Determination of g Using a SeeLab-Based Simple Pendulum Experiment

Aritra Mukhopadhyay
National Institute of Science Education and Research
Bhubaneswar, Odisha 751005, India
3rd year, Integrated M.Sc. Physics
Roll No.: 2011030
(Dated: March 31, 2023)

We present an inexpensive experiment to measure the gravitational acceleration (g) using a Seelab microcontroller-based system. The experiment uses a basic harmonic oscillating pendulum and an LED-photo transistor setup to determine the time period of the oscillation. Readings are automatically taken, and the constants are computed using Python code.

Lorem: The first part of the experiment is conducted in a lab frame, where the pendulum oscillates under the influence of gravity. The time period of the pendulum is measured using the LED-photo transistor assembly, and the value of g is calculated.

Ipsum Dolor: In the second part of the experiment, the pendulum experiment is repeated in an elevator frame to determine the acceleration of the lift. By comparing the period of the pendulum in the lab frame to the period in the elevator frame, the acceleration due to the lift's motion is calculated. This extension provides a way to measure gravity in a non-inertial reference frame.

I. THEORY

A. About the experiment

The experimental setup involves a rod pivoted at one end with an LED and a photo-transistor mounted on opposite sides below the pivot point, near the end of the rod (Illustrated in Fig 1). When the rod oscillates through the middle, it blocks the light from the LED to the photo-transistor, generating a pulse. The photo-transistor, being an NPN transistor, gives a spike of current in response to the change in light intensity.

The circuit used in the experiment involves connecting the ground of the LED and the emitter of the photo-transistor to the ground of the Seelab device. The other end of the LED is connected to the PV1 input of the Seelab, while the other end of the photo-transistor is connected to the SEN1 input. Another wire is connected from the SEN1 input to the A1 input of the Seelab. The circuit is shown in Fig 2.

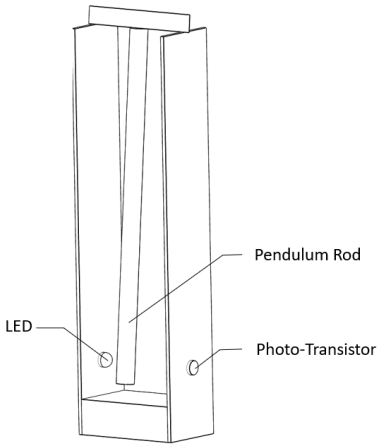

FIG. 1: The experimental setup

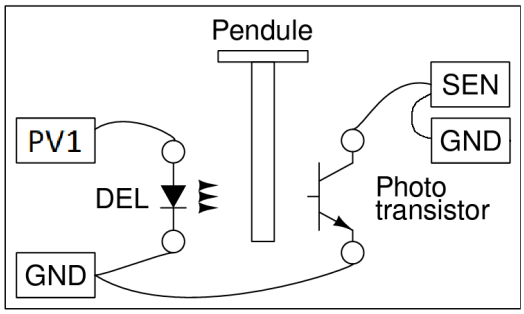

FIG. 2: The Circuit Used

By capturing the pulses generated by the photo-transistor using the Seelab device, the time period of the oscillations of the pendulum can be measured accurately. This data can then be used to calculate the values of g and G, as explained in the theory section. The circuit setup is simple and allows for precise measurements to be made, ensuring accurate results from the experiment.

B. Calculating the value of gravitational acceleration (g)

The time period of an oscillating rod of length l is given by:

$$T = 2\pi\sqrt{\frac{l}{g}}$$

For a complex pendulum, the effective length of the pendulum is given by $l_{eff} = \frac{2}{3}l$, where l is the length of the pendulum from the point of suspension to the center

of mass. Substituting this value in the above equation, we get:

$$T = 2\pi\sqrt{\frac{2l}{3g}}$$

Solving for g, we get:

$$g = \frac{8\pi^2 l}{3T^2} \tag{1}$$

C. The Data

We had an NPN photo-transistor. It is an analog output device (output ranging in 0 to 5V) and is characterizes by giving an `HIGH`er value in shadow and `LOW`er value in light.

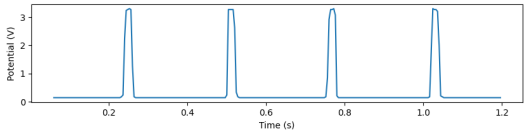

FIG. 3: How the graph looked (zoomed in time)

In Figure 3, we can see how a the data plotten on a graph looks like. In x-axis we plot the progression of time, and in y-axis we plot the value output given by the photo-transistor. In the graph we can see a some spikes. Those are created when the pendulum passes through the mean position and casts a shadow on the photo-transistor.

II. WORKING WITH THE DATA AND CALCULATING THE TIME PEIOD

A. Finding Time Period in Lab frame

We got the data in the form of two arrays, one with the time (in milliseconds) and the other with the value of the photo-transistor in volts (0 to 5V). We thought it would be easier for us to work with digital data. So we changed the array to only `True` and `False` values with a threshold of 3V.

We decided to call it a peak when we see a `True` value followed by a `False` value (i.e: we are detecting when it is going down every time any other point like when it is going up could have worked as well). We first find the first peak, and store the time to a variable t_0. Then we move along the graph, and when we found the next peak, we found the difference between the time of this peak and t_0. Thus we got the distance between these consecutive peaks.

Next we store the time of the current peak to t_0 and move forward to find the next peak. We repeat this process until we find all the peaks and store the time difference between consecutive peaks in an array.

Problem Faced: We found that, the time difference between consecutive peaks was not constant always. Two peaks sometimes have a tendency to come closer. We need to remember that the pendulum crosses the mean position twice in one oscillation. So, if the detector is not present exactly in the mean position, we will have unequal angles to swing on the left and the right. So, although the time difference between every odd^{th} peak and every $even^{th}$ peak is same, the time between two consecutive peaks don't remain same.

To avoid this issue, we decided to consider only even number of differences into consideration. Thus, if we took $2n$ differences, we will get n oscillation data (apart from two of the extreme points, in one oscillation a pendulum crosses every other point in the path twice). This was achieved by removing the last element of the list if the length of the list was odd.

Now the array was in milliseconds, and contained distances between consecutive peaks. So, we took the average of the array and multipled it by $(1000/2) = 500$. This gave us the average time period of the oscillation.

```python
def times_between_peaks(t, val):
  val = val > 3  # converting to binary values

  i = 0 # index of t
  got_first = False
  ts = []
  while i < len(val)-1:
    # finding the first peak is important
    if not got_first:
      if val[i] == True and val[i+1] == False:
        t0 = t[i]
        got_first = True  # first peak found
    else:
      if val[i] == True and val[i+1] == False:
        ts.append(t[i] - t0)
        t0 = t[i]
    i += 1

  # removing the last peak if oscillations are
    not complete
  if len(ts)%2: ts.pop()  # if len(ts) is odd
  return np.array(ts)  # in milliseconds

# CALLING THE FUNCTION: we had taken the data
    before in variables t and val
times = times_between_peaks(t, val)
T = times.mean()/500  # time period in seconds
```

Listing 1: Python code to find the time period of the oscillation.

B. Finding acceleration of lift

While calculating the acceleration in lift, we found that the acceleration was not uniform. So, we decided to take the maximum acceleration recorded from all the 6 trials

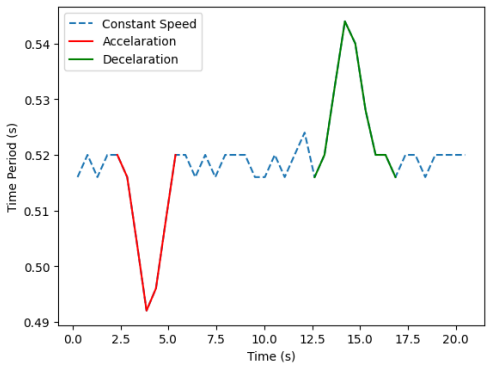

FIG. 4: Graph of the time period variations in lift

and to take the average. **In Figure 4 we can see one of the 6 data points we recorded (one data point corresponds to whole data of going up to 4th floor from Ground floor in one of the SPS lifts)**

III. OBSERVATION AND CALCULATION

A. In Lab Frame

We took the oscillation data for 30 seconds. Then using the above mentioned `time_between_peaks()` function, and multiplying it by 500 (explained earlier) we get the average Time Period of oscillation in seconds. We then calculated the acceleration of the lift using the Equation 1.

From the code, we got the Time Period of the pendulum as: **0.5185s**. Thus from the Equation 1, we got the acceleration of the lift as: $\mathbf{9.79 m/s^2}$.

B. In Lift Frame

In lift, we observed that the SPS lift takes around 15 seconds to go from the ground floor to the 4th floor. So, we took reading for 20-25 seconds while the lift was moving. Then we found out the time periods between the peaks and added every odd and even distances. Thus we got an array of time periods of the pendulum at different times. We got Figure 4 by plotting the time periods with time. next we labelled the peaks in the graph manually.

We take the extreme points in those peaks and intend to find the maximum acceleration and decelaration of the lift.

From the graphs[1] we found that, **Average Maximum Acceleration Time Period** is $0.492s$ and **Average Maximum Deceleration Time Period:** $0.544s$.

The length of the pendulum rod is measured to be $10cm$

Now using Equation 1 we get, while the lift is accelerating, the acceleration of the lift is $\mathbf{10.873\ m/s^2}$ and while the lift is decelerating, the acceleration of the lift is $\mathbf{8.893 m/s^2}$.

Now, the original acceleration of the lift is $\mathbf{9.79 m/s^2}$. So, the maximum acceleration of the lift with respect to the earth is $\mathbf{1.083 m/s^2}$. Similarly, the maximum deceleration of the lift with respect to the earth is $\mathbf{0.896 m/s^2}$.

IV. ERROR ANALYSIS

- **Alignment of Pendulum:** Even though the pendulum was oscillating on a knife edge, there could have been some misalignment in the pivot point which could have introduced errors in the measurement of the time period. To minimize this error, we made sure to align the pendulum carefully before taking measurements.
- **Peak Detection:** The pendulum rod was thick, so we had to take the falling curve (when the pendulum is going out) while detecting a peak. We should have taken the middle point in the peak to evade errors. Due to this, there could have been some error in the measurement of the time period of the pendulum.
- **Brief Lift Acceleration:** During the second part of the experiment, we conducted the same experiment inside a lift to find the acceleration of the lift. However, the lift accelerated and decelerated for a very small amount of time, and we could not take many data points during this short period. This limitation could have affected the accuracy of our measurements.
- **Lift Vibration and Jerk:** In the lift, there could also have been some vibration or jerk due to the lift's motion, which could have affected the motion of the pendulum and introduced error in the measurement of the time period.
- **Effect of Air Resistance:** Although the pendulum rod was small, there could still have been some effect of air resistance on the motion of the pendulum. This effect would be negligible but not completely negligible. To reduce this error, we made sure to keep the air around the pendulum as still as possible during the experiment.

V. CONCLUSION

In conclusion, the experiment allowed us to understand the theory behind the calculation of the gravitational acceleration (g) using a complex pendulum. By using SeeLab, we were able to automate the data collection process, making it easy and efficient to obtain accurate measurements.

We learned that SeeLab is a powerful device that enables hardware-software integration and allows for coding in Python. It has multiple input and output pins that can be utilized for various types of experiments.

The second part of the experiment, conducted in the lift, allowed us to measure the acceleration of the lift, highlighting the potential applications of this technique in other areas.

Overall, the experiment demonstrated the utility of SeeLab and the benefits of combining hardware and software to obtain accurate measurements. It provided a valuable learning experience and insight into the world of experimental physics.

---

[1] Note that in Figure 4 for acceleration time period is less and for decelaration time period is more.

[2] P. P. I.-U. A. C. A. R. C. of UGC), Experiments for young engineers and scientists, (2017), `https://csparkresearch.in/assets/pdfs/expeyes17.pdf`.

[3] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, Array programming with NumPy, Nature **585**, 357 (2020).

[4] J. D. Hunter, Matplotlib: A 2d graphics environment, Computing in Science & Engineering **9**, 90 (2007).