Computer Engineering and Mechatronics MMME3085

Lab 2 Coursework

1.

As the proportional gain was increased, the motor fluctuated with larger amplitudes around the desired position. Since the proportional gain increases the distance the motor moves based on the error, it improves the time taken for the system to reach steady state. However, if too high the large fluctuations will cause the settling time to be slower. Therefore, the ideal value of proportional gain will result in a fast rise time and there will be little fluctuations so that a fast settling time can be achieved.

2.

The goal of derivative control is to reduce the settling time. It acts as 'damping' to reduce the oscillation. This is useful in conjunction with the proportional gain as it reduces the overshoot (fluctuations) while keeping the benefit of the proportional gain (faster rise time). Increasing the gain to a value of 0.02 produced undesired results as the motor oscillated around the setpoint without ever settling.

3.

The complex operations in the main loop are only calculated when a new value is entered by the user. This means that Arduino only has to calculate them once and not during the movement of the motor. These calculations are done before the motor even starts moving, meaning that the time taken to calculate is not time sensitive and will not cause the stepper motor to skip steps and be unsynchronised.

4.

By contrast the function computeNewSpeed() is called multiple times during the operation of the motor. Complex calculations here will result in the Arduino not being able to calculate the values before the function needs to be called again. This will result in the motor skipping some steps and becoming unsynchronised.

5.

When using timed loops, the ramping of the motor was slow. This is due to the microprocessor having to calculate and compare values of the timer during the run time of the motor. Hardware timers are used to counteract this issue.

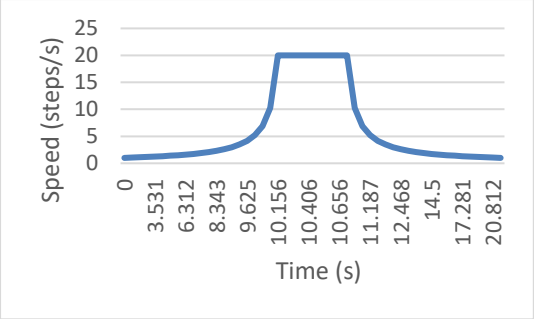

Figure 1 – Ramping profile using timed loops

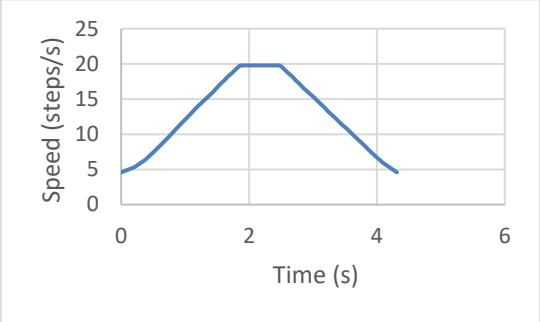

Figure 2 – Ramping profile using an Interrupt Service Routine (ISR)

A timer in CTC mode is used to generate the frequency of steps. In this mode the timer counts up to a specific value and when the values are equal, an Interrupt Service Routine (ISR) is called, and a step pulse is sent out. The timer is then set back to 0. The specific value it is comparing to is the output compare register OCR1A. This value acts as an interval p that is overwritten every time the ISR is called. The ISR is triggered after p ticks.

The if statements in the ISR are used to change the prescaler values. This is done to keep the OCR1A as large as possible within a desired range. This is done to take advantage of the full range of possible step rates.

6.

When the motor is run too fast the motor becomes desynchronised, so the position of the motor is no longer accurate. If the motor skips a step the motor can stall. To reach high velocities the acceleration must also be high, which is what causes desynchronisation. If the acceleration is low enough to avoid desynchronisation, the velocity will ramp slowly and the average velocity will be lower than before, negating the purpose to increase speed.