Calculating and displaying the magnetic field of a single moving charged particle

1 Introduction

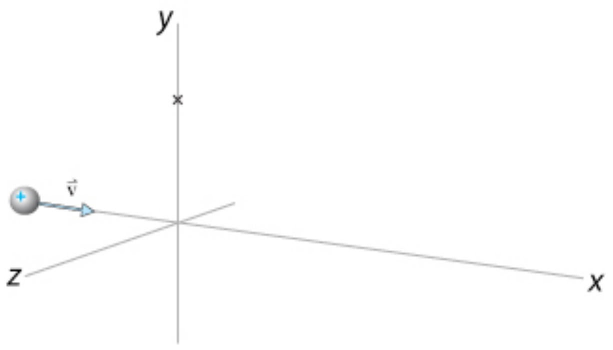
A proton moves with a constant velocity of 4$\times$10^4 m/s in the +x direction along the x-axis, starting at an initial position of $\langle -4 \times 10^{-10}, 0, 0\rangle$ m. You will write a computer model that calculates and displays the magnetic field of the proton at four observation locations in the y-z plane (two on the y-axis, two on the z-axis), each a distance of 8$\times$10^{-11} m from the x-axis.

Start with the program shell. Read through the program shell to see how the program is organized and what you will need to calculate.

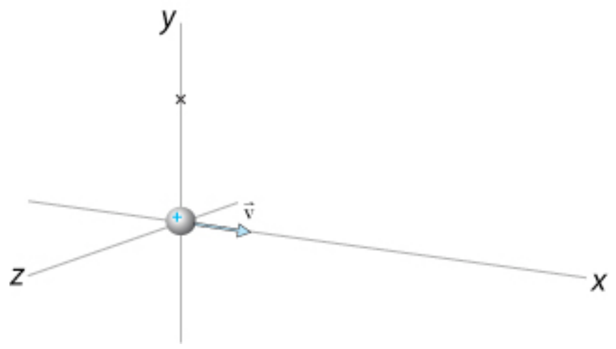
2 Planning and Predicting

Using the problem statement above, on your whiteboard, predict the magnetic field at the observation location (marked by the 'X') for the three "snapshots" shown below.

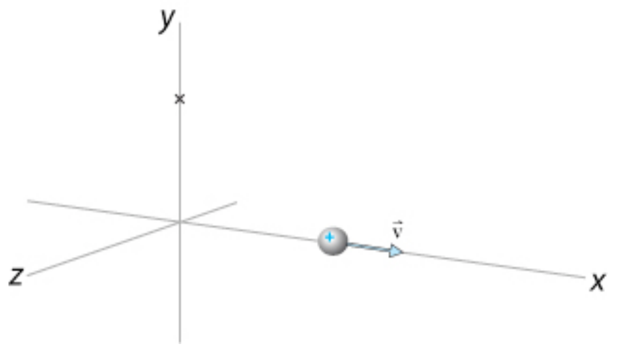
Time 1



Time 2



Time 3



3 Animating the motion of the proton

The program is divided into four sections:

1. Constants: where any important constants are named and given values
2. Objects: where the visible 3D objects like spheres and arrows are created and named
3. Initial values for important variables
4. Loop: where the physics calculations are done at each successive time step, and the motion is updated

3.1 Getting the proton to move

First, we will concentrate on getting the proton to move at a constant velocity.

In the objects section, change the initial position of the proton to the value given in the problem statement. Note that the proton is given a radius of 1×10^{-11} m - this is much larger than the real radius of the proton, but it allows us to see the sphere.

In the initial values section, enter the proton's velocity as given in the problem statement.

In the loop, write the calculation that updates the proton's position.

- A computer animation of the motion of an object uses the same principle as a movie or a flip book. You display the object at several successive locations, each at a slightly later time; to your eye, the motion can look continuous.
- In a program, you calculate the position of the object, and it is displayed in that position. You calculate where the object will be a very short time `deltat` later, and change its position to the new location, so it is displayed in the new location.
- You will use the relation between velocity (a vector) and position (a vector) to calculate the position of a moving particle at successive times, separated by a short time step Δt. Remember that, in vector terms

$$\vec{r}_f = \vec{r}_i + \vec{v}\Delta t \quad (\text{because} \quad \vec{v} = \frac{\Delta\vec{r}}{\Delta t})$$

- In VPython, this translates to a statement like: `particle.pos = particle.pos + velocity*deltat` where velocity is the vector quantity which you have initialized earlier in the program.

- This statement may look odd to you if you have not had much programming experience. The equals sign here has the meaning of "assignment" rather than equality. This instruction tells VPython to read up the old position of the particle, add to it the quantity, and store the new position as the current position of the particle. This will automatically move the particle to the new position.

Run the program and verify that the proton moves in a straight line in the +x direction.

4 Calculating the magnetic field of the moving proton at one location

In the Objects section, `barrow1` is an arrow object that will represent the magnetic field of the proton at the first observation location. The `pos` attribute of an arrow gives the location of its tail with respect to the origin, while the `axis` attribute gives the magnitude and direction of the arrow. Currently, the axis is set to $\langle 0, 0, 0\rangle$, so no arrow will be displayed until you add code that calculates a new axis.

In the Objects section, change the `pos` attribute of `barrow1` to the observation location on the +y axis 8×10^{-11} m away from the origin.

Because the magnetic field of the proton changes as the position of the proton changes, the magnetic field calculations go in the loop.

First, think about what information you will need to do this. What is the formula for the magnetic field of a moving charged particle (Biot-Savart law)? What quantities in the formula change as the proton moves? What quantities are constants? Answer these questions on a piece of paper.

- In the loop, first calculate `r1`, the position vector that points from the proton's position to the observation location. Think about how you would do this symbolically, not with numerical values. The vector position of an object, such as a proton, is given by `proton.pos`.

- After finding the vector `r1`, write lines that calculate its magnitude, `r1mag`, and the unit vector `r1hat`.

- Now, use these variables to find the magnetic field of the proton. Create and assign values to any constants you may need in the "Constants" section of the program.

- Remember that VPython has a built-in cross product function: $\vec{A} \times \vec{B}$ can be represented as `cross(A,B)` in VPython.

- Run the program. You will find that there is still no arrow displayed, because the arrow's axis is not yet updated.

5 Displaying the magnetic field arrow

Once you have calculated the magnetic field, you need to use it to determine the axis of the arrow. You will need to scale the axis so that the arrow has a reasonable length, so you will need to determine a scale factor.

To estimate a scale factor, suppose you want the arrow to be small but visible when the proton is in its current location. In the distance units of the display, how long should the arrow be? (What is the distance between the proton and the observation location? How do you want the length of the arrow to compare to that distance?) Show your calculation on your paper.

Remember that scalefactor = (max. length you want the arrow to have)/(max. magnitude of the field it represents)

At the moment you don't know the maximum magnitude of the magnetic field, so you may need to adjust the scale factor later.

(To save some work, you can add a `print` statement in the loop that prints the value of the magnetic field and use this to help find the scale factor. To quickly find the maximum magnitude of the field, temporarily change the proton's initial position to the origin. Be sure to comment out the `print` statement when finished.)

- Create a constant scalefactor in the Constants section and assign it a value.
- In the loop, change the axis of the arrow by adding the following statement:
  ```
  barrow.axis = (variable representing magnetic field)*scalefactor
  ```
- Run the program. Does your magnetic field vector behave as expected? Does it have the correct direction? Check direction by using the right hand rule.

6 Adding observation locations

Add three additional observation locations, in locations with the same x-coordinate, but different y and z coordinates, so that the four locations are at the corners of a square surrounding the path of the particle. Your finished program should display the magnetic field vectors simultaneously at all these locations as the particle moves. Set the scalefactor to a value that makes the arrows representing magnetic field large, but not so large that they extend offscreen. Answer these questions on your paper.

- Run the program. Do the field arrows behave as expected? Are they in the correct directions?
- Change the sign of the charge of the particle – turn it into an antiproton. What happens to the magnetic field vectors?
- Double the particle's speed. What happens to the magnetic field vectors?
- Change the position of one of the observation locations so that it is twice as far from the origin. What happens to the magnetic field at this location?

Last edited: Tuesday 22$^{\text{nd}}$ May, 2012 at 4:52pm