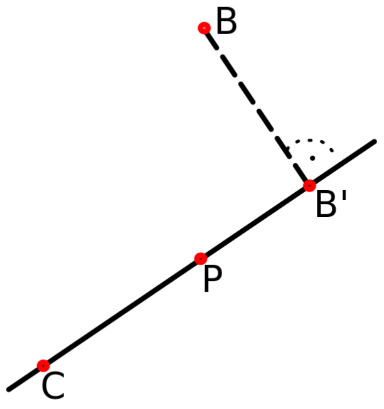
Assignment 4: Picking and dragging balls with the mouse

The code in the file `Assignment4.js` displays a set of balls in a canvas. The task is to complete the code in this file such that the ball picked with the mouse is highlighted. In addition, the picked ball is dragged around when the mouse with the left button pressed is moved.
Do *not* use the `THREE.Ray` and `THREE.Raycaster` classes for this assignment.

1 A math problem as preparation

Consider three given points C, P and B in $\mathbb{R}^3$. B' is the point on the straight line passing through C and P that is closest to B.

1. Derive a formula for the vector $\overline{B'B}$ from B' to B. Express your result just using the vectors $\overline{CP}$ and $\overline{CB}$.
2. Verify that the vectors $\overline{CP}$ and $\overline{B'B}$ are perpendicular to each other by working out the scalar product $\overline{CP} \cdot \overline{B'B}$.



2 Requirements:

1. Highlight a ball if the left mouse button is clicked when the mouse is over the ball by setting its emissive light component equal to its color (i.e. its diffuse reflection coefficient).
2. Undo the highlighting when the left mouse button is released by resetting the emissive light component of all balls to black.
3. When a ball is picked (and only then), print the following data to the console:
   - the normalized device coordinates of the center of the ball,
   - the camera space coordinates of the center of the ball,
   - the world space coordinates of the center of the ball,
   - the distance of the center of the ball to the straight line passing to the camera and the point picked on the screen.
4. Drag the picked ball with the mouse when the mouse with the left mouse button pressed is moved around after picking a ball.
5. Use the code structure as provided in the file `Assignment4.js`. Just implement the function `pickOrDragBall` and the mouseup-callback. Do not change the file anywhere else.

6. **Do not use `THREE.Ray` or `THREE.Raycaster` objects but implement the picking functionality all by yourself!!** Just use the Math functionality provided by `THREE.Vector3` (and `THREE.Vector4` if you like).

Hints

- Each ball is represented by a `THREE.Mesh` object. All balls are stored in an array called `balls`. A ball `b` has its radius stored in `b.userData.radius`.
- Without loss of generality you may assume that clicking into the canvas with the mouse defines a point P on the near plane. The viewport coordinates of this point are passed to the function `pickOrDragBall`. Work backwards through the graphics pipeline (slide 45 of chapter 9) to map the point P first to NDC space, then to camera space and finally to world space. Together with the camera position C this defines a straight line in world space. Use the result of the preparatory math exercise to find the smallest distance between this line and the position B of a ball.
- Feel free to handle the case of overlapping balls as you like.

3 Coding style

Stick to the coding style guide which can be found in the Readme file for chapter 3 in the gitlab repository. For this assignment `console.log` statements in the submitted code are allowed (but only those that are necessary for printing the required data).

4 Handing in the solution

No group work allowed. Every course partiticipant has to write her or his *own* code! The deadline for submission through gitlab is the **31st of January 2021**.