Colliding Pucks on a Carom Board*

Muhammad Umar Hassan, Bilal Aftab Usman and Muhammad Sabieh Anwar
Syed Babar Ali School of Science and Engineering, LUMS

V. 2017-1; May 2, 2017

Collisions are ubiquitous in the nature. They occur at microscopic levels between elementary particles as well as at grander scales when galaxies collide, merge and scatter. In all such processes, the linear momentum of the complete system is always conserved. In this experiment, we will study the collision of two objects in a controlled setup using the video tracking technique. We will show that the law of conservation of momentum holds in 2-D collisions and determine the type of collision through quantitative analysis.

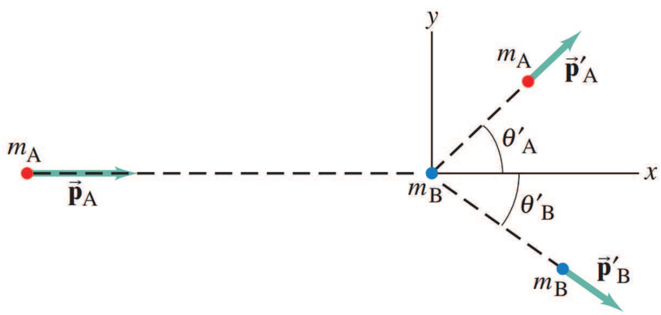
Essential pre-lab reading: "*Physics for Scientists and Engineers with Modern Physics; 3rd Edition*" by Fishbane, Gasiorowicz and Thornton; (Sections 8.1-8.4)

1 Test your understanding

1. What is the difference between an elastic and an inelastic collision?
2. Referring to Figure 1a, write down two scalar equations which must be satisfied for the illustrated collision.
3. How can you evaluate the angles θ'_A and θ'_B?
4. How can you measure the velocities of the pucks at the time of collision?

2 The Experiment

The collision between two carom pucks will be recorded using a high-speed video camera. We will use markers of contrasting colors on each puck to track their positions and rotation separately. The subsequent data acquired will then be used for quantitative analysis using PhysLab's video tracking library called "PhysTrack". It is suggested that the experimenter peruses our primer on video tracking to fully utilize the library.



(a) Illustration of the collision.

(b) The camera is fixed on the top and is being aligned using a `Bull's eye level`.

(c) The apparatus used for the experiment showing a carom board, tripod stand and a camera.

(d) Marking the pucks

Figure 1: Experimental set-up used for observing the collision.

For this experiment, we have constructed a special tripod as shown in Figure 1c. Set up the camera to use high-speed video at $240 fps$ and fix it on the top of the tripod stand as shown in Figure 1b. Ensure that the camera is placed horizontal using a `bull's eye level` or a `spirit-level`. Mark two points on each puck with contrasting colors. See figure 1d as an example.

Q 1. Measure the mass and diameter of each puck and record it in your lab notebook. Take a few readings and calculate the mean.

Now, place one puck on the carom board such that it appears on the center of the camera's viewfinder. Use the camera's optical zoom to fill the scene with the entire carom board. Now, place the other puck just inside the zoomed frame and start recording. Target the puck placed at the center with the other one. Stop the recording when both of the pucks stop. Repeat as many times as necessary. A single video should be enough for quantitative analysis but you can obtain as many samples as you want. Measure the diameter of the central ring printed on the carom board. We will use this value later for calibration.

3 Analysis

3.1 Preparing the Data for Quantitative Analysis

Copy the videos to some appropriate location in the hard-drive. Download and extract on hard-drive the latest release of PhysTrack from our website. In Matlab, browse to the extracted PhysTrack directory where you should see the "+PhysTrack" folder and some "analyze motion scripts" in the "Current Folder" window. Run the script `analyze2DCollision` which presents a series of GUI tools to setup and perform video tracking. When asked by the interface, browse and select your video file. The script will automatically show a tool to trim and crop the video. Use the `slider` and the `go-to buttons` to seek different frames of the video. Mark as `In-Frame` the moment when the first puck has just left the striking finger and as `Out-Frame` the moment you want to stop tracking the motion after the collision. Preview the trimmed video if necessary and close the tool afterwards.

For quantitative analysis, the striking puck will be referred as "A" and struck puck as "B".

The script will now present a coordinate system tool. Follow the on-screen instructions to setup a coordinate system anywhere on the carom board. Use the `Reset Origin`, `Reset x-axis` and `Toggle Direction` buttons to move the origin, rotate the x-axis about the origin and toggle the y-axis direction respectively. Enter the value of the known distance marked on the horizontal surface in text-box labelled `Marked Distance` and click on the `Reset Unit Marking` button to draw a line on the distance markers with known real distance marked on the surface, followed by the `enter` key. Repeat any of these steps as many time as needed to achieve the desired results. Once everything is final, close the tool.

In the object selection tool, the script assumes that you mark four track points in a specific sequence. To select each object, click "Manually mark and object" and draw a tight rectangle around every marker visible in the sample frame. See figure 2 which guides how to setup the motion tracker. First mark the two track points on puck A and then on puck B. If any marker needs to be remarked, just draw a new rectangle at the same place which will replace the previous marker. When you close the tool to start the tracking process, there should be exactly four object in the objects listbox.

When the tracking is finished the script will open up a track point filter tool to manually remove any stray points. Use the on-screen controls to seek different instances of the video and filter out the unwanted track points by drawing rectangles around them. Each step in this tool can be undone using the `Undo` button. At the end of a successful run, the script will leave some variables in the workspace. See Table 1 for details of these variables. The script will also prompt to draw two circles to identify the colliding bodies. Identify three distant points on the periphery of each puck to draw two circles. We call this process "stitching object to the trajectory". At the end of a successful run, the script will leave some variables in the workspace. See Table 1 for the details of these variables.

To fully understand the significance of these variables, the experimenter is suggested to also read

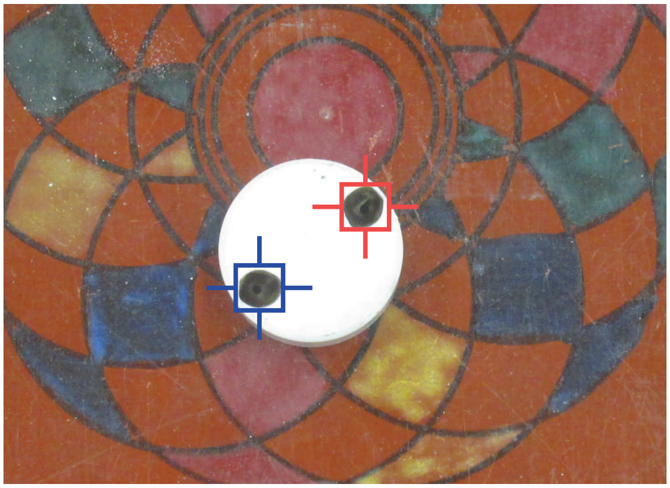


Figure 2: Setting up the point tracker. Note that the image has been artificially enhanced for visual clarity.

our Primer on Video Motion Tracking, that is uploaded on our website.

Physical Quantity	Variable Names
Trajectories of track markers on object a	*trajs.tp1, trajs.tp2*
Trajectories of track markers on object b	*trajs.tp3, trajs.tp4*
Trajectories of centroids of the puck a, b	*cenA, cenB*
Trajectory of a peripheral point on object a, b	*ppA, ppB*
Radii of objects a and b	*radA, radB*
Angular Displacement of object a and b	*dAngA, dAngB*
Time stamps	*t*
Index of Collision	*IoC*

Table 1: Base workspace variables generated by the script.

Notes about the extracted data:

- The variables `cenA` and `cenB` contain the trajectories of position vectors for the stitched circles. The position of a circular body is identified from the center of the circle. Hence, these variables contain the trajectories of the centroids of the pucks.
- Similarly, the trajectory of a fixed point on each puck is also calculated through the stitching process. These trajectories are stored in the variables `ppA` and `ppB`

When you obtain the results, plot the `ydata` against the `xdata` for both markers and try to identify the index of collision.

Q 2. How can you identify the index of collision using the position data? What is the index of collision according to your analysis?

Since the script has already left an estimate for the frame of collision in variable `IoC`, you need to decide if you are going to keep it or use your own value.

For puck A, sample indices `[1:IoC - 1]` represent $\vec{r}_A$ and `[IoC:end]` represent $\vec{r}'_A$. Similarly, for puck B, indices `[IoC + 1:end]` represent $\vec{r}'_B$. Extract these points from `objA` and `objB` and store in a separate variable.

3.2 Quantitative Analysis

We need to find the angles θ'_A and θ'_B according to Figure 1a. We also require the velocity $\vec{v}_A$ just before the collision and the velocities $\vec{v}'_A$ and $\vec{v}'_B$ just after the collision to test the law of conservation of momentum.

Let us first find the angles θ'_A and θ'_B.

Q 3. Plot the respective `ydata` and `xdata` data from `cenA` for $\vec{r}_A$. Fit this data to a linear function using the basic curve fitting tool. Save the results in appropriate variables. Repeat this procedure for fitting $\vec{r}'_A$ and $\vec{r}'_B$.

Q 4. Find the angles θ'_A and θ'_B using the fit results. Show the working to the lab instructor before proceeding.

$$d(t) = c_0 + c_1 t + c_2 t^2. \tag{1}$$

Get three separate datasets of displacement for the motion before and after the collision. Extract the datasets from variables `xA` and `xB` using the index of collision. See the following code which exemplifies this task.

```
dLinA_bCol = sqrt(...
        (cenA(1:IoC - 1, 1) - cenA(1, 1)).^2 + ...
        (cenA(1:IoC - 1, 2) - cenA(1, 2)).^2);
dLinA_aCol = sqrt(...
        (cenA(IoC:end, 1) - cenA(IoC, 1)).^2 + ...
        (cenA(IoC:end, 2) - cenA(IoC, 2)).^2);
```

We now need to fit the displacement data to a mathematical model. Since, we are only dealing with the kinematics, a suitable model function for displacement is the general quadratic function as shown in Equation (1).

Q 5. Why do you think that Equation (1) is a valid model for displacement? What do the coefficients c_1, c_2 and c_3 represent?

You will fit the extracted displacement datasets to the model using the least-squares fitting function `PhysTrack.lsqCFit`. The syntax for using this function is

```
fitResult = PhysTrack.lsqCFit(xdata, ydata, 'd', 'c0 + c1 * t_ + c2 * t_^2', 't_');
```

The variable **fitResult** is a **cfit** object which contains separate variables for each unknown parameter. The syntax to evaluate the value of the model function at a specific value of x is

```
y_At_x = fitResult(x);
```

You will now differentiate the interpolated data to obtain the velocity using the **PhysTrack.deriv** function. The syntax for using this function is

```
[xd, yd] = PhysTrack.deriv(xdata, ydata, order)
```

where order is 1 for computing a first-order derivative and 2 for a second-order derivative.

Q 6. Using the three displacement datasets, generate three velocity datasets and plot the results.

To get the velocities of the pucks at any instance, we need to fit the velocity data to an appropriate model. Use the least-squares fitting function **PhysTrack.lsqCFit** again to fit the three velocity data sets to Equation 2.

$$v = f(t) = b_0 t + b_1. \tag{2}$$

Clearly, to get the velocity of an object using this fit, we need to evaluate the value of the model function at some known time instance. For example, to get the velocity of puck A just before the collision, we need to evaluate the value of the model function using their respective coefficients at time **t(IoC)**.

Q 7. Find out the velocities of the pucks just before and after the collision.

Now, we need to verify the following two relationships to conclude that the momentum of the system is conserved.

$$m_A v_A = m_A v'_A \cos\theta_A + m_B v'_B \cos\theta'_B \tag{3}$$

$$m_A v'_A \sin\theta_A = m_B v'_B \sin\theta'_B \tag{4}$$

We can also determine the type of collision using the data we have gathered and processed. We can find whether the collision was elastic or inelastic.

Q 8. Using the mass of the puck, calculate the linear kinetic energy of puck A and B before and after the collision.

Q 9. Using the results of your analysis, explain the energy balance of the system?

4 Exploration Points (Optional)

Q 10. What is the total kinetic energy of the system before and after the collision? Is there any difference between the two values? What does this tell you about the type of collision between the two pucks in this experiment?

Change the mass of one puck by sticking two pucks together. Now, repeat the experiment and analyze the results.

You can get angular displacement of each puck using the trajectory of track markers `(trPtXN)` and function `PhysTrack.GetAngDispFrom2DtrackPoints`. The syntax is as following.

```
thA = PhysTrack.GetAngDispFrom2DtrackPoints(trajs.tp1,trajs.tp2);
```

This data can be used to obtain the angular velocities before and after the collision using similar fashion as we did with linear displacement.

Q 11. Calculate the kinetic energy of puck A before and after the collision considering the angular motion. Repeat for puck B.

Q 12. Plot the angular displacement of the pucks against time. What is the most appropriate function for fitting the angular displacements.

Q 13. Considering the angular motion, calculate the total kinetic energy of puck A and B before and after the collision.

Q 14. Considering the angular motion, explain the energy balance of the system?