

# 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	$m$
$\psi$	heading	$rad$
$\beta$	sideslip angle	$rad$
$v$	velocity	$m\,s^{-1}$
$v_x$	longitudinal velocity	$m\,s^{-1}$
$v_y$	lateral velocity	$m\,s^{-1}$
$\omega$	yaw rate	$rad\,s^{-1}$
$\delta$	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  $\delta$ . 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  $\delta$

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

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  $v_x \rightarrow 0$  the system becomes unstable. The matlab function I have saved this model to is 'rajanami.m'. Note that  $\beta$  and  $\omega$  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').

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

where  $F_{fy}$  and  $F_{ry}$  are the lateral forces generated by the front and rear tires, respectively. They can be modeled by:

$$F_{fy} = D_f \sin(C_f \arctan(B_f \alpha_f)) \quad \text{where: } \alpha_f = \delta - \arctan\left(\frac{\omega l_f + v_y}{v_x}\right)$$

$$F_{ry} = D_r \sin(C_r \arctan(B_r \alpha_r)) \quad \text{where: } \alpha_r = \arctan\left(\frac{\omega(l - l_f) - v_y}{v_x}\right)$$

where  $\alpha_f$  and  $\alpha_r$  are the front and rear 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:

$$F_{fy} = c_a(\delta - \arctan\left(\frac{\omega l_f + v_y}{v_x}\right)) \quad F_{ry} = pr c_a(\arctan\left(\frac{\omega(l - l_f) - v_y}{v_x}\right))$$

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  $\omega \ll 1$  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	$amps$
$\omega_m$	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

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

where  $p_{ij}$  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 longitudinal velocity and was presented in [2].

$$\begin{aligned}\dot{s} &= v \\ \dot{v} &= \frac{1}{m}(F_{rx} - F_{fy} \sin \delta + m v_y \omega)\end{aligned}\tag{6}$$

Where:  $F_{rx} = (C_{m1} - C_{m2}v_x)d - C_r - C_d v_x^2$  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 deceleration. 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  $r\omega$  ( $v = r\omega_m$ ) 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 Transfer Function**

I may want to look at empirically deriving a transfer function between throttle input and speed using Matlab's system id toolbox?

## **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.