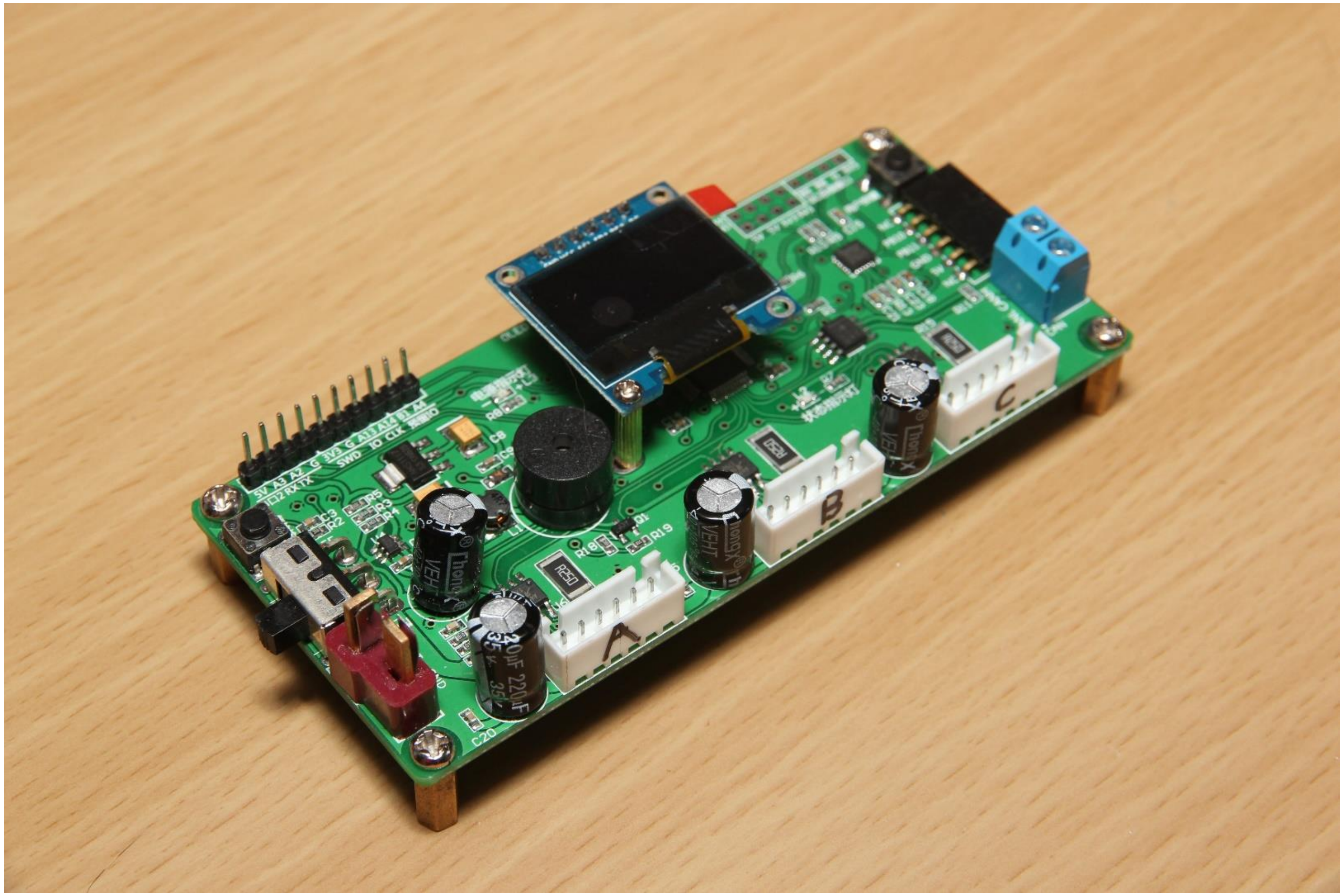
Neuron OmniBot Motor Driver Board

Communication and Operation Manual



Author: Hsin-Yi Kang (Ewing Kang)
Contact: https://github.com/EwingKang/

Ver. 0.3

Table of Contents

Operation State Machine

Initialization / Paused

- Serial Tx: 1 Hz System info (Initialization / Paused)
- Serial Rx: system state command (Ascii)
- Screen showing “system pause” information

Nominal mode

- Serial Tx:
  (a) 100Hz IMU data
  (b) 10Hz Encoder data
- Serial Rx:
  (a) system state command (Ascii)
  (b) Control Data Frame

- Screen showing general control information

IMU only mode

- Serial Tx:
  (c) 100Hz IMU data
- Serial Rx:
  (c) system state command (Ascii)
  (d) Control Data Frame

- Screen showing general control information

Serial Command (Rx)

- Serial initialization: 3 arbitrary byte
- System state command:

Function	Force start	Pause/ Un-Pause	Reset
Character	S	P	R
Hex (Ascii)	0x53	0x50	0x52
Note:			

- Control Method: Velocity control / Positional control:
  Set by the onboard switch. Please reboot to enable the different mode.

- Control data frame:

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
0xFF	0xFE	MODE	CMD_A_H	CMD_A_L	CMD_B_H	CMD_B_L	CMD_C_H	CMD_C_L

  - MODE
    0x01: Mode1/ Base Vector Control (BVC)
    0x02: Mode2/ Independent Wheel Drive (IWD)
    0x10: Emergency Stop (ES)
  - CMD_A / CMD_B / CMD_C
    - Format: short (16-bit) _H : MSB, _L : LSB [+32,767 -32,768]

    Mode1 BVC: $[u_A \quad u_B \quad u_C] \rightarrow [X \quad Y \quad \Theta]$ or $[V_x \quad V_y \quad \Omega]$, where $[X \quad Y \quad \Theta]$ is the vehicle position and $[V_x \quad V_y \quad \Omega]$ velocity command.
    Mode2 IWD: $[u_A \quad u_B \quad u_C] \rightarrow [v_A \quad v_B \quad v_C]$ direct motor velocity command

Serial Data Output (Tx)

- 1 Hz System info (only tx at Initialization / Paused)
  - Ascii string: "**Paused: *i* \n**", where *i* is an ascending uint8_t number

- 100Hz IMU data

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]
header		Accel_X		Accel_Y		Accel_Z		Gyro_X		Gyro_Y		Gyro_Z	
0xFF	0xFA	AX_H	AX_L	AY_H	AY_L	AZ_H	AZ_L	GX_H	GX_L	GY_H	GY_L	GZ_H	GZ_L

  - Length: 14 byte
  - Data: raw MPU6050 accelerometer/gyro output byte

- 25Hz Encoder data

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
0xFF	0xFB	X_DIF_H	X_DIF_L	Y_DIF_H	Y_DIF_L	TH_DIF_H	TH_DIF_L	SEQ

  - Length: 9 byte
  - Format: signed short (2 bytes, 8-bit), range [+32,767 -32,768]. _H: MSB; _L:LSB
  - `X_DIF` / `Y_DIF`: Odometry positional difference between this transmission and last transmission, in 0.1mm (10000 = 1meters)
    - Velocity is limited at ~3.2767 meters per 1/25s, or ~81.92m/s (294.9km/h).
  - `TH_DIF`: Odometry rotational difference between this transmission and last transmission, in 1/10000 radians (10000 = 1 rad.)
    - rotation at 3.2767 rad per 1/25s, or ~13 rps
  - `SEQ`: 1-byte sequence (0-255) for continuity check.

- 5Hz Data

[0]	[1]	[2]-[5]	[6]-[9]	[10]-[13]	[8]
0xFF	0xFC	X_E	Y_E	TH_E	SEQ

  - Length: 15 byte
  - Format: float (4 bytes, 32-bit). Little-endian (ex: [2] is the MSB of XE, [5] is the LSB of XE)
  - `X_E` / `Y_E`: Odometry linear coordinate in earth frame, in millimeters (1000 = 1m)
  - `TH_E`: Body rotation angle from earth frame, in milli-radians (1000 = 1rad)
  - `SEQ`: 1-byte sequence (0-255) for continuity check.

Vehicle Parameters

Symbol	Value	Unit	Description

Geometrical parameters:

L	0.143	m	Radius of the vehicle chassis.
r	0.029	m	Radius of the bi-directional wheel.

Electromagnetic parameters:

N	390×4	-	Quadrature wheel encoder count per wheel revolution

Controller parameters:

K_{Ω}	0.8		Rotational gain
$K_{\omega_y^B}$	0.1		Gyro scaling gain
K_{ff}	40		Wheel velocity controller feed-forward gain
K_{V_P}	10		Wheel velocity controller feedback Proportional gain
K_{V_I}	1.5		Wheel velocity controller feedback Integral gain

Control equation

Vehicle control with input $[u_{\mathrm{A}} \quad u_B \quad u_C]$:

Base Vector Control (Mode1):

In BVC mode, we'll control the movement of the vehicle, that is the $[x, y]^B$ translation of the body frame and the vehicle rotation ω_Z^B. First, we map the 16-bit, signed short input to velocity and angular rate command:

$$[V_x \quad V_y \quad \Omega]^T = [u_{\mathrm{A}} \quad u_B \quad u_C]^T$$

Than velocity command for each wheel is calculated:

$$\begin{cases} v_A = -\cos(30°)\ V_x - \sin(30°)\, V_y - K_\Omega L\left(\Omega - K_\omega \omega_y^B\right) \\ v_B = \quad \cos(30°)\ V_x - \sin(30°)\, V_y - K_\Omega L\left(\Omega - K_\omega \omega_y^B\right) \\ v_C = \qquad\qquad\qquad\qquad\quad V_y - K_\Omega L\left(\Omega - K_\omega \omega_y^B\right) \end{cases}$$

Where L is the radius of the vehicle chassis.

Individual Wheel Drive (Mode2):

In IWD mode, three packet of the command data is directly sent to wheel as velocity command.

$$[v_A \quad v_B \quad v_C]^T = [u_{\mathrm{A}} \quad u_B \quad u_C]^T$$

Wheel control: 100Hz loop

With velocity command of each wheel $[v_A \quad v_B \quad v_C]$, wheels are controlled by three independent PI controller:

$$u(k) = K_{ff} v_A + u(k-1) + K_{V_P}\left(e_A(k) - e(k-1)\right) + K_{V_I} e(k) \quad ,where \begin{cases} k: this\ sample\ ; \ \ (k-1): last\ sample \\ K_{ff}: feedforward\ term \\ K: PI\ controller\ gains \end{cases}$$

Where error:

$$e_A = v_A - \left(n_A(k) - n_A(k-1)\right), \qquad n: Encoder\ count$$

Each wheel motor driver is than fed with the generated PWM signal $u(k)$

Odometry equation

Encoder estimation is called at the same block of control loop, i.e. 100Hz.

Given encoder count $[n_{\mathrm{A}} \quad n_B \quad n_C]^T$ at $\mathrm{t} = k$

$$\begin{bmatrix} d\theta_{\mathrm{A}} \\ d\theta_B \\ d\theta_{\mathrm{C}} \end{bmatrix} = \frac{2\pi \mathrm{r}}{N} [n_{\mathrm{A}} \quad n_B \quad n_C]^T$$

Where N is the quadrature encoder count per wheel revolution.

The deviation of state at given sampling time is calculated as follow:

$$\begin{bmatrix} dx \\ dy \\ d\theta \end{bmatrix} = \begin{bmatrix} -\frac{1}{2\cos(30°)} & \frac{1}{2\cos(30°)} & 0 \\ -\frac{1}{3 \times 2\sin(30°)} & -\frac{1}{3 \times 2\sin(30°)} & \frac{2}{3} \\ -\frac{1}{3L \times 2\sin(30°)} & -\frac{1}{3L \times 2\sin(30°)} & -\frac{1}{3L} \end{bmatrix} \begin{bmatrix} d\theta_{\mathrm{A}} \\ d\theta_B \\ d\theta_{\mathrm{C}} \end{bmatrix} = \begin{bmatrix} (-d\theta_{\mathrm{A}} + d\theta_B)\cos(30°)/2 \\ \big((-d\theta_{\mathrm{A}} - d\theta_B) + 2d\theta_C\big)/3 \\ \big((-d\theta_{\mathrm{A}} - d\theta_B) - d\theta_C\big)/3L \end{bmatrix}$$

With proper coordinate transformation, we can calculate our velocity w.r.t earth frame, and make positional integration accordingly

$$\begin{bmatrix} V_x \\ V_y \\ \omega \end{bmatrix}^E = \frac{1}{\delta t} \begin{bmatrix} \cos\left(\theta + \frac{d\theta}{2}\right) & -\sin\left(\theta + \frac{d\theta}{2}\right) & 0 \\ \sin\left(\theta + \frac{d\theta}{2}\right) & \cos\left(\theta + \frac{d\theta}{2}\right) & 0 \\ 0 & 0 & 1 \end{bmatrix}_B^E \begin{bmatrix} dx \\ dy \\ d\theta \end{bmatrix}^B, \qquad \delta t = t(\mathrm{k}) - t(\mathrm{k}-1)$$

$$\begin{bmatrix} X \\ Y \\ \theta \end{bmatrix}^E = \int \begin{bmatrix} V_x \\ V_y \\ \omega \end{bmatrix}^E \cong \sum \begin{bmatrix} V_x \\ V_y \\ \omega \end{bmatrix}^E \delta t = \sum \begin{bmatrix} \cos\left(\theta + \frac{d\theta}{2}\right) & -\sin\left(\theta + \frac{d\theta}{2}\right) & 0 \\ \sin\left(\theta + \frac{d\theta}{2}\right) & \cos\left(\theta + \frac{d\theta}{2}\right) & 0 \\ 0 & 0 & 1 \end{bmatrix}_E^B \begin{bmatrix} dx \\ dy \\ d\theta \end{bmatrix}^B$$

Appendix

- Useful serial command

```
                      MODE    CMD_A       CMD_B      CMD_C
```

  - Stop:

    `0xFF 0xFE 0x02 0x00 0x00 0x00 0x00 0x00 0x00`

  - E-stop:

    `0xFF 0xFE 0x10 0x00 0x00 0x00 0x00 0x00 0x00`

  - Direct wheel control

    Rotate all: (32)

    `0xFF 0xFE 0x02 0x00 0x20 0x00 0x20 0x00 0x20`

    Rotate all reversed: (-32)

    `0xFF 0xFE 0x02 0xFF 0xE0 0xFF 0xE0 0xFF 0xE0`

    Rotate A:

    `0xFF 0xFE 0x02 0x00 0x20 0x00 0x00 0x00 0x00`

    Rotate B:

    `0xFF 0xFE 0x02 0x00 0x00 0x00 0x20 0x00 0x00`

    Rotate C:

    `0xFF 0xFE 0x02 0x00 0x00 0x00 0x00 0x00 0x20`

  - Vehicle control

    $V_x = 32$

    `0xFF 0xFE 0x01 0x00 0x20 0x00 0x00 0x00 0x00`

    $V_y = 32$

    `0xFF 0xFE 0x01 0x00 0x00 0x00 0x20 0x00 0x00`

    $\Omega = 8$

    `0xFF 0xFE 0x01 0x00 0x00 0x00 0x00 0x00 0x08`

Deprecated

- 25Hz Encoder data

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
0xFF	0xFB	ENC_DIF_A_H	ENC_DIF_A_L	ENC_DIF_B_H	ENC_DIF_B_L	ENC_DIF_C_H	ENC_DIF_C_L	SEQ

  - Length: 9 byte
  - ENC_DIF: encoder position difference between this tx and last tx
  - signed short (2bytes, 8-bit), [+32,767 -32,768]
  - SEQ: 1-byte sequence (0-255) for continuity check.