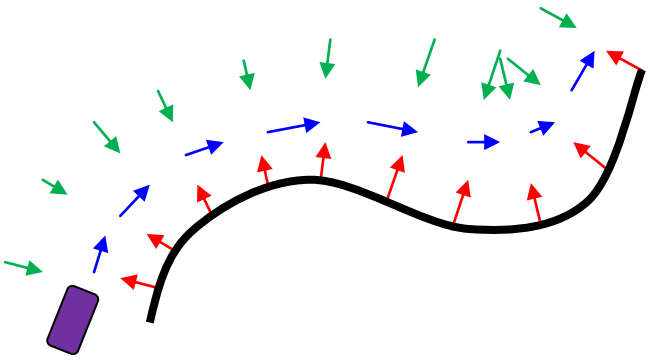
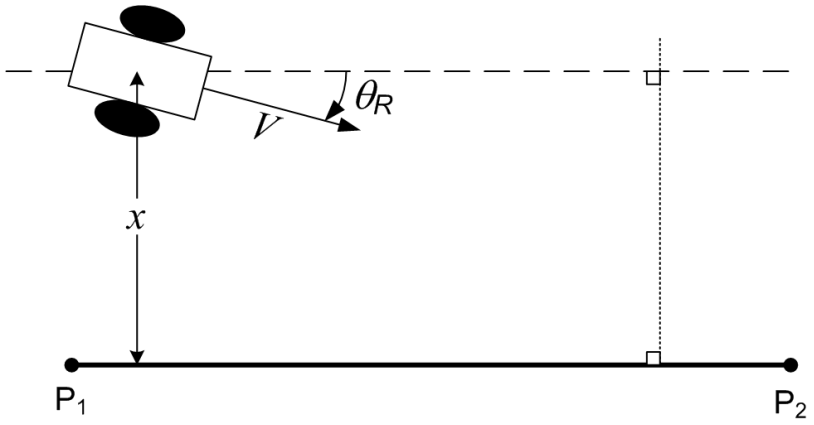
Assignment 3, Due November 15th, 2010
(Worth 10%)

1. **45%**) Use a potential field to guide the robot to the desired distance from the wall, e.g. 1 m and then use an attractive force parallel to the obstacle (perpendicular to the orientation the obstacle is detected) to move it along. Use attractive and repulsive forces similar to assignment 2. Using player and stage, write a program that will make the robot follow the closest obstacle.



2. **45%**) In class, we saw different components that can be used in a controller:
   a) a proportional component (P)
   b) a derivative component (D)
   c) an integral component (I)

   The goal of this question is to get you familiar with how to use this type of controller, and particularly in the context of robotics. As such, you will get a differential drive robot to perform a path following task.

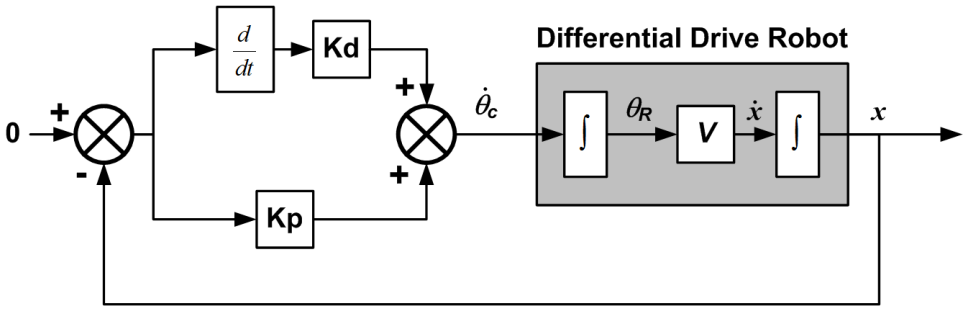


In terms of variables for this problem, we have:

- distance x to the wall;
- desired distance x_d to the wall
- desired heading of the robot , relative to the wall θ_d;
- the desired rate of turn of the differential drive robot ω_R, (*turnrate*);
- actual heading of the robot θ_R;
- the *fixed* forward velocity of the robot V.

In the context of control theory, wall following can be defined as trying to maintain x = x_d. This also implicitly requires θ_d = 0, otherwise the robot would drive away from the wall.

We saw in class how we can use a P-D controller to maintain the trajectory of a robot on a line, while driving at a constant velocity V. The P part was getting the robot to move towards the line, and the D part "cued" the controller to reduce the correction as it gets closer to the goal. Without the D part, the robot was oscillating constantly around the line. The block diagram for this approach is shown below. The leftmost 0 simply indicate we want to be at a distance 0 from the wall. The grey box is a simplified model of the differential drive robot.



Using player and stage, write a program that will make the robot follow the closest obstacle. You can either start the robot fairly close to one obstacle, or write a function that finds the closest obstacle and then moves towards it until it reaches near x_d.

3. **10%**) Create a world (modify one of the *png* files) in such a way to demonstrate limitations of the two techniques. For example, have a corner that is two narrow, thus requiring the robot to make a big rotation; introduce a second obstacle that is closer than twice the desired distance. Discuss the resulting behavior of the two approaches.

Hints

- **Direction**
  If your system diverges, it might be because the commands have the wrong direction. Double check that by placing the robot near an obstacle, and look how your error and

commands change. If this is the source of your problem, simply multiplying by -1 the troublesome value.

- **Computing angles**
  Use atan2, not atan.

- **Phase Unwrapping**
  Remember that computations on angles are tricky. After any computation, you might want to shift the angle between $-\pi$ and π by adding or subtracting multiples of 2π. Simple examples of faulty computation if you do not do that, with the correct phase unwrapping (example given in degrees but the same applies for rad):
  - 0-360=-360 (wrong) -360+360 (phase unwrap) = 0 (correct)
  - 170 - -180 = +350 (wrong) +170 - 180 (phase unwrap) = -10 (correct)

- **Capping the Heading Command**
  If the robot is far from the wall, the heading command θ might be more than $\pi/2$ rad, in which case the robot will go in the wrong direction, or start making spirals. You might therefore want to cap this command c to be within +/- $\pi/2$.

Your assignment should consist of a pdf document that discusses your approach, difficulties and potential improvements, and a copy of your source code for verification purposes.