An Interactive Visualization of Polarized Light Decomposed into its Left and Right-Handed Components

Noah Rossignol
Rensselaer Polytechnic Institute, Troy, New York 12180

The study of polarization is an important part of any basic optics course. Knowledge about polarization enables one to understand its uses in everyday life, such as in polarized sunglasses and polarizing filters for photography, and is critical for understanding interesting optical phenomena such as birefringence and dichroism. Thus it is worthwhile to ensure good understanding of polarization and the way physicists think about it. One useful way to represent polarized light is with its representation as the superposition of two circularly polarized components. I present a program that allows students to explore polarized light represented this way with an interactive animation. The student can change the parameters in the equation that represents the light wave, and immediately see how those parameters affect the polarization state. It would be useful as a hands-on approach to demonstrating polarization and its decomposition into circular components, and could even be generalized as a demonstration of phasors.

I. Introduction

In 1985 Halloun and Hestenes published a study where they examined the results of testing students before and after taking college level courses in classical mechanics courses [1]. The results, based on their Mechanics Diagnostic Test which was designed to assess conceptual understanding of Newtonian mechanics, showed that many students showed minimal gain in understanding from their physics course. This prompted a good deal of research into understanding how to improve physics education. In 1998 Hake published a paper where he took a large survey of mechanics test data to evaluate how much improvement there was. Specifically he evaluated whether courses heavily based on "Interactive Engagement" produced a better gain in understanding than "Traditional" courses [2]. What he found was that students who took courses that made heavy use of interactive engagement techniques did significantly better than those who took more traditional courses that were based primarily on passive lectures and "recipe labs."

The evidence points to the idea that including interactive, hands-on activities improves comprehension, at least in physics courses. I make the assumption that that this idea would hold over in optics as well as in classical mechanics. This provides the motivation for designing an interactive tool to understand polarization.

The difficulty I have set out to address is understanding the representation of polarized light as the combination of two circular components. From personal experience taking taking Fundamentals of Optics, I remember that we spent more class time on this subject than on the simpler decomposition into x and y components. The circular decomposition is certainly less intuitive. Another difficulty is that when people think of polarized light they usually think of linearly polarized light. I know that the idea of having circularly polarized light was a new concept for me when I took the course.

The goal of the program I have made is to clarify this concept with an interactive visualization. My hypothesis is that by seeing how the two components and their sum change in time, and by being able to interactively modify the parameters optics students would more easily be able to understand the circular decomposition. My work is largely inspired by the math and physics applets from falstad.com [3], which are excellent tools for understanding many phenomena from math and physics.

II. Background

Here I present some background material on the concepts my program is intended to demonstrate. This background is drawn from Chapter 8 of the fourth edition of Optics by Eugene Hecht [4]. Light is described as an electromagnetic wave. Consider a light wave traveling in the z direction in standard Cartesian coordinates. At any given point the electric field is always parallel to the x-y plane, and how it varies in that plane determines what its state of polarization is. For linear polarization the E field oscillates along a straight line, changing its magnitude sinusoidally, but always forming the same angle with the x or y axis. For circularly polarized light the magnitude of the electric field is constant, but it is constantly changing direction tracing a circle in the x-y plane. Elliptical polarization is between the two; the electric field vector traces out an ellipse.

The simple way to represent the electric field vector at a particular position is with its x and y components

$$\vec{E}(t) = E_{0x}\cos(\omega t)\hat{i} + E_{0y}\cos(\omega t + \varphi)\hat{j}$$

In this case if φ is a multiple of π then the light is linearly polarized, If $E_{0x} = E_{0y}$ and φ is an odd multiple of $\pi/2$ then the light is circularly polarized, and otherwise it is elliptically polarized.

Circularly polarized light can be characterized as left-circularly polarized and right-circularly polarized. If the electric field vector is rotating clockwise as seen by an observer toward whom the wave is moving, it is classified as right-circularly polarized. If it spins counterclockwise, it is left-circularly

polarized. Light can be represented as the superposition of a right-circularly polarized component (the R-state) and a left-circularly polarized component (the L-state).

$$\vec{E}(t) = E_{0L} e^{i\omega t} + E_{0R} e^{-i\omega t - \varphi}$$

Here I have made use of complex numbers by having the real part be the x component and the imaginary part be the y component. Note that the phase difference φ in this equation is different than the one in the linear equation. In this case, the light is linearly polarized when $E_{0L} = E_{0R}$, and φ determines the angle of its polarization. It is of course circularly polarized if either E_{0L} or E_{0R} is zero, and is elliptically polarized when they are not equal to each other, but neither is zero.

Representing the polarization state in this way has certain benefits. It can be easier for calculations in certain problems. But it is also has a good physical explanation. In the quantum-mechanical description, an electromagnetic wave transfers energy in quantized packets called photons. Each photon has an angular momentum which is either -ħ corresponding to right-handedness, or ħ corresponding to left-handedness. Thus light consists of a stream of photons each of which is measured in a left- or right-handed state, and the polarization state of the light is dependent on how many end up being measured as left-handed, and how many are measured as right-handed. I say *measured as* left- or right-handed since before they are observed they are in a spin state where they have a certain probability of being left- or right-handed. For instance, linearly polarized light corresponds to the case where each photon has a 50-50 chance of being observed as left- or right-handed. If each photon does not occupy both spin states with the same probability, one angular momentum will be found to occur more often than the other, and with many photons the net result will be elliptically or circularly polarized light.

III. Implementation

I implemented the program in Python, and made use of the matplotlib module. It works in both Python 2.7 and Python 3. The window is divided into three major parts. The most important is a plot showing the x-y plane. The electric field vector's variation in time is shown on this plot, as well as the two circular components that add up to make it. The electric field is shown as a red line, and each circular component is shown as a blue line. The path of each circular component is a dashed blue circle, and the path of the electric field is either a red dashed line, ellipse, or circle based on its polarization state. Yellow dashed lines demonstrate the vector addition by the parallelogram method.

To the side of the plot is some explanation, showing the equations that are being depicted by the visualization. Three sliders are at the bottom of the window, allowing the user to interactively change E_{0x}, E_{0y} and φ. Figure 1 shows a picture of the program.

The program uses a matplotlib animation. The values for the endpoints of the vectors at every moment in time are calculated and put in numpy arrays. Each array holds either the x or y values for the endpoint of one of the three vectors that are shown. Each value in an array corresponds to a different moment in time. The animation function increments a global variable i, and then sets all the lines on screen to be based on the value at the i-th index of the arrays. The sliders are implemented as a matplotlib Slider widget. On its update function the arrays are recalculated and subsequently replotted in the next frame. Since i is a global variable, the animation will continue at the i-th position instead of restarting, allowing the animation to change smoothly as the sliders are changed.

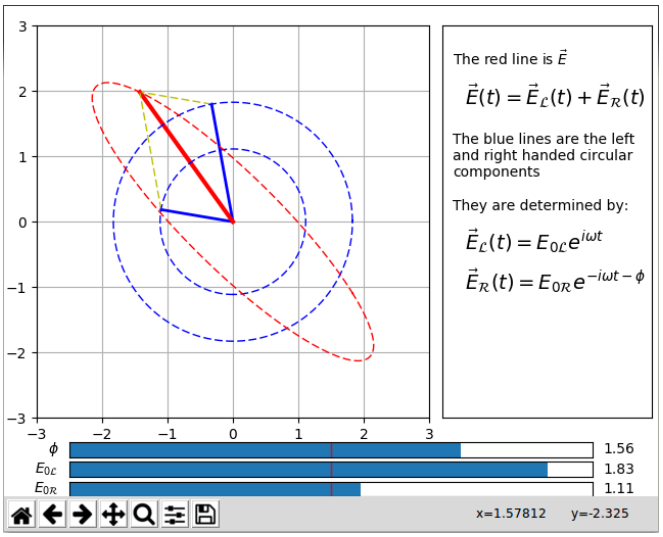

Figure 1 The program set to display elliptically polarized light

IV. Design Process

At the beginning the final result presented above is what I had in mind for this project, but in order to make it there were a few intermediate phases. The first step was to simply get the animation going. I got just the animation going with no way change the parameters except in the code, just to see if I could implement the animation. The next step was to provide some way to easily set the parameters. The intended audience for this program is physics students, and I did not want to force the user to modify code to use the program. My first interface for changing the parameters was to provide a simple GUI that would ask for the parameters, then start the visualization based on those parameters. Figure 2 shows this basic interface.

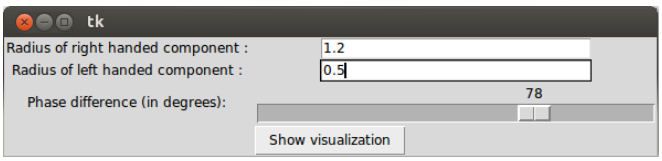

Figure 2 Basic GUI for launching an animation

While this was better than having to modify the code, this solution still wasn't satisfactory. Once the animation was launched, it could not be changed. Changing the parameters and clicking "Show visualization" again would simply launch a new pyplot with a different animation based on the changed parameters. This isn't really interactive; it is simply an easier way of producing static visualizations.

To further improve the program I decided to see if I could add a way to change the values interactively in the plot. The solution turned out to be the Slider widget from the matplotlib.widgets module. The widget can be placed in the pyplot, and it will call an update function when the slider is changed. One of the principal challenges was making the update function work properly. It has to recalculate the data, then make sure that the animation recognizes that the data has been changed. The way I did this was by having the data arrays declared to be global variables. When the update function changes the values in them, it resets the animation's frame sequence by using the method `new_frame_seq()`.

At this point the program was mostly complete, but there was still a subtle but significant problem. The animation would restart every time a slider was changed. This made using the sliders feel much less smooth and natural. At the time the animation worked by using the input argument of the animate function to determine at which index of the arrays to get the data from. When a new frame sequence was generated it would restart at i=0. The solution to this problem was to ignore the input argument and instead have the index be determined by a global variable. Each call to animate would increment the global variable. This way when a new frame sequence is generated the global variable stays the same, so the animation starts where it had left off.

V. Peer Feedback

Partway through the design process I showed this program to my peers in the course Interactive Visualization, to get their feedback. This is not my target audience, but it was still worthwhile to ask them about the usability of the program and how much supportive information should be included with the tool. My peers gave me some good ideas of ways I could tweak and improve the program, but said that fundamentally that it was already very usable. One of my classmates described it as fun to play with, and another said he had learned a lot of physics from it. Here are some of the specific points of feedback that I received, and my responses:

There should be a way to enter precise values for the different parameters, like a text box or keyboard controls for the sliders.

This would be an interesting addition, but ultimately I decided not to include it. I did add a text entry field, but I found that the clutter it added was not necessary. The purpose of this tool is more centered around seeing how changing values affects the polarization state, and setting the parameters to precise values is not of the greatest importance.

The sliders are thin. It would be easier to use if they were bigger.

I increased the width of the sliders in response to this.

There should be a way to pause the animation, and rewind or fast forward to particular points.

I think this could be a good addition, but did not end up implementing it due to time constraints.

There should be tool tips or some sort of UI to explain what the sliders do.

This is of course important, and hadn't been implemented yet when I was showing the program to my peers. This is the purpose of the pane in the right side of the window that shows the equations used.

The units should be displayed.

I find that this is not too much of a concern. The units for the radii are arbitrary, and for φ I used radians, which in physics is standard. Based on the way the equations are written, φ should be in radians.

The animation restarts whenever a slider is changed. This is annoying.

I was aware of this problem, but hadn't figured out how to fix it yet. This feedback confirmed that it was a noticeable and irritating problem, and that I should prioritize finding a way to fix it. This was one of the first issues I addressed after receiving feedback.

When I asked about how much supplementary information to include I was advised to avoid going too in depth and to just give a general overview. My plan at that point was to try to embed this visualization into a larger application which would have a large text explanation. This could have taken the form of a web page or a Jupyter notebook. Firstly, this did not turn out to be easy to do, and after thinking about the intended use I eventually decided that it wasn't necessary. The intent of this tool is to be a supplementary aid to optics students. The people who would likely want to use this tool should have notes and their textbook to explain the background. This program would be to visualize what they are learning in class, is not intended to teach the material all by itself. So the only explanation that is necessary is to show the form of the equations that it is displaying.

VI. Discussion and Possible Extensions

I would see this tool being used as an aid in the teaching of an optics course. After explaining how polarized light can be represented as the superposition of a left- and right-circularly polarized states a professor could use it as a demonstration of the concept. Students could use the program and be asked to answer questions about polarization as a hands-on activity. Seeing how interactive engagement is so effective in improving comprehension this activity could be a good way to improve understanding about polarized light. A good future work would be to perform a study examining how much this program improves comprehension.

There are other implementation details that it would be interesting to try. One of my peers mentioned the idea of having the origin of one of the components moved to the endpoint of the other component, so that instead of spinning around the origin it spins around the endpoint of the other component. This would essentially being showing the vector addition of the two components with the triangle method

rather than the parallelogram method. I have implemented this, and have found that it also produces interesting results. Another future work could compare this visualization to the one I have presented here.

This tool could also be extended to demonstrate phasors in general. A phasor is a concept used to help think about any system where there are many periodic components that add up, and basically represents the value of a sinusoidal function with a complex exponential. With some modifications this program could be modified to show many phasors with different frequencies. Such a program could be useful for a wide range of subjects such as electrical circuit design.

Citations

[1] I. Hallouri, D. Hestenes, *The Initial knowledge state of college physics students.* Am. J. Phys. **53** (11), Nov. 1985. 1043-1055

[2] R. Hake, *Interactive-engagement versus traditional methods: A six-thousand-student survey of mechanics test data for introductory physics courses.* Am. J. Phys. **66** (1), Jan. 1998. 64-74

[3] *Math, Physics, and Engineering Applets.* n.p. n.d. http://falstad.com/mathphysics.html

[4] E. Hecht, *Optics.* Addison Wesley, San Francisco, 2002