1.MIT Mode Operation Description

The following diagram shows the control block diagram of the motor MIT mode.

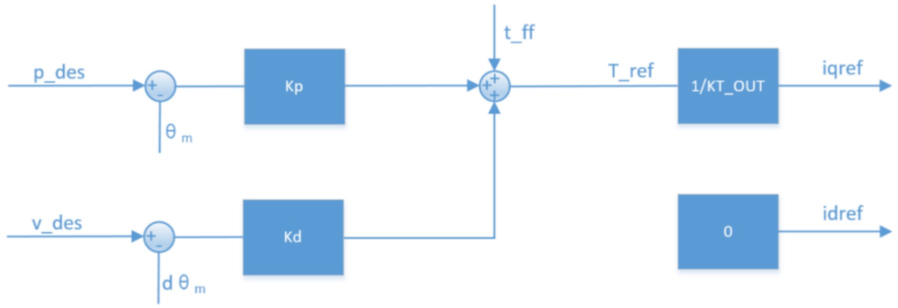


Figure1 Control Block Diagram of the MIT Mode

The MIT mode enables mixed control of torque, position, and speed. In the figure above, the position loop and speed loop are arranged in parallel. The outputs of the position and speed loops are summed with the feedforward torque t_ff to produce the reference torque T_ref :

$$T\_ref = kp*(p\_des - \theta_m) + kd*(v\_des - d\theta_m) + t\_ff$$

Where:

T_ref is the reference torque, expressed in N·m.
(2)

kp denotes the position gain, and kd denotes the speed gain.

p_des represents the desired position of the motor output shaft , expressed in radians (rad).

θ_m represents the current position of the motor output shaft , expressed in radians (rad).

v_des is the desired speed of the motor output shaft , expressed in rad/s .

dθ_m is the current speed of the motor output shaft , expressed in rad/s .

The reference torque T_ref is converted via KT_OUT to obtain the reference current iqref , which then feeds into the subsequent current PI controller.

Where:

$$iqref = T\_ref / KT\_OUT$$

$$KT\_OUT = Kt * GR$$

$$Kt = 1.5 * NPP * flux$$

iqref is the reference current, expressed in A .

GR is the motor gear reduction ratio.

K t is the torque constant before reduction, expressed inN·m/A.

NPP is the number of pole pairs.

flux is the magnetic flux linkage, expressed in Wb , and can be obtained from the motor parameters.

2.MIT Mode Usage Instructions

1. When kp=0 and kd≠0 , setting v_des alone enables constant speed operation. A steady-state error exists during uni-form rotational motion; furthermore, kd should not be excessively large, as an overly large kd will induce oscillations.

2. When kp=0 and kd=0 , providing t_ff alone can realize the specified torque output. In this case, the motor continuously out-puts a constant torque. However, when the motor is running without load or under light load, if the specified t_ff is too large, the motor will continue to accelerate until reaching the maximum speed, at which point the target torque t_ff is still not attained.

3. When kp≠0 and kd=0 , oscillations will occur. That is, when controlling position, kd must not be set to 0; otherwise, This will cause motor oscillation and may even result in loss of control.

4. When kp ≠ 0 and kd ≠ 0 , multiple scenarios exist; here, two cases are briefly presented.

(1) When the desired position *p_des* is constant and the desired velocity *v_des* is 0, point control can be achieved. During this process, the actual position heta_m approaches *p_des* , and the actual velocity dot{ heta}_m approaches 0 .

(2) When *p_des* is a continuously differentiable function of time, and *v_des* is the derivative of *p_des* ,position and velocity tracking can be realized, i.e., rotating the desired angle at the desired velocity.

The following is a simple example based on the DmiaoH7development board:

```
void TIM2_IRQHandler(void)
{
  /* USER CODE BEGIN TIM2_IRQn 0 */
    time=time+0.001f;
    kp=1.0f;
    kd=1.0f;
    tor_set=0.0f;
    pos_set=sin(2*3.1415926f*1.0f*time);
    vel_set=2*3.1415926f*1.0f*cos(2*3.1415926f*1.0f*time);
    mit_ctrl(&hfdcan1,        1,pos_set,      vel_set,kp,      kd,tor_set);      //MIT    mode torque command transmission

  /* USER CODE END TIM2_IRQn 0 */
    HAL_TIM_IRQHandler(&htim2);
  /* USER CODE BEGIN TIM2_IRQn 1 */

  /* USER CODE END TIM2_IRQn 1 */
}
```

As shown in the figure, control commands are sent to the DM4310 motor within the 1 ms timer interrupt function, during which the motor is in a no-load state. The desired position p_des is set as a 1 Hz sine wave with an amplitude of 1, and the desired velocity v_des is the corresponding derivative, with Kp set to 1, kd set to 1, and the feedforward torque tor_set set to 0. The results are as follows:

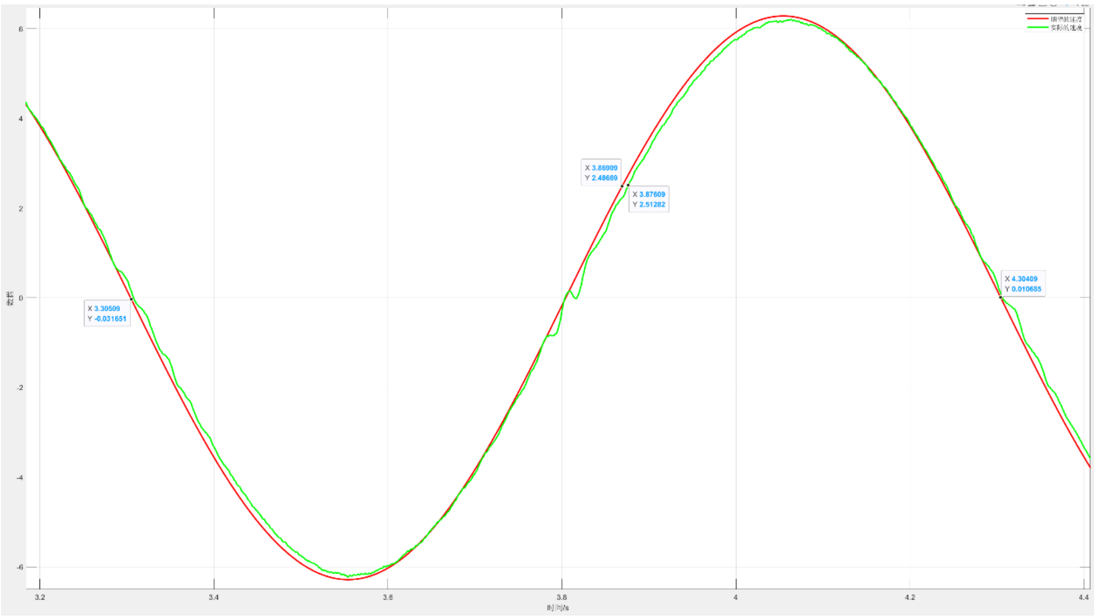
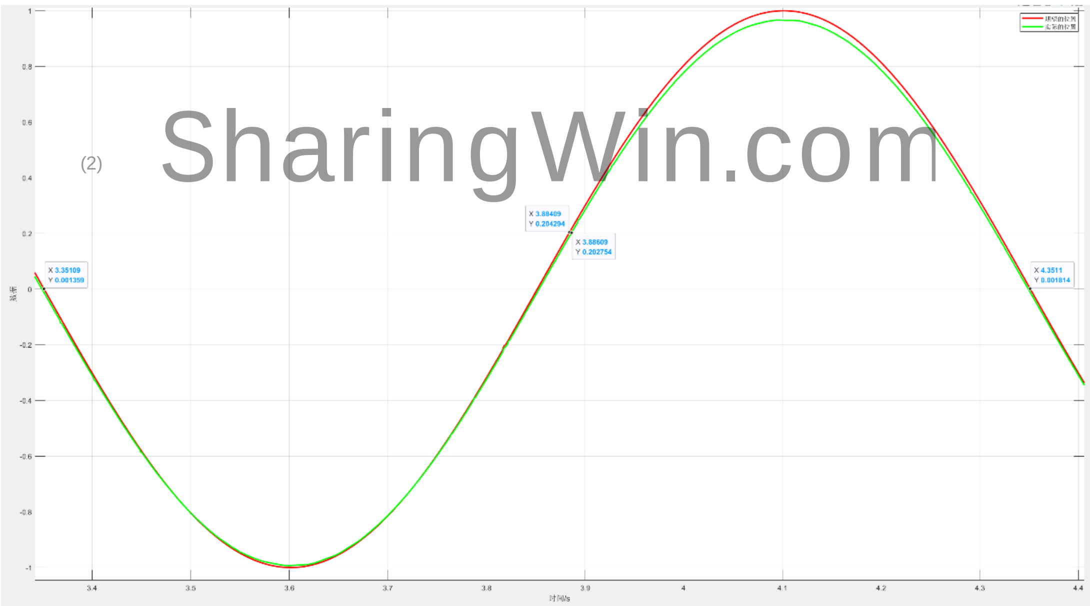


Figure 2 Speed tracking graph



(2)

Figure 3 Position tracking graph

As shown in the figure, under this condition the motor demonstrates a certain tracking capability. When the motor is under load, feedforward torque compensation must be applied.

3. Example Code

The following is the MIT control mode transmission function

```
/**
***************************************************************************
* @brief:         mit_ctrl: Motor control function in MIT mode
* @param[in]:     hcan:             Pointer to CAN_HandleTypeDef structure, used to specify the CAN bus
* @param[in]:  motor_id: Motor ID, specifying the target motor
* @param[in]:     pos:              Position setpoint
* @param[in]:     vel:              Speed setpoint
* @param[in]:     kp:               Position proportional gain
* @param[in]:     kd:               Position derivative gain
* @param[in]:     torq:             Torque setpoint
* @retval:        void
* @details: Sends control frames to the motor in MIT mode via the CAN bus.
***************************************************************************
**/
void mit_ctrl(hcan_t* hcan, uint16_t motor_id, float pos, float vel,float kp, float kd, float torq)
{
    uint8_t data[8];
    uint16_t pos_tmp,vel_tmp,kp_tmp,kd_tmp,tor_tmp;
    uint16_t id = motor_id + MIT_MODE;// MIT_MODE=0x00

    //Scale floating-point data proportionally into integers
    pos_tmp = float_to_uint(pos,  P_MIN,  P_MAX,  16);// (-12.5~12.5)
    vel_tmp = float_to_uint(vel,  V_MIN,  V_MAX,  12);// (-30.0~30.0)
    kp_tmp  = float_to_uint(kp,   KP_MIN, KP_MAX, 12);// (0.0~500.0)
    kd_tmp  = float_to_uint(kd,   KD_MIN, KD_MAX, 12);// (0.0~5.0)
    tor_tmp = float_to_uint(torq, T_MIN,  T_MAX,  12);// (-10.0~10.0)

    data[0] = (pos_tmp >> 8);
    data[1] = pos_tmp;
    data[2] = (vel_tmp >> 4);
    data[3] = ((vel_tmp&0xF)<<4)|(kp_tmp>>8);
    data[4] = kp_tmp;
    data[5] = (kd_tmp >> 4);
    data[6] = ((kd_tmp&0xF)<<4)|(tor_tmp>>8);
    data[7] = tor_tmp;

    // Transmitted to the motor driver via the CAN bus
    canx_send_data(hcan, id, data, 8);
}
```


The MIT command employs floating-point data proportionally converted into integers for transmission to the driver, which then converts the received integers back into floating-point data proportionally. This conversion requires the use of the con-version function float_to_uint , which first determines the maximum and minimum values for the two proportional conversions.

These values can be found on the upper computer parameter setting page. By default, the maximum and minimum values for KP and KD are 0.0~500.0 and 0.0~5.0 respectively. Pos , Vel , and Torque are preset to ±12.5 , ±30 , and ±10 respectively. These parameters can be adjusted according to the actual motor specifications. However, when sending control commands, it is essential to maintain consistency with the set values; otherwise, the control commands will be proportionally scaled.