Control modelling the MicroMouse

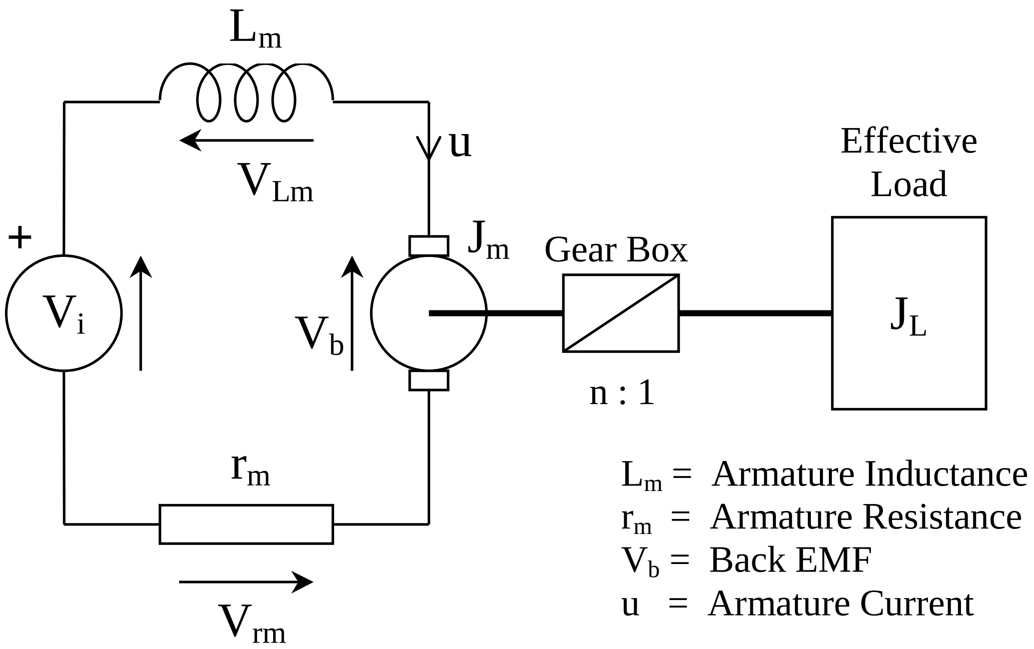

Figure 1: The Drive (One motor, gearbox and load)

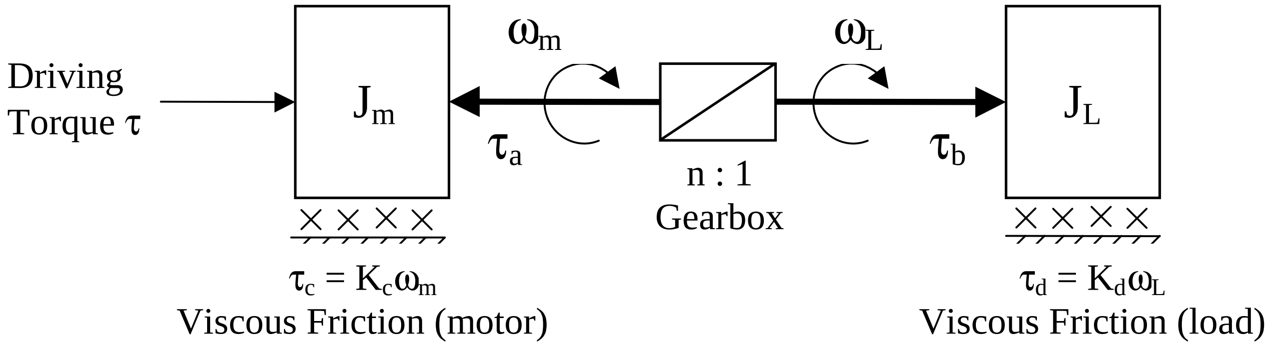

Figure 2: Torque, speed and friction

The equations that describe the system may be written down by inspection.

From Figure 1:

$$V_i = V_{Lm} + V_b + V_{rm} \qquad V_{Lm} = L_m\dot{u} \qquad V_b = K_b\omega_m \qquad V_{rm} = ur_m$$

Hence

$$V_{Lm} = V_i - V_b - V_{rm} \qquad L_m\dot{u} = V_i - K_b\omega_m - ur_m$$

From Figure 2:

$$\tau = K_t u$$

$$\tau_m = J_m \dot{\omega}_m = \tau - \tau_a - \tau_c$$ - Armature

$$\tau_a = \frac{\tau_b}{n} \qquad \omega_L = \frac{\omega_m}{n}$$ - Gearbox

$$\tau_c = K_c \omega_m$$ - Viscous Friction (motor)

$$\tau_L = J_L \dot{\omega}_L = \tau_b - \tau_d$$ - Load

$$\tau_d = K_d \omega_L$$ - Viscous Friction (load)

These equations now describe the block diagram, Figure 3.

Any attempt at intelligent control depends on the knowledge of all the parameters above. Some of them can be obtained from the chassis manufacturer, but others will have to be estimated by experiment. The table below summarizes the known and unknown values.

Parameter	Value
n	16
r_m	2.1 Ω
L_m	1.2 mH
K_b	0.75 mV / rpm
K_t	61 gm.cm / A
K_c	?
K_d	?
J_m	?
J_L	?

Table 1 Drive Parameters

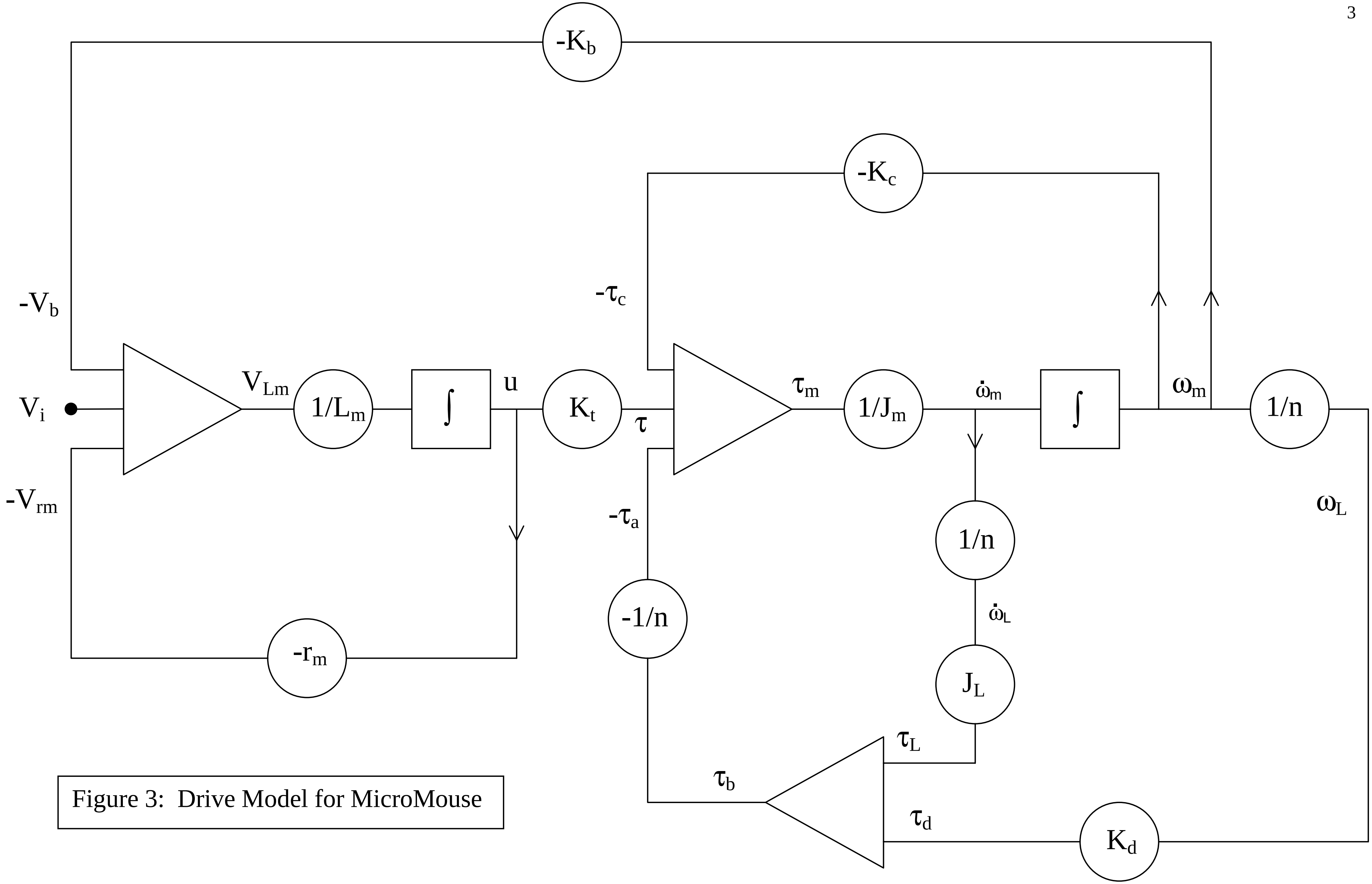

Figure 3: Drive Model for MicroMouse

Calculation of V_I

The driving voltage is the average DC value of the PWM signal provided by the microcontroller chip. This is a variable mark-space ratio pulse train of Period T and Pulse Width t. Using a PCA timer clock of $(f_{OSC})/12$ yields a PWM pulse frequency of 3600 Hz.

$$V_i = \frac{t}{T} V_{BAT} = \left(\frac{256 - P}{256} \right) V_{BAT} \qquad \text{where } 0 \le P \le 255$$

P is the 8-bit value placed in the PCA Capture register to set the speed. The value for V_{BAT} should be measured across the motor while running the mouse off load (see later) with a value of P = 0.

The Gearbox

Because of the type of gearbox employed (worm and spur gear), we must avoid the situation, when slowing down, in which the load is driving the motor. This causes the gearbox to lock up and the mouse to skid.

In Fig. 2 it is seen that τ_a is so defined as to be positive when opposing the driving torque: when therefore τ_a becomes negative, it is driving the motor and this is the condition that must be avoided.

From Fig. 3, τ_b (and therefore τ_a) remains positive while $\tau_L + \tau_d > 0$

i.e. $\dot{\omega}_m \left(\frac{1}{n} \right) J_L + \omega_m \left(\frac{1}{n} \right) K_d > 0$

Thus to avoid lock up:

$$\dot{\omega}_m > -\omega_m \left(\frac{K_d}{J_L} \right)$$

Note at the limit:

$\dot{\omega}_m = -\omega_m \left(\frac{K_d}{J_L} \right)$ and this equation can be solved to give:

$$\omega_m = \omega_m(0) e^{-Kt} \quad \text{where } K = \frac{K_d}{J_L}$$

So ω_m follows an exponential decline from its running speed to zero when decelerating at the maximum rate.

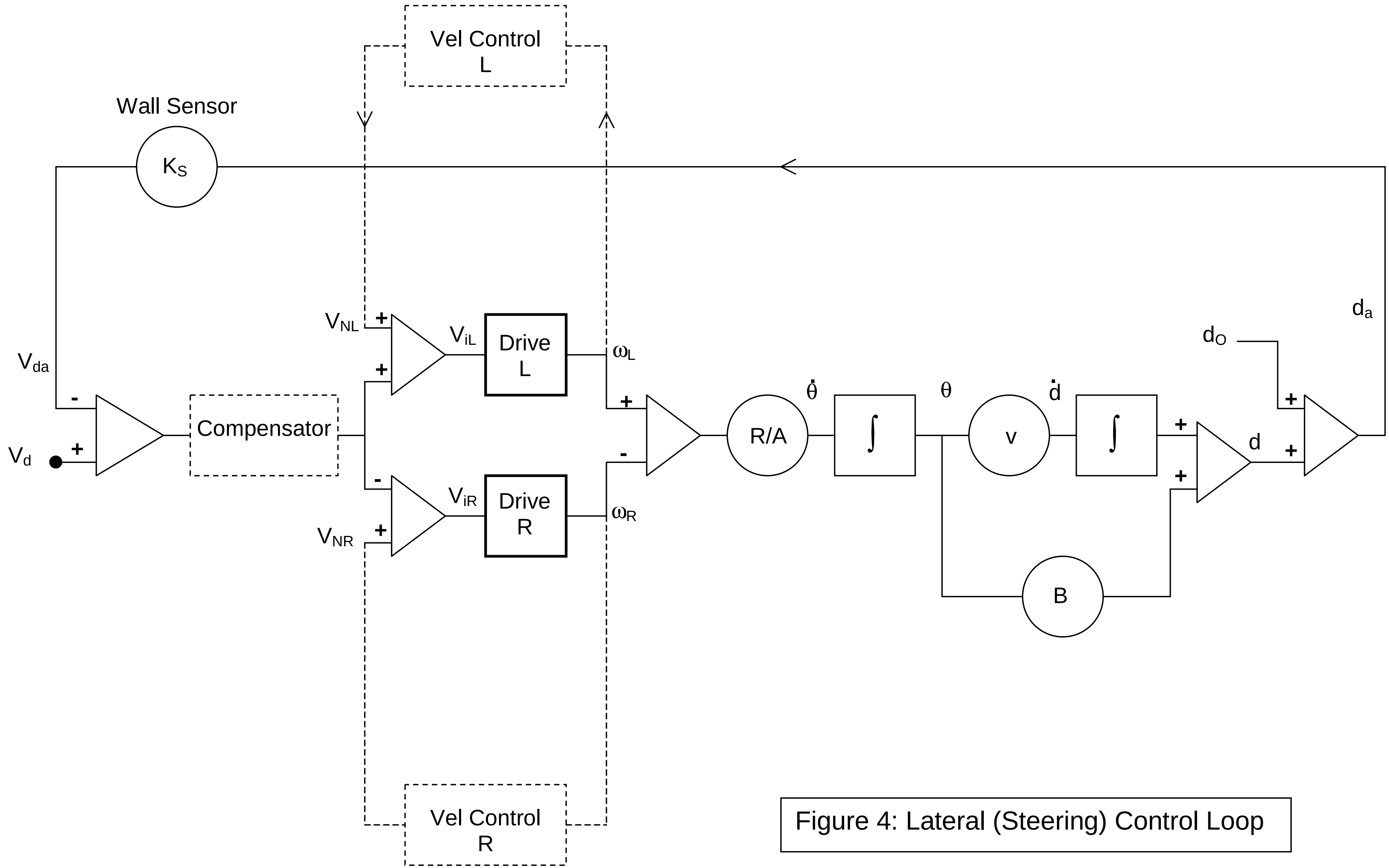

Figure 4: Lateral (Steering) Control Loop

Steering Control Loop

The mouse is steered by slowing down a motor on one side of the chassis. It is assumed that there is a sideways-looking sensor on each side of the mouse providing a measure of the distance between it and the maze wall. If there is no steering input from the maze solving algorithm, then this control loop attempts to keep the mouse moving straight ahead down the centre of the ‘corridor’. This presents a processing problem in that the sensors provide a lateral position error, but the correcting factor applied to the motors is a variation in the wheel rotational speed! The part of the loop (Fig. 4) to the right of the drives performs this conversion.

Some of the constants and variables shown in Fig.4 are determined by the physical layout of the mouse chassis (Fig.5).

V_d = an offset voltage (= 0 if running up the centre).
K_S = Sensor constant (V/m).
V_{NL}, V_{NR} = nominal driving voltages for motors.
B = distance sensors ahead of axles.
R = wheel radius.
A = wheel track.
v = nominal forward speed.
θ = heading angle.
d = lateral drift error.
d_a = wall clearance.
d_O = required clearance.

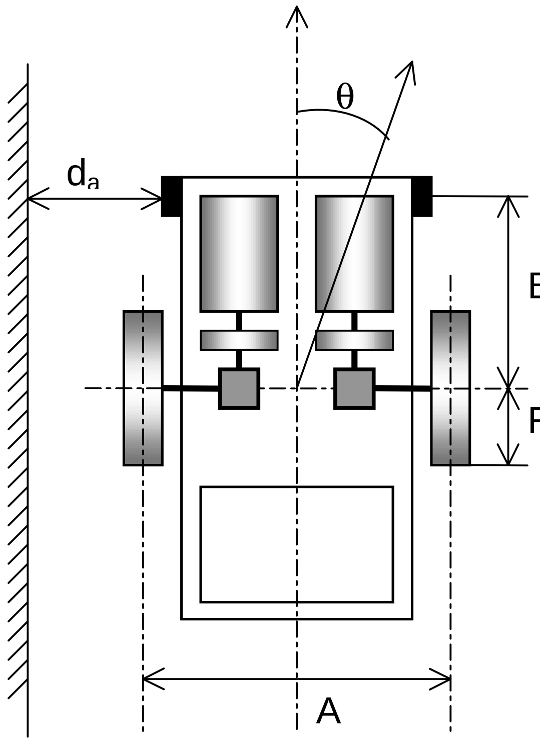

Figure 5: Chassis geometry

Note: positive d $\rightarrow$ rightwards, positive $\theta \rightarrow$ clockwise. That is, the left-hand wall is taken as the datum.

This loop is entirely unstable without compensation! If operated at low speed, the compensation should be simple however, so this is the first objective.

Operation at high speed needs complex compensation and would require more work.

Evaluating the Unknown Drive Parameters

K_C : Run motor at constant speed without load J_L, recording u and ω_m. The mouse is 'jacked up' off its wheels for this test. Use a multimeter set to DC current range, in the power supply lead to one of the motor control circuits. Run Test Program 1 in **MOUSEMON** by typing 01XXT where XX is a speed setting parameter. Use the R command to read the actual motor speed C (a 16-bit number in registers 00 and 01). ω_m is in rads/sec, but the value captured C, is a count of PCA timer clock pulses. The conversion is as follows: The PCA is clocked at $(f_{OSC})/12$ = 921.6 kHz. A speed 'capture' is performed after 4 pulses from the tachometer, or half a shaft revolution. Hence shaft speed is:

$$s = \frac{27648000}{C} \text{ rpm} \quad \text{and} \quad \omega_m = \frac{2895292}{C} \text{ rads/sec}$$ where C is a decimal number

$$K_C = \frac{uK_t}{\omega_m}$$ from Figure 3.

K_d : As above but now under load. Run Test Program 2 in **MOUSEMON** by typing: 02XXT. The mouse will run a short distance and then stop. Use the actual maze floor for this test, because a desk or carpet surface will give different results (different friction). Measure the current while it is moving, and then once the mouse has stopped, get a value for the running speed (logged in registers 00 and 01) using R command.

$$K_d = \frac{n^2 uK_t}{\omega_m} - n^2 K_C$$ from Figure 3.

Take several readings of u and ω_m at various PWM settings. Draw rough 'calibration' graphs of u against ω_m and V_i against ω_L. Remember how V_i is related to the PWM setting and that $\omega_L = \omega_m/n$. The values of u not will be very accurate, but will give you a starting point for the simulation.

J_m : Run Test Program 3 in **MOUSEMON** by typing: 0300T. The mouse will briefly run at full speed them slow down to a stop. Speed measurements (same format as above) will have been logged in an array starting at 9000H in RAM. Using a value for ω_m, a corresponding value of deceleration $\dot{\omega}_m$ derived from adjacent speed values, and the corresponding motor current u derived form the 'calibration' graph above, a value for J_m can be derived. Use a speed value well before the last 'lock up' value.

$$J_m = \frac{\tau_m}{\dot{\omega}_m}$$ from Figure 3.

J_L : The last value before gearbox 'lock up' will be the last non-zero value in the array. Use this value, and an estimate of the deceleration from the last two values to provide ω_m and $\dot{\omega}_m$ respectively. At 'lock up', $\tau_d = \tau_L$ and thus $\tau_b = 0$. Hence:

$$J_L = -\frac{K_d \omega_m}{\dot{\omega}_m}$$ from Figure 3.

Run all these tests several times to eliminate rogue results. The resulting parameters will not be accurate, but when plugged in to the simulation package they should enable you to home-in on a good model for the micromouse drive system.

Suggested Programme of Work

1. Study and understand the model. Some lectures will given during the first two weeks.
2. Evaluate the parameters K_C, K_d, J_L and J_m.
3. Model the system on MatLab/Simulink.
4. Design the Velocity control loop.
5. Implement Lateral control at low speeds.
6. Obtain braking control.
7. Derive cornering/turning algorithms.
8. Increase the speed.

W.Forsythe
W.G.Marshall
January 2000