Lab 2: Q Factor

Robert Purcaru

October 27, 2020

Contents

1 Experimental Design

The pendulum used in this experiment was constructed by fixing an adjustable G clamp to the top of a doorframe and suspending the mass from the throat of the clamp using a length of fishing line tied in fixed loop (bowline) at one end and attached to the weight at the other (Figure 1). This ensured a large range along which the length of the string could be adjusted and a wide field of view for the camera to capture. To prevent damage to the doorframe and allow for an increased clamping force, a pair of books were used between the clamp and the door frame. The clamp was tightened as much as was possible by hand to minimize any unwanted movement in the setup. The string length can be varied by using a shorter or longer piece of fishing line attached to the mass and the clamp in the same way.

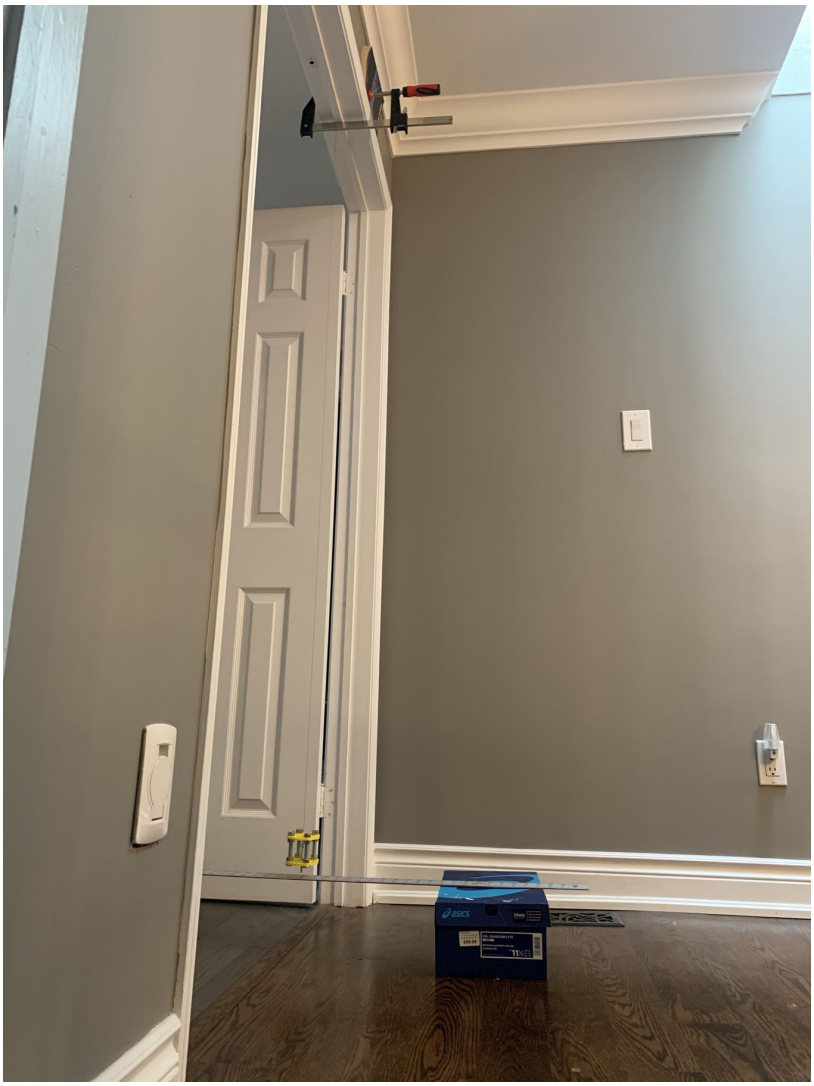

Figure 1: Experimental setup used, excluding tripod and recording device. Left half of setup is obscured by wall.

The mass used was composed of four 3/8-inch x 2-1/2-inch steel carriage bolts attached to a pair of 3D-Printed plates, one of which received an additional 3/16-inch x 1-1/2-inch steel eyebolt used to attach the mass to the end of the string. This design was chosen because it allowed for the location the string is mounted to to be varied continuously along the axis of the eyebolt, meaning it can be adjusted so that the string is closer to the center of mass for any number of carriage bolts. The mass can be adjusted by either removing or adding carriage bolts in pairs of two, giving four different configurations of the mass (Figure 2). Four carriage bolts were used for this experiment, despite the capacity for eight, because only four were available at the time the experiment was conducted. If more variation in the

mass is required, larger plates can be printed to accept more carriage bolts or additional weights can be fixed to the part of the eyebolt the extends beyond the lower plate. The fishing line used was chosen because of its high tensile strength to weight ratio.

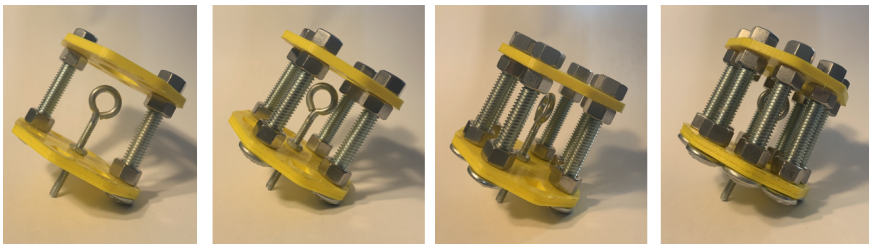

Figure 2: All combinations of bolts used to vary mass available with plates that were used for this experiment. The mass with four carriage bolts (second from the right) was used in this experiment.

Before the mass was attached to the string, the position of the eyebolt was adjusted until the mass could be balanced side ways on the side of a pencil. However, once the mass was attached to the string, this configuration proved very unstable; the mass would rock about the point the string was attached to violently and irregularly. To address this, the eyebolt was raised by about 6 ± 0.5 threads on the eyebolt, corresponding to 0.50 ± 0.02 inches, or 1.27 ± 0.05cm. The string was then attached before its length was measured several minutes later to allow the pendulum to stabilize and to account for any strain in the string lengthening it. The length of the string was measured using a soft tape measure and found to be 184.5 ± 0.1cm between the top of the clamp and the nearest end of the interior of the loop of the eyebolt. However, since the eyebolt was raised by 1.27 ± 0.05cm, the distance between the center of mass and the pivot in the neutral position can be calculated as $184.5 \pm 0.1 + \frac{1.27 \pm 0.05}{2} = 185.1 \pm 0.1$cm, assuming the change in position of the center of mass is negligible from moving the eyebolt.

The masses of the mass and string were measured using a kitchen scale, giving 269.8 ± 0.1 grams for the mass and 0.2 ± 0.1 grams for the mass of the string. Since the mass of the string is three orders of magnitude smaller than the mass of the mass, we can consider it negligible for the purposes of this experiment. It was also found that the grooves cut into the throat of the G clamp were sufficient to prevent excessive lateral movement of the string while the experiment was being conducted (Figure 3).

Figure 3: View of string hanging between teeth on clamp.

Finally, a measuring stick was placed underneath the pendulum and aligned with the eyebolt so that the 0 centerline marker lined up with the neutral axis of the pendulum (Figure 4). The video was taken from a top down view at a resolution of 1080x1920 pixels, where the pendulum swings along the axis of the 1920 pixels.

Figure 4: Side view of mass zeroed against ruler.

2 Data

To collect data, a video was recorded by mounting an iPhone X to a tripod and placing it close to the string approximately, 1.3 meters above the mass, giving a top down view of the pendulum. This video was then analyzed in python using the OpenCV library [1]. The program used looks at every frame in the video for the specified color, approximates the shape formed by the color to a circle, and reports the pixel position of the center of the circle. (Figure 5).

Figure 5: Screenshot of thresh (left) and contour (right) views provided by OpenCV during video. The thresh view shows which areas of the frame are within the specified colour threshold determined in advance by identifying the RGB values of the yellow mass. The contour shows the circular approximation made by OpenCV in green and the center of the circle in blue. The position in pixels from the top of the frame is recorded every time the mass can be identified.

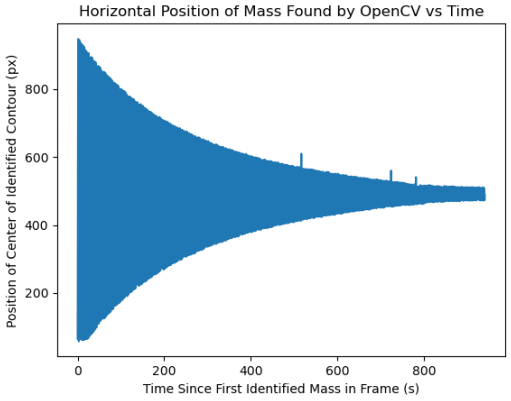

Figure 6: Raw data as mapped by OpenCV at specified frame. Video was recorded at 29.97 frames per second. Three distinct outliers can be seen around 520 seconds, 730 seconds, and 790 seconds. These occur because a foot entered the frame at those times, causing a tracking error in OpenCV. Error bars are excluded from this graph because data from OpenCV is interpreted with uncertainties in Figure 7; this graph only depicts information developed by OpenCV.

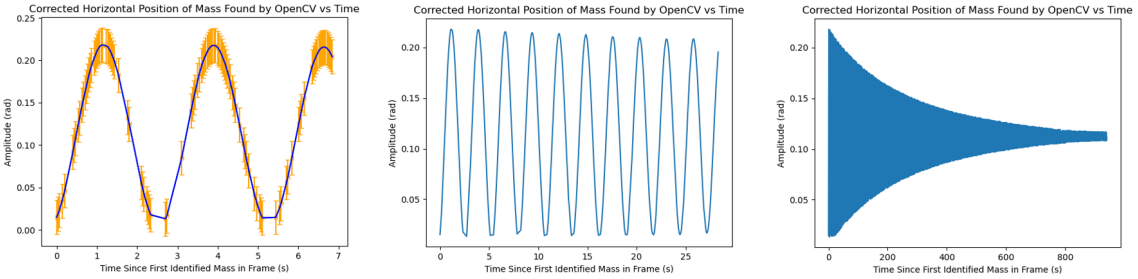

Figure 7: Data collected at specified frame, correcting for outliers and tracking errors after approximately N cycles. The graph featuring 3.5 cycles (left) depicts vertical error bars marking an uncertainty of$\pm$0.02 radians (see 3.2) and horizontal error bars marking $\pm$0.02 seconds (half of a frame). Graphs for 10.5 and 352 cycles (center and right) are shown to better illustrate the decay of the amplitude of the pendulum. It is worth noting that during the first few cycles the mass goes out of frame at its maximum amplitude on one side. This can be seen in the apparent sharp edges in the early troughs of the graph depicting 10.5 cycle and the 'blunt' edge of the bottom left side of the graph depicting 352 cycles.

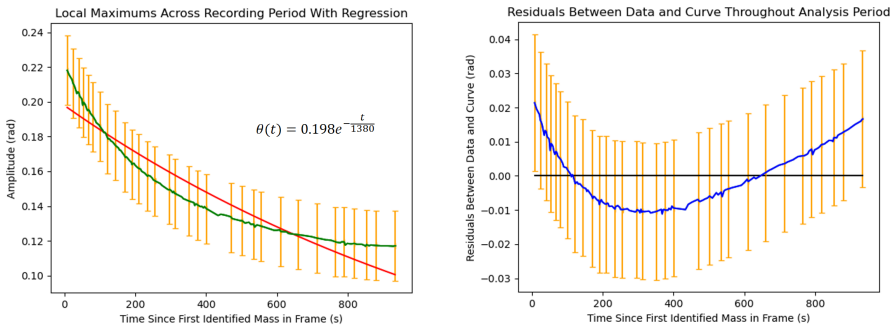

Figure 8: Graph depicting relative maximums at specified times as determined by Open CV (green) with overlaid regression (red) in the form $\theta(t) = \theta_o e^{-t/\tau}$, where $a = 0.198$ and $\tau = 1380$ (left). The graph depicting the residuals (blue) between the data and the regression (right) illustrates some significant discrepancy between the model and the measurement (a black line at 0 rad discrepancy is used to help illustrate this). This is further discussed in 3.2. Both graphs feature error bars (orange) every 5 data points, illustrating a measurement error of 0.02 radians in the amplitude. Note that there are some large gaps between error bars due to problems determining a local maximum for certain times as a result of measurement errors.

3 Sources of Error

3.1 Measurement Errors

The pendulum and its amplitude was supervised throughout the experiment. The exposed portion of the eyebolt was used as a reference to find the position of the pendulum against the ruler underneath the pendulum. The ruler was zeroed within 0.0625 inches (Figure 4) so that any error resulting from an offset as the ruler was read could be minimized. Still, the measurement error here is significant. Given that the ruler was being read while the weight was moving, an uncertainty of about $\pm$ 0.3 inches seems reasonable, as this gives 4 increments in both direction to account for parallax and reading error at the start of the experiment. As the experiment proceeded, and the pendulum slowed down, it became easier to read the ruler, thereby increasing the confidence of the measurement and decreasing the error to about $\pm$0.1 inches, or about 2 increments on the ruler in both directions. This corresponds to 6 pixels, using Equation 6.

3.2 Analysis Errors

The OpenCV library doesn't explicitly provide any information for uncertainty or accuracy of the values provided. However, from experience, OpenCV provides the center of a

relatively good circle to within about a pixel. Since the circle identified was broken by the bolts (Figure 5), the error is likely around 8 pixels at the start when the pendulum is moving at its fastest, and decreasing as it slows down, resulting from motion blur. Using Equation 6, we obtain a value of ± 0.02 rad for our uncertainty in the reported amplitude of the pendulum. As the pendulum slows and motion blur become less significant, this value likely decreases, however we will use ± 0.02 radians as it is the largest source of uncertainty we find. This is a significant amount and should be addressed in future experiments. The measurements can be made more accurate and more frequently if OpenCV can identify the mass more consistently. To do this, a cover that is an unbroken color should be made to obscure the nuts on the top of the mass. This will allow the OpenCV to develop a better approximation of the pendulum, thereby reducing the error. It would also be worth trying a view perpendicular to the axis of rotation ('in front' of the pendulum) as this would largely eliminate measurement errors due to parallax.

4 Q Factor

4.1 Counting Cycles

By watching the first cycle and later reviewing the video manually, it was found the initial amplitude was 13.75 ± 0.25 inches, as read on the ruler bellow. This corresponds to about 905 ± 15 pixels of the video and 0.22 ± 0.02 rad. If we consider the parallax error in reading the ruler negligible, we find that the amplitude corresponding to $e^{-}\pi$ of 14.0 in is about 0.605 ± 0.1 inches. The This corresponds to about 41 ± 1 pixels. Counting in real time, I found that the amplitude reached 0.605 inches after 328 complete oscillations. Reading the raw data, we find that the amplitude reaches 4% of its original amplitude after 332 cycles, corresponding to 15 minutes and 5 seconds of video time. While these two values are very close, the uncertainty of reading the ruler is greater than the uncertainty using OpenCV; the ruler used only has marks every 0.0625 inches, which must be judged to line up with the center of the weight in real time. The data generated by the program on the other hand has an estimated error of 8 pixels, which likely decreases as the pendulum speed decreases. We will therefore consider 332 as the Q factor determined by this method.

4.2 Determining τ

To affirm our value of Q obtained by counting, we can consider the formula

$$Q = \pi \frac{\tau}{T} \tag{1}$$

where T is the period of one complete oscillation and τ is the time constant of decay. To find the period, we can take the average of the differences of neighboring maximums to get 2.73s. To find the error in this measurement, we can take the standard deviation of the difference of consecutive maximums, giving us 0.07s. Therefore, the period of one oscillation

is 2.73 ± 0.07 seconds. To find a value for τ, we can use the formula

$$\theta(t) = \theta_o e^{-t/\tau} cos(2\pi \frac{t}{T} + \phi_o) \tag{2}$$

where $\theta(t)$ is the amplitude in radians as a function of time, θ_o is the initial amplitude in radians, and ϕ_o is the phase constant that accounts for the difference between when the analysis starts and the pendulum starts moving. If we only consider the maxima of our data, $cos(2\pi \frac{t}{T} + \phi_o) = 1$, so we are left with

$$\theta(t) = \theta_o e^{-t/\tau} \tag{3}$$

The maximums calculated and the line of best fit approximated using Equation 3 are shown in Figure 8. The value for τ used in the approximation is 1380 ± 40 seconds. This is provided by the square root of the first element in the covariance matrix give by the curve fit function in the SciPy library [2]. However, it is clear from the graph that this approximation is very rough. Keeping that in mind, the value we calculate for using this method is 1590 ± 40 (Equation 5 in 5.1). This value may be excessively large as a result as the relatively small original amplitude; about 0.218 radians (about 12.5 degrees). A small amplitude like this leads to a decreased maximum speed and, assuming air resistance, given by the equation

$$D = Cd\rho \frac{v^2 A}{2} \tag{4}$$

is the largest slowing factor, the pendulum would be impacted less by air resistance at slower speeds (v). However, this may also be a result of an analysis error. Future experiments will attempt to reduce the uncertainty resulting from analysing the data using methods described in 3.2.

5 Appendix

5.1 Calculations

Approximation for Q using τ obtained in Figure 8:

$$Q = \pi \frac{2.73 \pm 0.07}{1380 \pm 35} = 1590 \pm 40 \tag{5}$$

Conversion factors used to determine angle from distance. P is a position in pixels:

$$\theta(P) = \arcsin \frac{\frac{2.54cm}{in} \frac{31.875 \pm 0.25in}{1920px} P}{185.1 \pm 0.1cm}, \tag{6}$$

5.2 Programs Used

The following is an excerpt of the program that tracks the mass. The program looks for the colors specified in the given frame, approximates the region the color is identified in with a circle and records the center of the identified circle.

```
def find_mass(frame, show_trackbars = False):
    frame = cv2.GaussianBlur(frame, (7, 7), 0)
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    hul, huh, sal, sah, val, vah = 23, 40, 126, 255, 127, 255

    if show_trackbars:
        hul=cv2.getTrackbarPos(hl, wnd)
        huh=cv2.getTrackbarPos(hh, wnd)
        sal=cv2.getTrackbarPos(sl, wnd)
        sah=cv2.getTrackbarPos(sh, wnd)
        val=cv2.getTrackbarPos(vl, wnd)
        vah=cv2.getTrackbarPos(vh, wnd)

    HSVLOW = np.array([hul, sal, val])
    HSVHIGH = np.array([huh, sah, vah])
    thresh = cv2.inRange(hsv, HSVLOW, HSVHIGH)
    floodfill = thresh.copy()
    h, w = thresh.shape[:2]
    mask = np.zeros((h+2, w+2), np.uint8)
    cv2.floodFill(floodfill, mask, (0,0), 255)
    floodfill_inv = cv2.bitwise_not(floodfill)
    thresh_filled = thresh | floodfill_inv
    edged = cv2.Canny(thresh_filled, 30, 200)
    all_contours, hierarchy = cv2.findContours(edged,
    cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

    filtered_contours = []

    for contour in all_contours:
        area = cv2.contourArea(contour)
        x, y, w, h = cv2.boundingRect(contour)
        if abs(max(w, h)/min(w, h)) < 1.5:
            if area > 100:
                filtered_contours.append(contour)
                break
```

```
if len(filtered_contours) < 1:
    return -1, -1, -1, thresh_filled

mass_contour = filtered_contours[0]
center, radius = cv2.minEnclosingCircle(mass_contour)

return int(center[0]), int(center[1]), int(radius),
thresh_filled
```

References

[1] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[2] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.