ME C231B / EE C220C Final

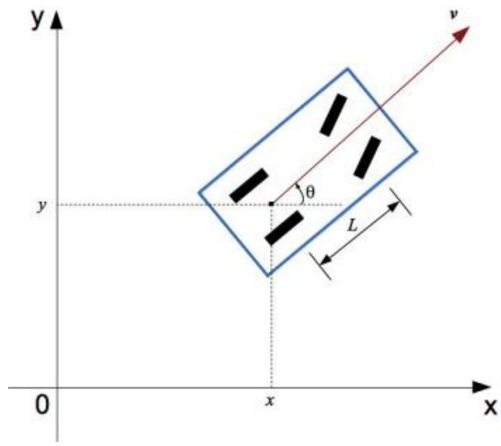
Your Name and Student ID:

Please answer all questions. Make sure to review the Exam Instructions on bCourses before starting the exam. Before each answer report the question number you are answering: 3.a.ii means problem 3, question (a), part ii.

Problem:	Max Score	Score
1	50	
2	5	
3	5	
4	20	
5	20	
Total	100	

1. **Navigation for Parking**
Note this problem seems long, but it is a "simple" navigation problem. You can solve it by starting from the yalmip code I have provided to you. The description is long because I want to make sure you understand the model.

We will consider the following simplified geometric model of a car. Note: this model is 'simpler' than the kinematic bicycle model used in class.



The above figure depicts a car in a global cartesian coordinate space. The vehicle is characterized by its position and heading, $[x, y, \theta]$, where x and y are the coordinate of the center of the rear-axel and θ is the heading relative to the x-axis. The distance between rear and front axel is L.

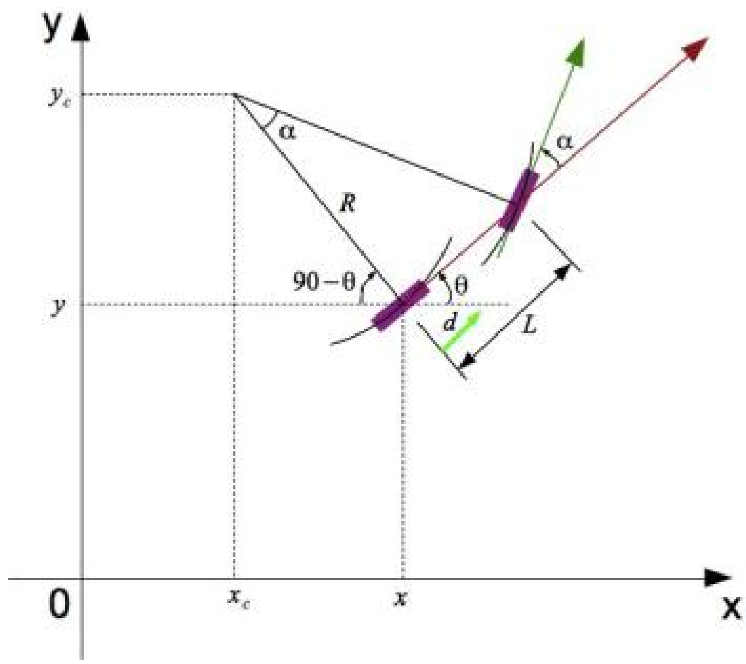
If the car moves of a distance d while steering its front wheel with an angle δ, we have that the turning radius is

$$R = \frac{L}{\tan \delta} \tag{1}$$

and the turn angle is

$$\beta = \frac{d}{L} \tan \delta = \frac{d}{R}. \tag{2}$$

In the next figure the vehicle and its center of turn (with coordinates $x_c,\ y_c$) are depicted, the steering δ is denoted by α in the next figure.



The coordinates of center of turn can be easily computed from the figure as

$$\begin{aligned} x_c &= x - R\sin\theta \\ y_c &= y + R\cos\theta. \end{aligned} \tag{3}$$

The new heading of the car after the turn can be obtained by adding β and original heading θ. The new coordinates $[x^+, y^+]$ of the car after the turn are

$$\begin{aligned} x^+ &= x_c + R\sin(\theta+\beta) \\ y^+ &= y_c - R\cos(\theta+\beta) \\ \theta^+ &= (\theta+\beta). \end{aligned} \tag{4}$$

In conclusion combining (3) and (4) our vehicle model can be rewritten as

$$\begin{aligned} x_{k+1} &= x_k - R_k\sin(\theta_k) + R_k\sin(\theta_k+\beta_k) \\ y_{k+1} &= y_k + R_k\cos(\theta_k) - R_k\cos(\theta_k+\beta_k) \\ \theta_{k+1} &= \theta_k + \beta_k \end{aligned} \tag{5}$$

where the vehicle state at time step k is $z_k = [x_k, y_k, \theta_k]$ (coordinates of center real-axel and heading angle) and the inputs are $u_k = [R_k, \beta_k]$ (turn radius and turn angle). Model (5) will be compactly rewritten as

$$z_{k+1} = f_{car}(z_k, u_k). \tag{6}$$

A few remarks about the model:

- A straight line motion corresponds to an infinite turn radius R_k (i.e., 0 steering angle). Numerically you will observe 'almost' straight lines when R_k is very big.
- For a given input $u_k = [R_k, \beta_k]$ the traveled distance by the car (from eq. 2) at step k is $|R_k \cdot \beta_k|$.

- If the vehicle goes at constant speed the length of the time steps is nonuniform. This means that at time step k you can choose to travel a short distance $|R_k \cdot \beta_k|$ or a long distance $|R_k \cdot \beta_k|$. In other words, the model is event-based. Nothing changes in what you have learned and in the code you have worked on in class.
- Cars have a min and max steering angle $\delta \in [\delta_{min}, \delta_{max}]$. Often δ_{min} is negative and $\delta_{max} = -\delta_{min}$. This implies that in terms of constraints on the turning radius we have that $R \leq \frac{L}{\tan(\delta_{min})}$ OR $R \geq \frac{L}{\tan(\delta_{max})}$. This "OR" constrains will be simplified later. Steering right means $R < 0$, left $R > 0$.
- I have shared with you the function "car_plot". It plots the vehicle trajectory and shows its motion. Try in MATLAB:

```
car_plot([10,-10;pi, pi/4],[0;0;0])
```

  it plots the car trajectory with $z_0 = [0; 0; 0]$, $u_0 = [10; \pi]$, $u_1 = [-10, \pi/4]$, note that in the second part of the maneuver the car is moving backward (assuming you can use a reverse gear). The front wheels are colored in red.

  The syntax is car_plot(U,z_0) where z_0 is a 3×1 vector of initial conditions, U is a $2 \times p$ matrix collecting a generic input sequence $[u_1, u_2, ...u_p]$.
- Note that the simulation of the model (6) with $z_0 = [0; 0; 0]$, $u_0 = [10; \pi]$, $u_1 = [-10, \pi/4]$ gives as output the two states z_1, z_2. The state sequence z_0, z_1, z_2 are in depicted with the green markings (a point and an arrow) by the function car_plot().
- In order to show a nice trajectory, the function car_plot() plots a number "**rep**" of points between z_i and z_{i+1}. You can change the variable "**rep**" in the function car_plot(). You can decrease it if the animation takes too long, increase it to see smoother curves.

You are asked to use model (5) to formulate and solve a parking problem as a finite-time optimal control problem formulated as follows:

$$
\begin{aligned}
\min_{z_0,\ldots,z_N,u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{k=N-1} (R_k \cdot \beta_k)^2 \\
& z_{k+1} = f_{car}(z_k, u_k) && \forall k = \{0, \ldots, N-1\} \\
& z_{min} \leq z_k \leq z_{max} && \forall k = \{0, \ldots, N\} \\
& u_{min} \leq u_k \leq u_{max} && \forall k = \{0, \ldots, N-1\} \\
& z_k(1:2) \notin \text{Obstacle}^{(m)} && \forall k = \{0, \ldots, N\} \text{ and } \forall m = \{1, \ldots, N_{obs}\} \\
& z_0 = \bar{z}_0 \\
& z_N = \bar{z}_N
\end{aligned}
\tag{7}
$$

The vehicle starts from the initial state $\bar{z}_0$. Our goal is to park the vehicle in the terminal state $\bar{z}_N$ while minimizing total traveled distance.

Settings:

- Assume the steering command is broken and you can steer only to the left ($R > 0$) – this gets rid of the 'OR' constraint.
- Set $R_{min} = \frac{L}{\tan(\delta_{max})}$ with $L = 2.8$ and $\delta_{max} = 25\pi/180$. Set $\beta_{min} = -\pi$ and $\beta_{max} = \pi$.

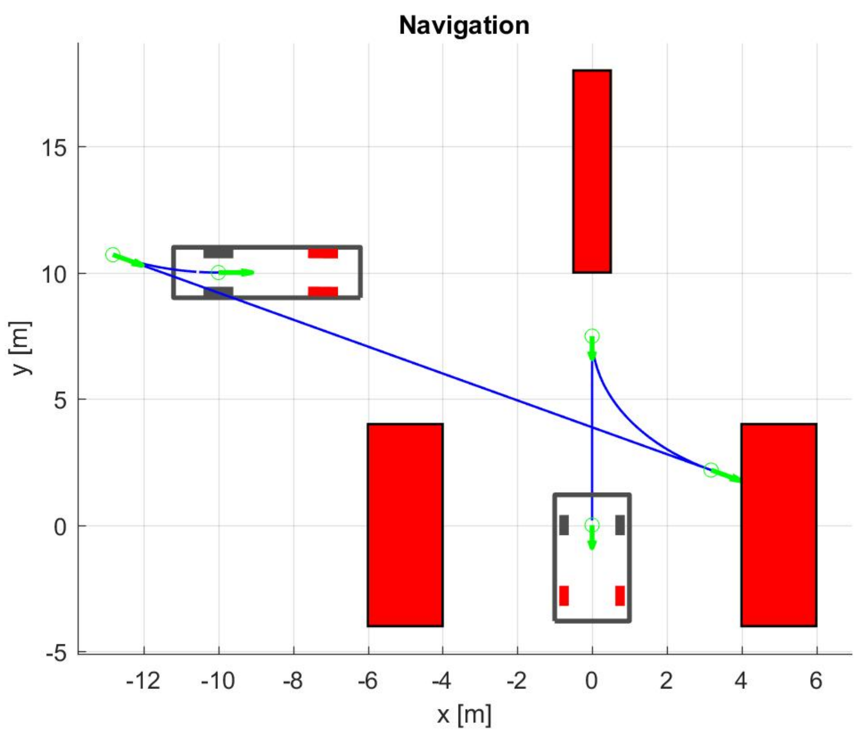
- The min and max input constraints are $u_{min} = [R_{min}; \beta_{min}]$ and $u_{max} = [+\infty; \beta_{max}]$.
- The state has the constraints of $[-20, \quad -10, \quad -2\pi]^T \leq z(k) \leq [20, \quad 20, \quad 2\pi]^T$
- Set $\bar{z}_N = [0; 0; -\pi/2]$
- I have provided an .m code with all the parameters and three obstacles ($N_{obs} = 3$)
- Note that the obstacle-avoidance constraint is formulated only for the point representing the coordinates of the center real-axel.

HERE ARE THE QUESTIONS

(a) Use $\bar{z}_0 = [-10; 10; 0]$, $N = 3$. Look at the the code:"parkingOptimalControl_toshare.m" I provided you *which does not include obstacle avoidance*. Run the code and make sure you can park the car. No deliverables.

(b) Use $\bar{z}_0 = [-10; 10; 0]$. Submit a code and a plot which shows parking while avoiding the three obstacles given in the code. Use the duality approach method. Submit the final plot generated by the function car_plot() and your code.

Note: (1) You can pick the N and the solver you want to solve the problem as fast as you can. I solved it with $N = 4$ and ipopt (my solution below). You are not allowed to change the constraints or the method to encode obstacle avoidance constraint. (2) You might need to enforce constraints at a number of points p between z_i and z_{i+1}. (3) Do the best you can to have "some" obstacle avoidance, it is ok to hit obstacles because the car is modeled as a point mass or because the p is small for computational reasons.



(c) Show what happens with an N larger than the one used at the previous point, submit the final plot generated by the function car_plot().

(d) Code now and solve the same problem in (b) by using the signed-distance formulation of Equation (13) of the document CollisionAvoidanceLatex_final.pdf. Submit the final plot generated by the function car_plot() and your code. NOTE: same comments as point (b) apply.

(e) (ONLY IF YOU HAVE TIME, FINISH THE OTHER PROBLEMS FIRST) Elaborate on how to get rid of the assumption that the car can steer only to the left ($R > 0$). Do not try to code the "OR constraints" in yalmip or with bigM, you would need an MINLP solver. Think of a simple approach and code it if you have time (submit the code and a plot showing that it works).

2. Examine (and run) the code below

```
1 G = tf([-5.8 -3.2 9],[0.02 0.98 4.8 9]);
2 normTol = 1e-5;
3 [A,B] = norm(G,inf,normTol);
4 [A abs(freqresp(G,B))]
5 bodemag(G, frd(freqresp(G,B),B), 'ro')
```

Explain what lines #3, #4, and #5 are doing.

3. Examine (and run) the code below

```
1 normTol = 1e-5;
2 delta = complex(randn,randn);
3 wBar = exp(2*randn);
4 D = cnum2sys(delta,wBar);
5 [freqresp(D,wBar) delta]
6 [norm(D,inf,normTol) abs(delta)]
7 bode(D)
```

Explain what lines #4, #5, #6 and #7 are doing. **Hint:** Refer to slide 3 in the "WorstCaseUncertainSystemAnalysis" powerpoint file for a reminder about the behavior of `cnum2sys`

4. Transfer functions of plant P and controller C are given below

$$P(s) = \frac{1}{s-1}, \qquad C(s) = \frac{5.8s + 9}{s(0.04s+1)}$$

   (a) Verify that C stabilizes P in a standard negative-feedback loop configuration

   (b) Consider additive perturbations to P of the form $P + \Delta$, where $\Delta \in \mathbb{C}$ is a complex number. Redraw the uncertain closed-loop system (ie., negative-feedback interconnection of $P + \Delta$ and C) as a positive feedback loop consisting of Δ and some system G (which depends only on P and C). Determine G.

   (c) Use the commands `feedback`, `norm`, `freqresp`, and basic arithmetic (eg., multiplication, addition, division) to find the smallest (in absolute value) value of Δ such that the feedback connection of $P + \Delta$ and C is unstable? Verify the instability using `feedback` and `pole`. **Hint:** Refer to slides 3-7 in the "WorstCaseUncertainSystemAnalysis" powerpoint file.

   (d) Duplicate the results in part 4c using a `ucomplex` uncertain element, and the commands `feedback` and `robuststab` (or `robstab`). Explicitly show that the results are the same. Explain in 2 sentences how the first two output arguments from `robuststab` (or `robstab`) confirm your own calculation in part (4c).

(e) With some of the results from part 4c, use `cnum2sys` to find a stable transfer function $\hat{\Delta}$ such that $\left\|\hat{\Delta}\right\|_\infty = |\Delta|$ (where Δ was obtained part 4c) such that the feedback connection of $P + \hat{\Delta}$ and C is unstable? Verify the instability using `feedback` and `pole`.

(f) Duplicate the results in part 4e using a `ultidyn` uncertain element, and the commands `feedback` and `robuststab` (or `robstab`). Explicitly show that the results are the same. Explain in 2 sentences how the first two output arguments from `robuststab` confirm your own calculation in part (4e). **Note:** Recall we established that for robust-stability questions, constant complex-valued uncertainty is equivalent to linear dynamic-system uncertainty.

(g) Now consider multiplicative perturbations to P of the form $P(1 + \Delta)$, where $\Delta \in \mathbb{C}$ is a complex number. Redraw the uncertain closed-loop system (ie., negative-feedback interconnection of $P(1+\Delta)$ and C) as a positive feedback loop consisting of Δ and some system G (which again depends only on P and C, and is different from that in (4b)). Determine G.

(h) Use the commands `feedback`, `norm`, `freqresp`, and basic arithmetic (eg., multiplication, addition, division) to find the smallest (in absolute value) value of Δ such that the feedback connection of $P(1+\Delta)$ and C is unstable? Verify the instability using `feedback` and `pole`.

(i) Duplicate these results of part (4h) using a `ucomplex` (or `ultidyn`) uncertain element, and the commands `feedback` and `robuststab` (or `robstab`).

5. Consider the same nominal plant and controller data as in the previous problem

$$P(s) = \frac{1}{s-1}, \qquad C(s) = \frac{5.8s + 9}{s(0.04s + 1)}$$

In this problem we consider the response to output disturbances, so we will focus on the sensitivity function $S := \frac{1}{1+PC}$, and variations of that due to uncertainty in the plant behavior.

(a) Define

$$W_p := \frac{0.667s + 3}{s + 0.003}$$

Show that $\|W_p S\|_\infty \leq 1$, using the commands `feedback` and `norm`.

(b) Make a Bode magnitude (magnitude only) plot of S and $\frac{1}{W_p}$ (on the same axis) to confirm that $|S(j\omega)| \leq \frac{1}{|W_p(j\omega)|}$ for all $\omega \in \mathbb{R}$.

(c) Consider 40% multiplicative (ie., percentage) uncertainty in P, modeled as $P_u := P(1 + 0.4\Delta)$, where Δ is any stable linear system, with $\|\Delta\|_\infty \leq 1$. Based on results from 4g, can you confirm that for all such P_u, the feedback connection of C and P_u is stable? Why?

(d) Again, consider 40% multiplicative (ie., percentage) uncertainty in P, modeled as $P_u := P(1 + 0.4\Delta)$, where Δ is any stable linear system, with $\|\Delta\|_\infty \leq 1$. Using commands `ultidyn`, `feedback` and `wcgain`, find the value of

$$\max_{\text{allowable } \Delta} \left\| \frac{W_p}{1 + P_u C} \right\|_\infty$$

and a specific stable linear system $\bar{\Delta}$ that achieves this worst-case performance.

(e) Plot the Bode-magnitude response of S, for both the nominal plant P, and the "worst-case" P_u on the same axis. Include a legend, and $\frac{1}{W_p}$ as well. Does the plot confirm the worst-case gain computed in part (5c)?

(f) Plot the step response of S, for both the nominal plant P, and the "worst-case" P_u on the same axis. Include a legend. Make the final time $T_F = 4$

(g) Suppose the uncertainty model is not a constant 40% level at all frequencies, but rather it starts at 40%, and grows to 100% at $\omega = 20$, and continues to grow to 400%. This can be modeled as

$$P_u = P(1 + W_u \Delta)$$

where $W_u = \frac{4s+33.8}{s+84.5}$ (using `makeweight(0.4, 20, 400)`) and Δ is any stable linear system, with $\|\Delta\|_\infty \leq 1$.

- **Task 1**: Is the closed-loop system robust to this uncertainty model?
- **Task 2**: If the closed-loop is robustly stable, use `wcgain` to compute the worst-case gain degradation for $\left\| \frac{W_p}{1+P_u C} \right\|_\infty$.
- **Task 3:** Plot the step response of S, for both the nominal plant P, and the "worst-case" P_u on the same axis. Include a legend. Make the final time $T_F = 4$