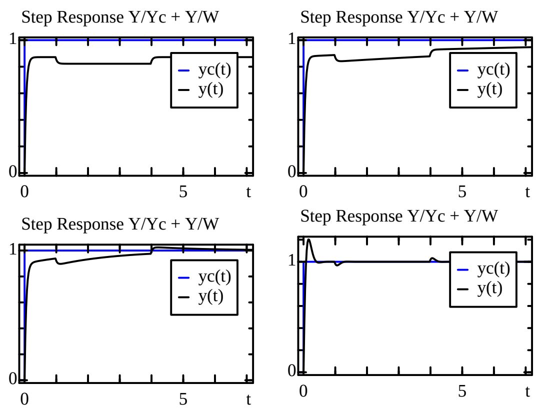

Fig. 5. Increasing effect of the integrator term on set-point tracking and disturbance rejection in PI control.

controlled system should reject perturbations at low frequency and be robust to model uncertainties. However, these objectives are known to be contradictory: it is not possible to achieve simultaneously arbitrarily good disturbance rejection and robustness.

One way to observe this is to consider the frequency response of the *sensitivity function* $S(z)$, defined as $S(z) = 1/(1 + L(z))$ where $L(z) = K(z)H(z)$ is the open-loop transfer function, i.e. the product of the controller $K(z)$ and the plant $H(z)$ transfer functions. The magnitude of the frequency response $S(e^{\mathrm{j}\omega h})$ (where $\mathrm{j} = \sqrt{-1}$, ω is the angular frequency and h the sampling period) is the inverse of the distance between the frequency response $L(e^{\mathrm{j}\omega h})$ and the critical point -1 in the Nyquist diagram. Robustness against model uncertainties is achieved by keeping this distance sufficiently large, or equivalently $\left|S(e^{\mathrm{j}\omega h})\right|$ small (typically lower than 10 dB). The sensitivity function $S(z)$ is also the transfer function between disturbances added to the output of the system and the disturbed output. To reject this kind of perturbation, it should be small, typically much smaller than 1 in the closed-loop bandwidth.

A naive way would be to request a small sentivity function magnitude for all frequencies. However, this is impossible to achieve, because of the following equation, given here without proof:

$$\int_{0}^{\omega_N} \log\left|S(e^{\mathrm{j}\omega h})\right| \mathrm{d}\omega = \omega_N \sum_{i=1}^{P} \log|p_i| \qquad (1)$$

where ω_N is the Nyquist angular frequency (half the sampling angular frequency $\omega_s = 2\pi/h$) and $p_i, i = 1, 2, \ldots, P$ the P poles of the open-loop transfer function $L(z)$ located outside the unit circle. Moreover, the closed-loop system must be stable.

A consequence of Equation 1 is that frequency ranges where the sensitivity function has a magnitude much smaller than 1 must be compensated with ranges where it is larger. This is known as the *waterbed effect*: when one pushes down the sensitivity response at one place, it pops up elsewhere. For unstable systems, this effect is even worse; it is a reason why we should avoid unstable controllers.

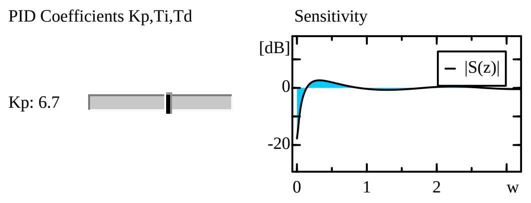

Fig. 6. Interactive application for waterbed effect.

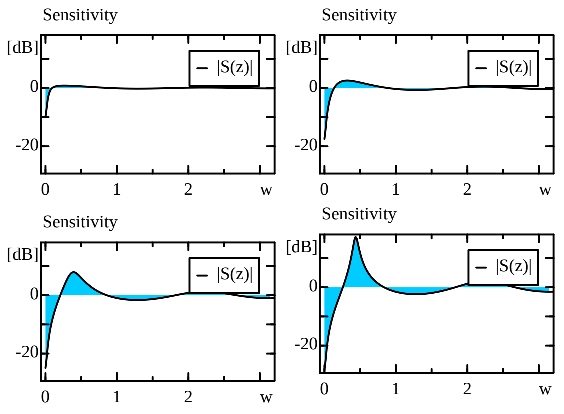

Fig. 7. Waterbed effect for different values of the feedback proportional gain.

Figure 6 shows how the waterbed effect is illustrated through an interactive application. The system is described by $H(z) = 0.015z^{-3}/(z - 0.985)$, and the controller is proportional. Since there is no open-loop pole outside the unit circle, the surface delimited by the sensitivity function magnitude, with the linear-dB scale, is the same below and above 0 dB. When the controller gain is changed, this property is preserved, as is shown in Figure 7.

The student who wants to explore more deeply the problem could display the Nyquist diagram and the time response to a disturbance step. (S)he would observe on one hand that a system like this one, with a time delay, has the modulus margin (given by the maximum of the sensitivity function magnitude, or the inverse of the distance between the Nyquist diagram and the critical point -1) which decreases when the feedback gain increases; and on the other hand, that the disturbance is rejected faster if the gain increases (the sensitivity function is smaller at low frequency). (S)he will better understand the relationships between different ways to analyze the system, and how to choose them. Changing the system to include an unstable system, with a pole outside the unit circle, will let her/him verify Equation 1.

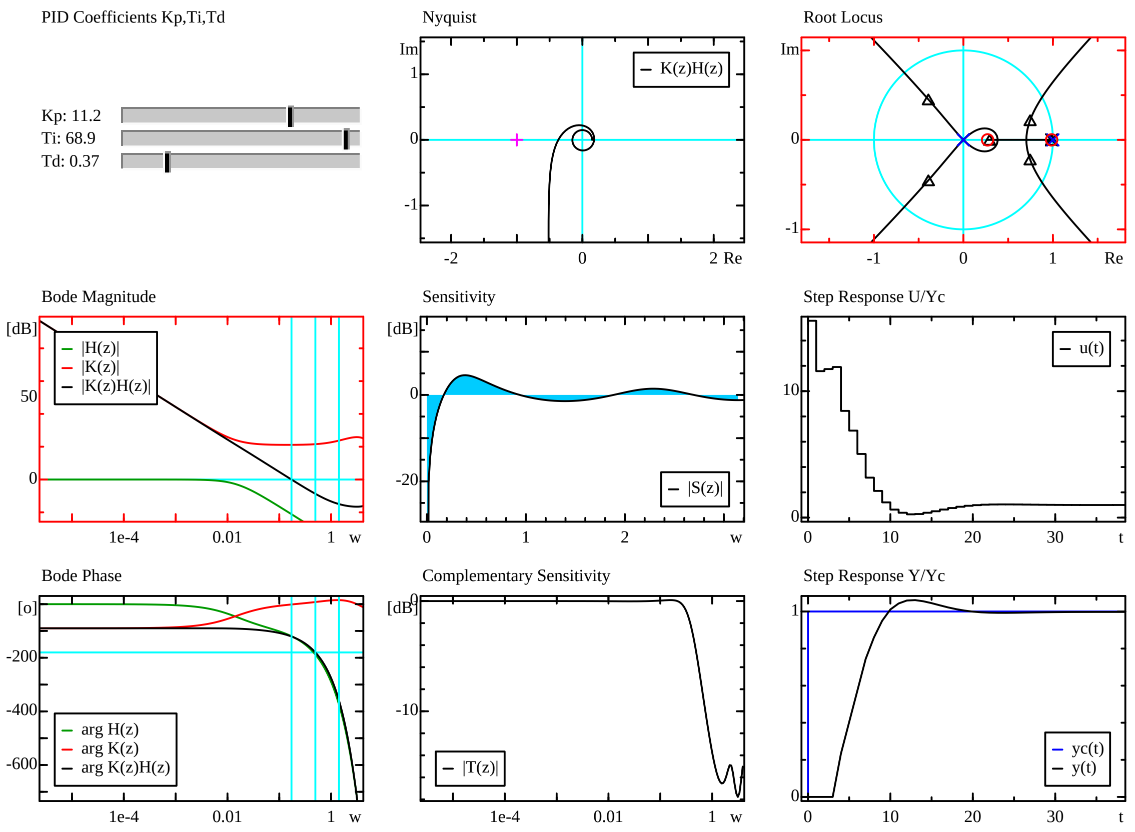

Fig. 8. PID controller with Nyquist diagram, Bode diagram, sensitivity and complementary sensitivity functions, and input and output responses to a set-point step; all diagrams are simultaneously updated when parameters are modified.

Other optional figures include the complementary sensitivity function, the Bode diagram, and the root locus (see Figure 8).

4. SYSQUAKE

Sysquake is a scientific application with the following features (see Piguet (2004)):

- Programming language compatible with Matlab, well suited to numerical computations, with native support for complex numbers and linear algebra.
- Graphics with low-level generic primitives and higher-level graphical commands for automatic control.
- Support for interactive graphics, i.e. graphics which can be manipulated easily with the mouse to modify the parameters they depend on.
- Runs on Windows and Mac OS X computers.

Sysquake applications are made of a description of the user interface, and the implementation of the algorithms in Sysquake's programming language. For each possible figure, a function defines how to display the graphics, using some parameters as input arguments; and (for interactive figures) a function defines how to change parameters, using mouse action as input argument. All figures are automatically synchronized, because they are based on the same parameters. Sysquake handles the undo/redo mechanism, zoom, scaling options, figure layout, etc. without further programming.

4.1 Applications included with Sysquake

The Sysquake software suite includes several interactive applications which cover a large part of the topics teached in the control course. This is to be expected, due to the origin of Sysquake. These applications are the result of a compromise: on one hand, they are sufficiently general and open to be used for solving typical problems a control engineer encounters. They can exchange data easily: for instance, the model obtained with parameter identification based on experimental data can be used for designing a controller.

One the other hand, the applications are simple enough to let the user understand easily the fundamental features of the system they represent. They do not implement all the possible refinements.

The applications which are provided for the control course are of two kinds: some of them are based on modified versions of the applications distributed with Sysquake, and the other ones have been developed specifically for the course.

4.2 Customization for control course

A large part of the course is concerned with classical control, i.e. analysis and design of controllers with single-input single-output transfer functions in the time and frequency domains, and use of step responses, ramp responses, Bode and Nyquist diagrams, and root locus. The main controller architecture is the digital PID (proportional-integral-derivative) controller. Other types are covered as well: two degrees of freedom controllers, adaptive control with model parameter identification, and fuzzy control.

One of the applications provided with Sysquake is dedicated to the design and analysis of digital PID controllers. It includes all these kinds of graphics. The PID parameters can be changed with sliders, by entering numerical values with the keyboard, or by manipulating the Bode diagram of the controller. This application has been modified to use the same notations as those of the course; new figures with combined responses to set-point and disturbance steps have been added; some minor modifications have been implemented, such as legends, colors, and scaling.

Other applications modified for the course include the simulation of a feedback loop with on-off controllers. An optional dead zone has been added; with the preexisting support of hysteresis, the most common types of on-off controllers can be simulated with systems represented by arbitrary transfer functions. For two degrees of freedom controllers, zeros can be added to the reference model to cancel steady-state errors.

The addition of new graphics does not sacrifice the application simplicity. At startup, an application has default settings for the value of all variables (the model and controller transfer function or the PID coefficients, for example) and the layout of figures. This is similar to the initial state of a dynamical system. For the same application, different initial states can be defined and stored in text files.

4.3 Applications specific to the course

Other applications, more limited in their scope, have been developed specifically for the course. This is the case for the relationship between poles and time responses, or between the complex planes of the Laplace and z transforms. The initial state is defined in the source code of the application, not in separate files.

4.4 Creation of stand-alone applications

Sysquake applications are usually stored as source code in text files. While it would not be a problem in an institution with a Sysquake license, this is not suitable for distribution with a book. Sysquake Application Builder, included with Sysquake, converts Sysquake applications with or without separate initial state files into stand-alone executable applications which can be distributed at no cost. For the control course, fifty stand-alone applications have been created for Windows and Mac OS X. They are distributed on the CD-ROM included in the book.

5. CONCLUSION

This paper has described how interactive applications are used in a mandatory course of automatic control. Emphasis is put on simplicity, so that the student can focus on the illustrated topic. Applications serve three successive purposes: enhanced illustrations of the course, ways to explore further the examples, and help to solve other problems. The applications, for Windows and Macintosh, are provided on the CD-ROM included with the textbook; they are also used during lectures.

While the release of the book marks an important step, the application development will not stop now. Further developments are expected, especially for the graduate control courses on multivariable and nonlinear control.

REFERENCES

Calerga Sàrl. Calerga web site. http://www.calerga.com.

M. Dimmler and Y. Piguet. Intuitive design of complex real-time control systems. In *Rapid System Prototyping 2000*, Paris, 2000.

S. Dormido. The role of interactivity in control learning. In *Proc. of 6th IFAC Symbosium on Advances in Control Education*, pages 11–22, Oulu, Finland, 2003.

R. Longchamp. *Commande numérique de systèmes dynamiques*. Presses polytechniques et universitaires romandes, Lausanne, 2nd edition, 2006.

Y. Piguet. *Sysquake User Manual*. Calerga, Lausanne, 4th edition, 2004.

Y. Piguet, U. Holmberg, and R. Longchamp. Multi-model weighted pole placement. *European Journal of Control*, 3:216–226, 1997.

Y. Piguet, U. Holmberg, and R. Longchamp. Instantaneous performance display for graphical control design methods. In *Proc. of the IFAC world congress, Beijing*, volume L, pages 403–408, 1999.