Background

The **hw7_gate.py** (Python) and the **hw7_gate.m** (MATLAB) examples from the course website show how an artificial potential function could be used to steer a robot through a “gate” made from two pylons. Download either of the two examples, read over it, and run it before answering the following questions (please type your answers).

When you run it, you will see four plots corresponding to four different starting positions of the robot. Each plot is titled to indicate the starting point in the gate’s coordinate frame, along with indications of whether the robot hit a pylon, and whether the robot successfully made it between the pylons to get through the gate. A run is considered successful when the robot doesn’t hit the pylon and goes through the gate.

1. Getting through the gate

Increase the `k_gate` constant until the robot successfully passes through the gate from each starting point (I’d start at about 0.5 and ramp up from there). Then, gradually increase `t_gate` and `k_gate` together until the robot goes through the pylons without collisions.

   a. What values worked out well for the two constants?

   b. Describe what happens to the potential field (contour plot) and the resulting force vectors (arrow plot) as you modified these values.

2. Potential function

Alas, I didn’t comment my code. Good thing you’re here to make sense of it.

   a. What is the purpose of the `U_x` component of the potential? The `U_y` component? `U_obs`?

   b. Consider the two infinite rays extending up and down along the `y`-axis from the points at `y = ±gate_rad`. Explain how the `d_obs` variable computes the distance from any point `(x, y)` to the nearest point on either ray. Why does computing the absolute value `|y|` come into play here?

   c. What is the purpose of the variable `close` being used as an index into the `d_obs` array?
d. Sketch (or graph) a plot of the relationship between \( d_{\text{obs}} \) (independent variable) and the \( U_{\text{obs}} \) potential component (dependent variable). Clearly indicate where \( t_{\text{gate}} \) is on the \( x \)-axis.

3. Getting stuck

Why does the robot seem to get stuck when you set \( k_{\text{gate}} \) to 3.0 and \( t_{\text{gate}} \) to 1.0? What undesirable phenomenon does the potential function develop?

4. Checking for collisions

Explain how the collision checking code involving \texttt{dpylon} works.

5. Computing forces

Explain how the code inside \texttt{force.at} can be said to be taking a “poor man’s gradient” of the potential function. Why is this easier to code than computing the correct analytic gradient?

6. Integrating forces

Outline the method being used to simulate the robot motion in \texttt{integrate_forces}. What configuration space does our simulated robot live in? Are we explicitly modeling the robot’s orientation?