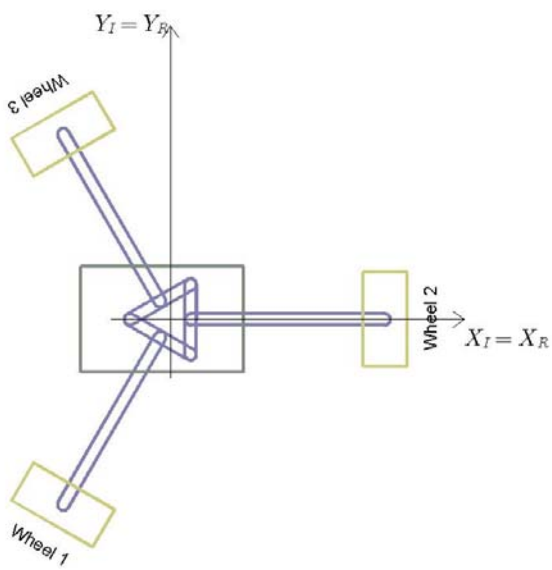
Prelab 4: Omnibot trajectory planning and feedforward control

This prelab involves 3 goals: 1) Planning kinematics for wheel rotations of the omnibot, to move the “body” of the vehicle as desired, 2) determining how “quickly” to play back these trajectories over time, and 3) designing a feedforward element to your controller, to compensate for both linear and nonlinear dynamics of the motor system.

The prelab involves both paper calculations and generation of MATLAB code. Even if your code does not work perfectly, it should help enormously for you to have worked out some of the issues in planning and control ahead of time. You will need to build off of your code this week to prepare for Lab 5, as well, so don’t fall behind!

Problem 1) Derive the Jacobian matrix that relates joint velocities, $\dot{q} = [\dot{q}_1 \quad \dot{q}_2 \quad \dot{q}_3]^T$ (i.e., wheel rotations,) to velocities of the center of body of the vehicle chassis, $\dot{\xi}_I = [\dot{x}_b \quad \dot{y}_b \quad \dot{\phi}_b]$, where the subscript I indicates that coordinates are with respect to the global (aka “inertial”) reference frame. (Recall the Jacobian is defined by: $\dot{\xi} = J\dot{q}$.) For full credit, your derivation should include steps such as 1A through 1C below; parts 1D and 1E are required MATLAB. Pages 47-60 in the Siegwart handout (from Lecture 7) may be helpful; equation (3.19) has the essential relationship you are trying to derive!



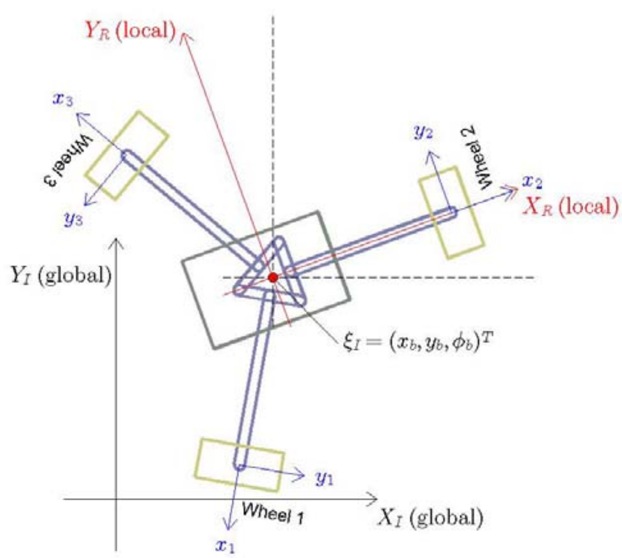


Figure 1. Left: Omnibot, with global (aka Inertial) and local (aka Relative) coordinate frames aligned. Right: Relative frame is now offset and rotated, wrt inertial frame.

Figures 1 above defines ξ_I. Where each wheel contacts the ground, there is a local coordinate frame, (x_i, y_i). For the i^{th} wheel, x_i points radially outward from the body, along the direction of the wheel axis, and y_i is oriented 90 degrees counter-clockwise to this (as viewed from above). The rotation of the i^{th} wheel is the i^{th} joint angle, q_i, where counter-clockwise rotation of the wheel about the x_i is considered the “positive” direction of rotation for q_i. The wheel has a radius of r_w, thus:

$$\dot{y}_i^b = -r_w \dot{q}_i$$

That is, as the wheel rotates in the "positive" direction, the instantaneous motion along the y_i^b axis is "negative", scaled by the wheel radius. Note the superscript "b" is simply a reminder that y_i is a coordinate defined with respect to the BODY of the vehicle: the instantaneous direction of y_i actually points depends on the rotational orientation of the body, ϕ_b.

1A) Write expressions for the x and y coordinates of each wheel contact point with respect to the origin of the global coordinate frame, x_i^o and y_i^o, in terms of the body coordinates (x_b, y_b, and ϕ_b), the distance L from the center of body to each wheel contact point, and the angle, η_i, of each wheel's coordinate from with respect to the body coordinate from. (i.e., $\eta_1 = -120^\circ$, $\eta_1 = 0^\circ$, $\eta_3 = 120^\circ$)

1B) Differentiate your answer in 1A to derive expressions for $\dot{x}_i^o$ and $\dot{y}_i^o$, in terms of the body coordinates (x_b, y_b, and ϕ_b) and the body velocities ($\dot{x}_b$, $\dot{y}_b$, $\dot{\phi}_b$).

1C) We still need to relate wheel rotations to body motion. Derive an expression relating $\dot{y}_i^b$ (which is linearly related to wheel rotation) to $\dot{x}_i^o$, $\dot{y}_i^o$, $\dot{\phi}_b$, and the geometry for each wheel

1D) Finally, write a MATLAB function that combines all results above to solve for the 3x3 matrix F, below, that can transform desired vehicle velocities to required joint velocities:

$$\dot{q} = \frac{-1}{r_w} \cdot \begin{bmatrix} \dot{y}_1^b \\ \dot{y}_2^b \\ \dot{y}_3^b \end{bmatrix} = F \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{\phi}_b \end{bmatrix} = F\dot{\xi}$$

Assume $r_w = 0.032$ (meters) and $L = 0.146$ (meters). You function should take the current rotation of the vehicle, ϕ_b, as an input.

1E) Recall the Jacobian is defined as: $\dot{\xi} = J\dot{q}$. Thus, assuming F is invertible, then $J = F^{-1}$. Using MATLAB, numerically calculate the value of J for ϕ_b of 0, 30, 45, and 90 degrees.

Problem 2) Using your code from Problem 1, write additional MATLAB code to generate joint trajectories to move the center of body of the robot in a square pattern. Use 1 meter length sides for your square. For this problem, you may assume the vehicle moves at a constant velocity along each of the four sides of the path.

Problem 3) Now, ensure that your motions are "smooth". Recall that at peak power, the motor spins at only about 950 deg/s (roughly) without a load; it may spin a bit slower when moving the vehicle. Also, there is a time constant of at least .06 seconds for the motor to reach its peak velocity. It is important to "ramp up" to full speed, to achieve accurate trajectories. Also, unlike a robot arm with a fixed based, the final position of robot depends on the order in which joint commands are executed; therefore, it is important none of the wheels "lags behind" the others. Smoothing out planned trajectories will help in achieving this.

For problem 3, write a MATLAB function that modified the joint trajectories from Problem 2, so that motions are commanded more smoothly. It is up to you to define appropriately inputs to your function.

Problem 4) Sketch (on paper) the structure for a feedforward compensator that will generate a signal to be added to the feedback "power level" command to the motor. Use your motor model developed in Labs 1-3.