

Warning: there's a lot to do in this lab, so you'll want to work very efficiently.

(Equipment: 2 motion sensors, 2 carts, weight for 1 cart, projectile catcher, projectile launcher & ball, 2 photogates.)

## Objectives

In this lab, you will do the following:

- Perform and analyze experiments involving various collisions in one dimension.
- Write a program to explore how the scattering angle varies with the impact parameter for Rutherford scattering.

## I. One-Dimensional Collisions

### A. Background Theory

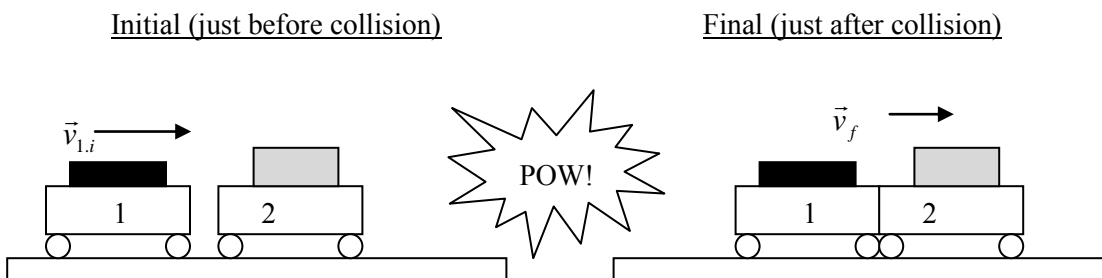
We define “collision” as an interaction that's short and sharp enough so that we can ignore any *other* interactions during it. For example, for the brief moment that a dropped ball hits the floor, you can ignore air resistance. Thus the colliders can be approximated as an isolated/closed system so  $\Delta\vec{p}_{system} \approx 0$  and  $\Delta E_{system} \approx 0$ . A collision is “elastic” if  $\Delta K_{system} \approx 0$ ; otherwise it's “inelastic.”

### B. Set Up

- Carefully level the track. (This step is very important!)
- Plug two motion sensors into Lab Pro in “DIG/SONIC 1” and “DIG/SONIC 2”.
- Place one motion sensor at each end of the track and set them to the narrow range.
- Open file “Collisions” which is in “PHYS231”.
- With the two carts stuck together near the center of the track, zero both motion sensors by selecting “Zero” in the “Experiment” menu.
- Check that the sensors are set up so that they both consider the same direction positive: press the “Start” button, then move your hand between the sensors. Since you will always be measuring the  $x$  components of velocities, the subscript “ $x$ ” will be left off. Be aware that the signs of the velocity components are important because they indicate direction.
- Find the mass of the cart with the weight on it (call this cart  $b$ ).
- Find the mass of the cart with the catcher attached to it (call this cart  $c$ ).


### C. Carts Sticking Together – Maximally Inelastic

The first collision that you'll look at will have one cart hit another that was at rest, the two stick together and roll off together.



**Question:** How do you expect the total momentum before this collision to compare to the total momentum afterwards?

**Question:** How do you expect the total measured translational *kinetic* energy before this collision to compare to the total measured translational *kinetic* afterwards?

- To make the carts stick together after a collision, the sides with Velcro should be facing each other.
- Place cart *c* (the one with the catcher) near the middle of the track. It should be at rest.
- Press the “Start” button and wait for the motion sensors to click rapidly then roll cart *b* at cart *c*.
- Determine *x* components of the initial (*just* before the collision) velocities of the carts by selecting the appropriate portions of the graph and performing a linear fit. To do that, press the linear fit button . Be careful to associate the right velocity with each cart! The initial speed of cart *b* should be around 0.6 m/s.
- In a similar way, determine *x* components of the final (*just after* the collision) velocities of the carts. The measurements should be very similar because the carts are stuck together.
- Find the average of the two *x* components of the final velocities.
- Calculate the *x* component of the total, initial momentum.
- Calculate the *x* component of the total, final momentum.
- Calculate the percent difference between the total initial and final momentum values. The measurements should be within about 10% of each other (which is consistent with the momenta actually being equal). If you don’t get good agreement, double check your work – either using data points from *during* the collision itself or using ones from too long before or after the collision will mess up your measurements, so you can try reselecting points or redoing the collision.
- Calculate the percent of the initially measured translational kinetic energy that gets converted to thermal/internal energy in the collision. That is the percent difference between the final and initial kinetic energy you measure.

#### D. Carts Repelling Each Other Gently - Elastic

Now you’ll again roll one cart into another, stationary cart, but this time they’ll *gently* bounce off of each other. Here, “gently” means that the carts won’t actually hit, magnets in their ends will repel them from each other.

**Question:** How do you expect the total momentum before this collision to compare to the total momentum afterwards?

**Question:** How do you expect the total translational kinetic energy before this collision to compare to that measured afterwards?

- To make the carts repel each other during a collision, the sides with magnets should be facing each other. Remember, the carts don’t have to hit each other for the interaction to be considered a collision.
- Place cart *b* near the middle of the track. It should be at rest.
- Press the “Start” button and wait for the motion sensors to click rapidly, then roll cart *c* at the other one. *If the carts touch each other, try the experiment again.*
- Use the graph to determine the *x* components of the initial and final velocities and record the measurements including the signs. The initial speed of cart #2 should be *at least 0.5 m/s* (then again, you don’t want the carts to ‘clang’ against each other, so you may have to try a few times before you get it just right.)

- Determine and record the carts' initial and final velocities.
- Calculate the x component of the total, initial momentum.
- Calculate the x component of the total, final momentum. Show your work.
- Calculate the percent difference between the total initial and final momentum values. The measurements should be within about 10% of each other (which is consistent with the momenta actually being equal). If you don't get good agreement, double check your work – either using data points from *during* the collision itself or using ones from too long before or after the collision will mess up your measurements, so you can try reselecting points or redoing the collision.
- Calculate the percent of the initially measured translational kinetic energy that gets converted to thermal/internal energy in the collision. That is the percent difference between the final and initial kinetic energy you measure.
- Calculate the percent of the initially measured translational kinetic energy that gets converted to thermal/internal energy in the collision. That is the percent difference between the final and initial kinetic energy you measure. To the extent that this collision is perfectly elastic, there should be no difference; given our measurement uncertainties, a measured difference of less than 10% is reasonably good agreement - if it's much larger, you should double check your measurements and calculations.

#### *E. Ball Fired at a Cart*

For a slightly more extreme collision, you'll 'shoot at' one of the carts.

- Remove the cart with the black mass attached to it.
- Unplug one of the motion sensors and remove it from the track (loosen a wing-nut under the track, then you can slide the detector and its bracket off the end.)
- Orient the track and the remaining cart (the one with the catcher) so that the cart is near the launcher and the launcher aims straight into the catcher.
- Find the mass of the ball.
- To load, push the ball into the launcher so that it clicks *three* times.
- Take a test shot or two to make sure the launcher and catcher are well aligned.
- When you're ready for the real shot, press the "Start" button and wait for the motion sensor to click rapidly, then shoot the ball into the catcher.
- Use the graph to determine the x component of the final velocity.
- Repeat this four more times and record the average of all five runs.
- Calculate the ball's initial speed (just before the collision) based on your measurements.

Now you'll set things up to more directly measure the ball's speed before collision – measure the time it takes the ball to shoot between two photogates that are a measured distance apart.

- The bracket holding two photogates should already be attached to the launcher; if you're your instructor can help you attach it.
- You'll need to unplug the remaining motion sensor so you can then plug the photogate closest to the launcher into "DIG/SONIC 1" and the other one into "DIG/SONIC 2".
- Close the current instance of LoggerPro and open the experiment file "Launch Timer" which is in "PHYS231".
- Measure the distance between the detector of one and the other photogate.

- Load by pushing the ball into the launcher so that it clicks *three* times.
- Press the “Start” button and shoot the ball into the catcher.
- Record the time it takes for the ball to pass between the photogates (the program reports the time at which a gate becomes blocked and again when it becomes unblocked – you’ll want to take the difference between two like measurements, say when one gate gets blocked and when the next gate gets blocked.)
- Repeat four more times and calculate the average.
- Use the distance and time measurements to calculate the speed of the ball.

**Question:** Are the speeds determined from the collision and from the more direct measurement with the photogates consistent (what’s their percent difference)? If not within 10% off each other, find and fix the error.

## II. Rutherford Scattering

### A. Explain and Predict Program Visualization

Now for something completely different...A minimal working program has been provided in WebAssign. Right click on the link file and save it. Remember to add your names in a comment.

**DO NOT RUN THE PROGRAM YET.** Read through the program together. Make sure everyone in the group understands the function of *each* program statement. Reading and explaining program code is an important part of learning to create and modify computational models.

After reading *every* line of the program, use a whiteboard to help you consider how the current version of the program compares with reality (and thus determine what changes you need to make to the program):

- **Reality:** What is the physical system being modeled? In the real world, how should this system behave? On the left side of the whiteboard, draw a sketch showing how you think the objects should move in the real world.
- **Program:** Before running the program, consider whether, *as it is now written*, the program would accurately model the real system; look over the program carefully. On the right side of the whiteboard, draw a sketch of how the objects created in the program will move on the screen, based on your interpretation of the code.

Predictions	
Real World	VPython Output

- Run your program after everyone agrees on both predictions. Discuss how closely your prediction of what the program would do matches what you see in the visual output.

### B. Modify the program

Next, modify this program to model the behavior of this system in the real world (consider the motion of both objects.) Recall how you cycle through determining force, momentum, and position for an object.

- All repeated calculations go inside the calculation loop in the program:
  - Calculate the (vector) forces acting on the object and their sum,  $\vec{F}_{net}$ .
  - Update the momentum of the object:  $\vec{p}_f = \vec{p}_i + \vec{F}_{net}\Delta t$ .
  - Update the position:  $\vec{r}_f = \vec{r}_i + \vec{v}\Delta t$ .
  - Repeat.
- Use the attributes of 3D objects and other variable or constants that already exist in the program you obtained.
- Predict what you would expect to see when you run the program.
- Run the program and make any necessary changes to the code to achieve your goal. Make sure you run the computation long enough to see a trail left by the gold nucleus. Is the

behavior what you expected? If not, check that your code correctly expresses the relevant physics.

### C. Prediction of momentum plots (on your whiteboard)

Think about what happens to the momenta of the alpha particle, the gold nucleus, and the combined system of alpha particle plus gold nucleus, during the collision modeled by your program.

- Sketch plots of the  $x$  component of momentum for the alpha particle, gold nucleus, and total momentum of the system vs. time.
- Sketch plots of the  $y$  component of momentum for the alpha particle, gold nucleus, and total momentum of the system vs. time.

CHECK: Compare your predicted plots to those of another group.

### D. Add momentum plots to the program

- To create plots for the gold and alpha particle's  $x$ -components of momenta, add the following lines of code before the loop.

```
gdisplay(xtitle="Time", ytitle="Px")
alphapx = gcurve(color=alpha.color)
goldpx = gcurve(color=gold.color)
totalpx = gcurve(color=color.green)
```

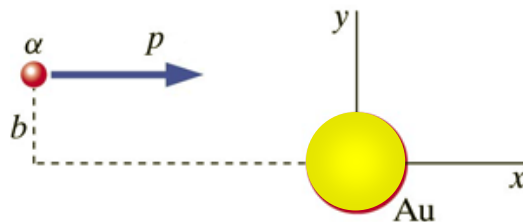
- Add another four lines for creating plots of the  $y$ -component of momentum.
- After the momenta are updated in the while loop, add commands to add points to the graphs. For example, the following line is for the  $x$  component of the alpha particle's momentum.

```
alphapx.plot(pos=(t, alpha.p.x))
```

- Compare the plots generated by the program with your predictions.

### E. Impact Parameter

The impact parameter  $b$  is what the closest distance between centers of the alpha particle and the gold nucleus would be if there were no deflection, as shown below.



- Set  $b = 5 \times 10^{-15}$  m in the initial values section of the program.
- Use  $b$  to change the initial position of the alpha particle.
- How do you think your  $p_y$  graphs will change due to the new impact parameter?
- Run your program.

## F. Scattering Angle

Following the discussion in section 9.7 of your text, if a broad and uniform beam of  $\alpha$ -particles were shot by a gold nucleus, then the fraction *approaching* with a particular *impact parameter* will simply scale with the impact parameter, meanwhile the fraction *coming out* with a particular *scattering angle* will also depend upon the interaction force's distance dependence. Therefore, a comparison of an experimentally observed probability of different scattering angles and the theoretically predicted probabilities allows us to determine the force's distance dependence. You won't be asked to work all that out, but hopefully you'll appreciate the significance of the one bit you will find: the Impact Parameter – Scattering Angle relationship.

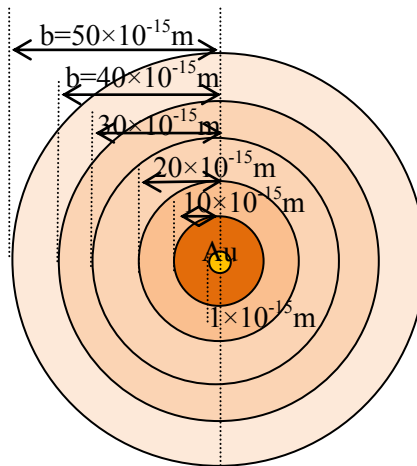
The scattering angle is calculated using the direction that the alpha particle is traveling (the direction of its momentum) after it is far from the gold nucleus. The angle is simply  $\theta = \tan^{-1}(p_{\alpha,y}/p_{\alpha,x})$ , where components of the final momentum are used. In VPython, when given the  $y$  and  $x$  components (*in that order!*), the function `atan2` returns the angle in radians.

- Place the following lines after and outside the while loop (that is, *un-indented*). The last part converts the angle from radians to degrees.

```
theta = atan2(alpha.p.y, alpha.p.x) * 180/pi
print theta
```

- Run the program with impact parameters from 0 to  $100 \times 10^{-15}$  m and record the results.
- Plot the scattering angle vs. the impact parameter below.

The actual experiment that someone would do (indeed, that some students do in our Advanced Experimental Physics course) involves comparing the fractions of alpha particles that are deflected through different angles. Here's how those fractions relate to the data that you've simulated. Imagine a target for a game of darts with its bull's eye and concentric rings. The gold nucleus sits at the bull's eye, and each impact parameter corresponds to the average radius of the target ring that scores a  $57^\circ$  deflection, a  $40^\circ$  deflection, etc.



If we fire a uniform spray of alpha particles at the target, then the fraction that land in a given ring is simply the ratio of that ring's area to the target's total area. So, the fraction that 'score' a particular deflection angle = the fraction that lands in that ring = the ratio of the ring's area to the total target's area.

So, you'll determine the fraction of alpha particles that should deflect around a given angle.

- Calculate the areas of rings of radii equal to each of the impact parameters that you'd used; for the impact parameter of 0, we'll use  $r = 1 \times 10^{-15} \text{m}$ . Note: the area of a given *ring* is the area of the full circle minus the area of the next smaller circle.
- To get the fraction of alpha particles that would shoot within each ring, divide each of the ring's area by the total area, that of a circle of radius  $100 \times 10^{-15} \text{m}$ .
- Plot that fraction against the corresponding deflection angles you'd recorded in the previous table.

**Pause and Consider:** Theoretical 'data' plots like the one you've generated can be used to test theoretical models even when it's impossible (or just very difficult) to write down analytical expressions for, say, the number of particles that should be deflected through a given angle. In this case, assuming a simple Coulomb force and the charge of the proton, you generated a specific plot of expected data; if it were a different element or a different force law, the data plot would be different. For example, at particle accelerators, working at *much* higher energy scales, particles are shot at nuclei to learn about the forces that hold the nucleus together; theoretical models are used to generate expected data which is compared against the real data in order to test the models.

**Save Rutherford.py and upload it via WebAssign.**