Models For Rover System Identification

Sean Vaskov

This document contains models I have tried for rover system identification. It will be divided into 4 sections as follows:

- **Lateral Models:** Section contains models to determine yaw and heading. These models will be decoupled from longitudinal velocity, in other words, the inputs will be longitudinal velocity and steering input and I will focus on fitting these to trials where the rover is turning.
- **Longitudinal Models:** Section contains models to determine longitudinal velocity. These models will be decoupled from longitudinal velocity, in other words, the only input will be the throttle input and I will try fitting these to trials where the rover is moving straight.
- **Full Models:** Section contains models to with both longitudinal and lateral velocity. These models will have throttle and steering commands as the inputs and I will try fitting these to all types of trials.
- **Misc Data/Models:** Section contains models and data to aid in creating models in aforementioned settings. For example: models relating steering input to steering angle or tire force models

1 Lateral Models

Lateral Models will use the following symbols:

States/Inputs		
Symbol	Meaning	Units
x	x position	m
y	y position	m
s	distance	m
ψ	heading	rad
β	sideslip angle	rad
v	velocity	$m\,s^{-1}$
v_x	longitudinal velocity	$m\,s^{-1}$
v_y	lateral velocity	$m\,s^{-1}$
ω	yaw rate	$rad\,s^{-1}$
δ	steering angle	rad

Parameters		
Symbol	Meaning	Units
m	vehicle mass	kg
I_z	moment of inertia for yaw	$kg\,m^2$
l_f	distance from front wheel to center of mass	m
l	wheelbase	m
c_a	cornering stiffness of front tire	$N\,rad^{-1}$
pr	ratio of rear:front tire cornering stiffness	

1.1 Kinematic Bicycle Model

Model is a variation of the kinematic model found in [1, pg. 26]. The rear steering angle is set to 0 for our model. Note the distance from the center of mass to the rear wheel is expressed by $l - l_f$. I did this because l is a measurable quantity and so we can build the constraint that $l_f + l_r = l$ in the model. The inputs I used are v and δ. Note the constraints in this model are based on constant v, which is not true for our trials, so I am not surprised that it didn't fit well.

$$\begin{aligned} \dot{x} &= v\cos(\psi+\beta) \\ \dot{y} &= v\sin(\psi+\beta) \\ \dot{\psi} &= \frac{v\cos(\beta)}{l}\tan(\delta) \end{aligned} \tag{1}$$

where: $\beta = \arctan\left(\frac{(l-l_f)\tan(\delta)}{l}\right)$ The matlab function I have saved for this is 'kinematicbicycle.m'

1.2 Dynamic Model Based on Yaw Rate and Sideslip

Model is found in [1, pg.38-39]. It is a dynamic model for sideslip and yaw rate and allows us to incorporate some basic tire dynamics. The model I'm presenting here adds position via kinematic constraints, similar to those in model 1 and the notion that $v\cos(\beta) = v_x$. It also sets the bank angle to 0. The inputs are v_x and δ

$$\begin{aligned} \dot{x} &= \frac{v_x}{\cos(\beta)}\cos(\psi+\beta) \\ \dot{y} &= \frac{v_x}{\cos(\beta)}\sin(\psi+\beta) \\ \dot{\psi} &= \omega \\ \dot{\beta} &= -\omega + \frac{c_a}{m\,v_x}\left(\delta - \beta - \frac{l_f\,\omega}{v_x}\right) + \frac{pr\,c_a}{m\,v_x}\left(-\beta + \frac{(l-l_f)\,\omega}{v_x}\right) \\ \dot{\omega} &= \frac{l_f\,c_a}{I_z}\left(\delta - \beta - \frac{l_f\,\omega}{v_x}\right) - \frac{(l-l_f)\,pr\,c_a}{I_z}\left(-\beta + \frac{(l-l_f)\,\omega}{v_x}\right) \end{aligned} \tag{2}$$

Note that the front tire cornering stiffness is given by $pr\,c_a$ this enables me to constrain the solver to make the two tire stiffnesses close or equal to each other if desired. also note that this model assumes small angle approximations for the velocity angles of the front and rear tires. I think this simplification has lead to erratic behavior if the vehicle turns at low speeds, because as $v_x \to 0$ the system becomes unstable. The matlab function I have saved this model to is 'rajanami.m'. Note that β and ω are the only states that depend on paramters

1.3 Dynamic Model with Lateral Velocity and yaw rate

Model is found in [2]. For now we will consider v_x an input. This model uses the same kinematic constraints for x,y position as model 1 In fact, this model is the same as model 2, if we make all of the small angle assumptions about tire forces (for comparison these assumptions are in the matlab function 'lygerosraj.m').

$$\begin{aligned} \dot{x} &= v_x\cos\psi - v_y\sin\psi \\ \dot{y} &= v_x\sin\psi + v_y\cos\psi \\ \dot{\psi} &= \omega \\ \dot{v_y} &= \frac{1}{m}(F_{r\,y} + F_{f\,y}\cos\delta - m\,v_x\,\omega) \\ \dot{\omega} &= \frac{1}{I_z}(F_{f\,y}\,l_f\cos\delta - F_{r\,y}(l-l_f)) \end{aligned} \tag{3}$$

where $F_{f\,y}$ and $F_{r\,y}$ are the lateral forces generated by the front an rear tires, respectively. They can be modeled by:

$$F_{f\,y} = D_f \sin(C_f \arctan(B_f\,\alpha_f)) \qquad \text{where: } \alpha_f = \delta - \arctan(\frac{\omega l_f + v_y}{v_x})$$

$$F_{r\,y} = D_r \sin(C_r \arctan(B_r\,\alpha_r)) \qquad \text{where: } \alpha_r = \arctan(\frac{\omega(l - l_f) - v_y}{v_x})$$

where α_f and α_r are the front and read slip angles, and parameters B_f, C_f, D_f, B_r, C_r, D_r define the shape of the friction curves. For now I will be simplifying the tire forces to:

$$F_{f\,y} = c_a(\delta - \arctan(\frac{\omega l_f + v_y}{v_x})) \qquad F_{r\,y} = pr\,c_a(\arctan(\frac{\omega(l - l_f) - v_y}{v_x}))$$

where pr and c_a are defined in the parameter table at the beginning of the section. This is a similar simplification made in model 2 however I am not making small angle assumptions about the slip angles. I hope this will eliminate some of the erratic behavior I saw in the model 2 around $v_x = 0$. If I can get this to fit well great, if not, I may need a better idea of the moment of inertia or steering trials with $\omega << 1$ to fit complex tire models. In matlab this function is saved as 'lygerostan.m' .

2 Longitudinal Models

Longitudinal Models will use the following symbols:

States/Inputs		
Symbol	Meaning	Units
s	distance	m
v	velocity	$m\,s^{-1}$
V	voltage	$volts$
i	current	$amps$
ω_m	motor shaft angular velocity	$rad\,s^{-1}$
u_2	throttle	

Parameters		
Symbol	Meaning	Units
m	vehicle mass	kg
J	moment of inertia of rotor	$kg\,m^2$
b	motor friction constant	$N\,m\,s$
k_e	electromotive force constant	$volts\,rad^{-1}\,s^{-1}$
R	motor resistance	ohm
L	motor inductance	H
C_r	rolling resistance	$m\,s^{-2}$
C_d	drag coefficient	m^{-1}
r	tire radius	m

2.1 Simple input

Initially unsure of what the input for the throttle (u_2) was, I attempted to fit 2 models to the data for straight motion. The first model was a simple double integrator with damping

$$\begin{aligned}\dot{s} &= v \\ \dot{v} &= p_1\, u_2 + C_r\, v + C_r\, v^2\end{aligned} \tag{4}$$

where p_1 is an arbitrary parameter chosen by the solver. This model did not fit well, attempting to fit the zero input "coast down" to provide tighter parameter bounds did not work.

2.2 Polynomial Model

The second model included polynomial terms up to degree 2 and was given by

$$\begin{aligned}\dot{s} &= v \\ \dot{v} &= \sum i = 0^2 \sum j = 0^2 p_{i\,j}\, {u_2}^i\, v^j\end{aligned} \tag{5}$$

where $p_{i\,j}$ were parameters chosen by the solver.

This model fit well for the data it was trained on, but would give unstable results for other inputs. I tried enforcing negativity on parameters that should have been negative, and while that may have helped, it did not eliminate the problem. I tried spoofing trials at different positions and times to reduce the over fitting. There is a lack of variation in inputs in the experimental data created. In order to use this model I think we need to do extensive motion capture data collection on a wide variety of throttle inputs.

2.3 RC Model

This model was for the longitidunal velocity and was presented in [2].

$$\begin{aligned}\dot{s} &= v \\ \dot{v} &= \frac{1}{m}(F_{r\,x} - F_{f\,y} \sin\delta + m\, v_y \omega)\end{aligned} \tag{6}$$

Where: $F_{r\,x} = (C_{m1} - C_{m2} v_x)\, d - C_r - C_d\, v_x^2$ and $C_{m1} and C_{m2}$ are constants for the motor. Note for the straight trials $\delta \equiv \omega \equiv 0$. I tried this model with u_2 as the input (d) and had similar issues to model 4, in terms of it not being able to handle the decellaration. This model is either insufficient or u_2 cannot be feed directly as the input.

2.4 Electric Motor Model

I found following model for an electric motor at [**?**].

$$\begin{aligned}\dot{\omega_m} &= \frac{-b}{J}\omega_m + \frac{K}{J}\, i \\ \dot{i} &= \frac{-K}{L}\omega_m + \frac{-R}{L}\, i + \frac{1}{L}\, V\end{aligned} \tag{7}$$

Quickly looking at this model with rwos ($v = r\omega_m$) the shape of the curve matched better than models 5 and 4. Particularly it correctly showed the behavior where the throttle input was positive, but decreasing, the speed decreased. This confirms my suspicion the motor is speed controlled. Next step for this model will be to incorporate elements of the physical vehicle into it, such as drivetrain, tire forces, drag, etc.

2.5 Generic Controller Models

I am looking at estimations using models for controllers. The first model I tried is and was inspired by a model for a controlled dc motor.

$$\begin{aligned} \dot{s} &= v \\ \dot{v} &= \frac{-1}{\tau} v + \frac{k}{\tau} u_2 \end{aligned} \tag{8}$$

Note τ and k are arbitrary time and gain constants. input here is the channel input-1500. Model is saved as 'gain.m'

I ran the optimizer on trials 2,4,7,9,10. Got a local minimum with cost as 1e2, feasibility 5.4e-11 optimality 1e-7 parameters p=[0.779076823232103;0.00785413996337274]. Error on some of the ones I looked at was as high as .5 m in x direction (simulation overshoot)

Maybe I will try other stuff in matlabs sys id toolbox

2.6 Cruise Control Models

I investigated some models for cruise control to try and translate the throttle input to a longitudinal velocity

2.6.1 PI Control

This model is generic PI control for cruise control.

$$\begin{aligned} \dot{s} &= v \\ \dot{v} &= -a - b\,v - c\,v^2 + u \end{aligned}$$

where $u = k_p(ps\,u_2 - v) + k_i \int_0^t (ps\,u_2 - v)d\tau$ and ps is a scaling factor to relate the channel input to a desired speed

as a linear system this is represented as the following set of equations (saved as 'picruise.m')

$$\begin{aligned} \dot{s} &= v \\ \dot{v} &= -a - b\,v - c\,v^2 + k_p(ps\,u_2 - v) + k_i\,e \\ \dot{e} &= ps\,u_2 - v \end{aligned} \tag{9}$$

When combined, the terms a and $c\,v^2$ represent the rolling resistance and drag. At this point I am leaving them as generic parameters; this way, the solver will easily fit acceleration and "steady state" conditions. However there is some error when the vehicle slows down significantly and huge error when the input is instantaneously removed

3 Full Models

4 Misc Data/Models

References

[1] Rajesh Rajamani. *Vehicle Dynamics and Control*. Springer US.

[2] Alexander Liniger, Alexander Domahidi, and Manfred Morari. Optimization-based autonomous racing of 1:43 scale RC cars. *Optimal Control Applications and Methods*, 36(5):628–647, 2015.