Car setup

This document explains how to setup an MdtCar based on the parameters in the CarData structure. See also the GreaseMonkey pdf file included with the release documentation.

The most complex parts of initialising an MdtCar are suspension and wheel/ground contact setup, but before looking at these it is useful to understand the frames involved in specifying the layout of the car.

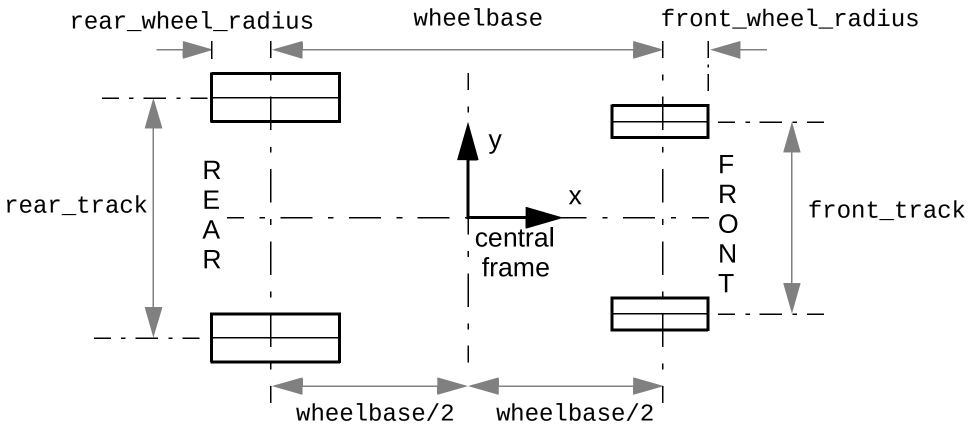

figure 1: *top view of car showing central chassis frame*

Figure 1 shows a plan view of the car, which is assumed to be symmetrical left-right. A central chassis reference frame is imagined on the symmetry axis, centred between front and rear axles at ground level. The *x* and *y* axes have been shown, corresponding to the forward and left directions. This central frame is purely notional for defining the placement of actual frames for graphics/collision and physical properties of the chassis body, see figures 2 and 3.

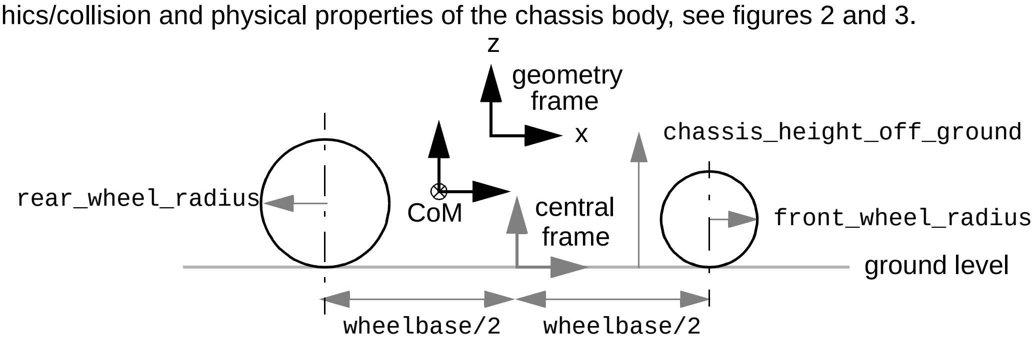

figure 2: *side view of car showing geometry frame and CoM frame*

The *z* axis points vertically up as shown in side view, figure 2. Note that these axis conventions need not correspond to graphics conventions, which often have the *y* axis vertical and/or use a left-handed frame.

The geometry frame acts as the graphics centre and currently also doubles as the centre for the collision geometry. The CoM or ‘centre-of-mass’ frame is the inertial centre of the physical body. The following explanation describes the data fields defined in the `CarData.h` header file.

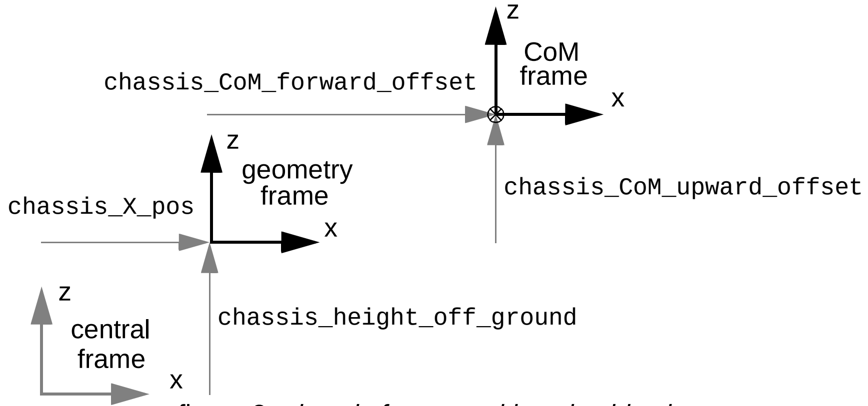

figure 3: *chassis frame positions in side view*

The geometry frame is `chassis_height_off_ground` units above the ground-level central frame and `chassis_X_pos` units in front of it, figure 3. (the `chassis_forward_offset` data field was meant for the same purpose as the `chassis_X_pos` field, but was not used).

The centre-of-mass frame is `chassis_CoM_upward_offset` units above the geometry frame and `chassis_CoM_forward_offset` units in front of it. Changing these fields allows the centre of mass of the chassis to be moved, to change the balance and handling of the car. Note that the centre of mass position is specified relative to the geometry frame, so moving the geometry frame moves the mass properties with it. In order to place the centre of mass directly above the central frame, the `chassis_CoM_forward_offset` should be set to minus `chassis_X_pos`.

Note that the `chassis_X_pos`, `chassis_CoM_forward_offset` and `chassis_CoM_upward_offset` will often be negative, as they are shown in figure 2.

The inertial properties of the chassis body are described by `chassis_mass, chassis_MoI_xx, chassis_MoI_yy, chassis_MoI_zz`. Similarly, for the wheels there is `wheel_mass, wheel_MoI_xx, wheel_MoI_yy, wheel_MoI_zz`. Note that, at present, the three moments of inertia along the principal axes are internally averaged to a single value so setting them to different values will have no effect. This guarantees good stability compared to ‘non spherical’ inertias. As usual, the `mass` and `MoI` values should be set consistent with the size and mass distribution of the bodies.

The code sets up a simple box collision geometry for the chassis, centred on the geometry frame. The collision box has x,y,z dimensions (length, width, height) specified in the `chassis_coll_box` array. (Offsetting the collision geometry requires a small change to the code, see the use of McdModelSetRelativeTransformPtrs and shape_offsets in MdtCar.c).

The wheels are given spherical collision geometries with radii specified by the wheel radii.

Suspension setup

When a car is at rest on level ground, or ‘statically loaded’, the suspension displacement of a particular wheel from its zero-load or ‘natural’ length depends on the static load it carries and its suspension spring rate. The static load that a wheel carries depends on the positions of the wheels relative to the chassis centre of mass. If the CoM is mid way between front and rear axles, for example, then the load is equally shared front and rear. Figure 4 shows a chassis with CoM closer to the rear than the front, which places a higher proportion of the load on the rear wheels.

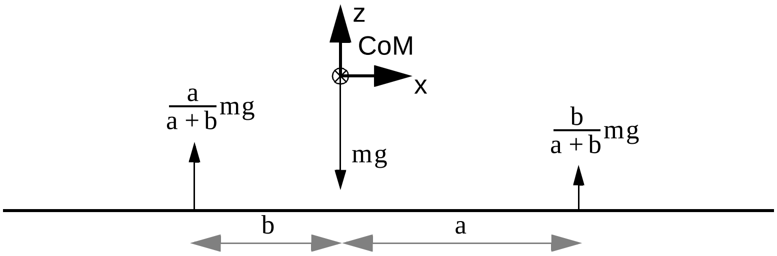

figure 4: *chassis load distribution*

Setting up suspension by directly tweaking spring and damping is tricky and the parameters would have to be re-tuned for any changes to the vehicle. The approach used here calculates the static distribution of wheel loads automatically from the chassis mass, CoM position and wheel positions and uses parameters which are either dimensionless or have units of length for ease of scaling.

With static wheel loads known, the front and rear suspension stiffnesses are specified by giving the displacements of the front and rear wheels from zero-load to twice their static load. Damping is set as a damping ratio so that a value of 1 gives critical damping. Two parameters specify the suspension extension and compression limits; (1) the total suspension travel between bottom stop and top stop limits and (2) a proportion which gives a point in the suspension travel, measured from the bottom stop, corresponding to the resting suspension displacement at static loading.

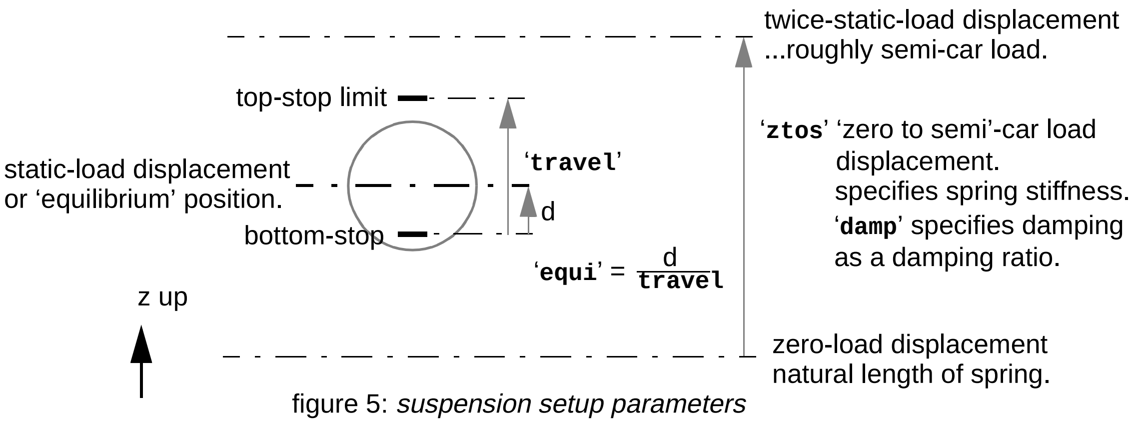

figure 5: *suspension setup parameters*

The wheel suspension heights are not specified directly. Rather, the heights are calculated so that, at the static loads and spring rates calculated above, the car will sit level with suspensions vertical and the chassis at its specified height off the ground. Actually, because of the approximation used to implement stiff-spring suspension, the calculation in not exact and the car may not sit quite level at precisely the specified height so a ‘tweak’ parameter is provided to correct this if necessary, `suspension_level_tweak`, which adds a forward offset as a proportion of the wheelbase.

The suspension ‘`ztos`’ parameters `rear_suspension_ztos, front_suspension_ztos` specify the spring extension resulting from a change in load from zero load to twice the static load. This represents the effective working range of the suspension; when the car is accelerated to the point of lifting the front wheels the front suspensions experience zero load and the rear suspensions each experience half the chassis load, roughly twice their static load. This ‘zero to semi’ car load displacement offers an intuitive way to set the suspension spring rates. A large `ztos` displacement parameter gives a soft suspension while a small value gives a stiff suspension.

The suspension travel parameters `rear_suspension_travel` and `front_suspension_travel` specify the full suspension travel distances between the bottom limit and the top limit. The parameters `rear_suspension_equi` and `front_suspension_equi` are proportions giving the position of the statically loaded suspension ‘equilibrium’ displacement above the bottom stop as a proportion of the full suspension travel. If the ‘`equi`’ parameter is set to 0.5, for instance, the bottom stop and top stop limits will be equally distances from the static resting displacement.