MEMS based Human-Machine Interaction System for Virtual Touring

Xun Xu
School of Electrical Engineering and Information
Sichuan University
Chengdu, China
alex.xun.xu@hotmail.com

Yusheng Liu
School of Electrical Engineering and Information
Sichuan University
Chengdu, China
yliu866@yahoo.com.cn

***Abstract*—To obtain a kind of cheap, convenient and user-friendly virtual reality interaction system, the Human-Machine Interaction System based on Micro Electronic Mechanical Sensor (MEMS), LM3S1138 embedded Micro Control Unit (MCU) and Vega modeling software for virtual touring is introduced in this paper. Two MEMS including a magnetic sensor and an accelerate sensor are utilized to calculate the state variable of the joystick. Based on the measurements of two sensors the state variable consists of azimuth and inclination are computed which can uniquely determine the state of joystick in spherical coordinates. The system, firstly, record the initial state of joystick. Then by comparing the variation between initial state and current state, action command changing the viewpoint in the virtual world are generated. Finally, in Vega established virtual world theoretical command boundary and measured command boundary are compared proving the feasibility of this system in practice.**

Keywords-MEMS; Virtual Reality; accelerate sensor; magnetic sensor

I. INTRODUCTION

Virtual Reality (VR) is a kind of advanced human-machine interacting technique simulating human actions like watching, hearing, moving, etc. Currently, research into VR techniques mainly focus on virtual world development and interacting tools design. In terms of virtual world development, Vega and Virtual Reality Modeling Language (VRML) [1] are among the most popular virtual worlds development tools. Besides, common interacting tools include traditional devices like keyboard and mouse and emerging head tracker. However, the traditional and emerging devices are either none user-friendly or too expensive [2, 3]. To overcome these drawbacks, this paper proposed a human-machine interaction system based on Micro Electronic Mechanical Sensor (MEMS), Micro Control Unit (MCU) and Vega modeling software for virtual touring. Freely touring in the virtual system can be realized by manipulating the joystick. This system can be easily transplanted to other application field due to its simple structure, low maintenance and convenience. Only by updating the virtual world user can take a different touring experience.

II. SYSTEM STRUCTURE

The human-machine interaction system proposed in this paper consists of two parts, the joystick module and the host module as illustrated in Fig. 1. On the joystick side, the inclination with respect to the ground and the azimuth angle with respect to the geomagnetic field can be generated by MCU according to the calibrating components on each axis of the magnetic sensor and accelerate sensor. These two angles will be considered as the state variable of the joystick and any further action commands are based on the initial state. Afterwards, action commands are transmitted to the Host, connected with joystick by serial port, to change the viewing angle in the virtual world.

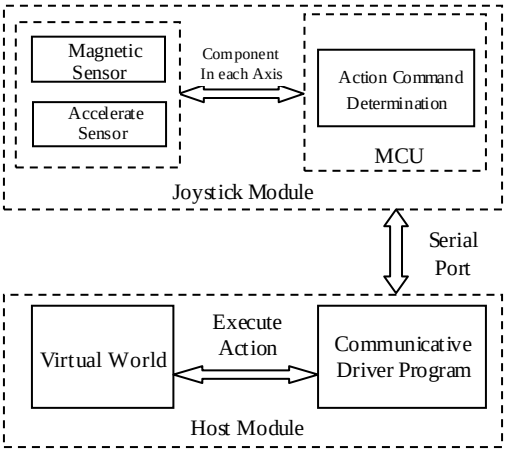


Figure 1. Scheme diagram of Human-Machine Interaction System

III. DESIGN OF JOYSTICK MODULE

The joystick is a key component of the human-machine interaction system. Rotating the joystick horizontally and vertically correspond to the change of viewing point horizontally 0°~360° and vertically -90°~90° respectively. While, pressing forward and backward buttons correspond to the viewing point moving forward and backward. The schematic appearance is illustrated in Fig. 2.

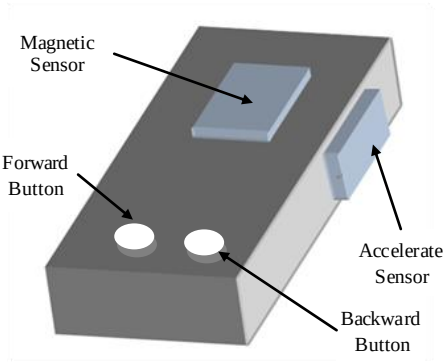
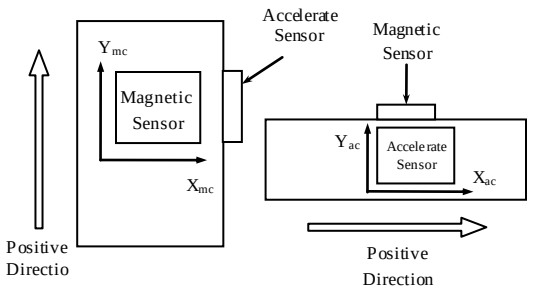


Figure 2. The schematic appearance of joystick

A. Hardware Design

The joystick module of this system consists of MMC212 magnetic sensor, MXC6202 accelerate sensor and LM3S1138 MCU. Each sensor has two calibrating axes, denoted as *X_{mc}-axis*, *Y_{mc}-axis* and *X_{ac}-axis*, *Y_{ac}-axis* respectively, which can calibrate magnetic field intensity within ± 2 gausses and gravity acceleration within ± 2 g. The analog measuring results will be converted to binary digits with the accuracy of 512 counts/gauss and 512 counts/g respectively and then transmitted to host by I^2C.

The geometrical placement of sensors in joystick is illustrated in Fig. 3 where magnetic sensor is parallel to the top of joystick and accelerate sensor is on the side of joystick. In original condition the positive direction of joystick, as denoted in Fig. 3, is parallel to the ground. Moreover, gravity is vertical to the plane spanned by X_c and Y_c axes. Thus *Y_c-axis* is parallel to gravity. In addition, Positive Direction (*PD*) marked in Fig.4 is collinear with Y_{mc} and X_{ac} indicating the positive direction of joystick. Therefore the state of joystick can be represented by *PD* [4, 5].



(a) Topview (b) Sideview

Figure 3. Placement of sensors in joystick

Close to the equator, geomagnetic filed is almost parallel to the surface of earth and gravity is perpendicular to the surface of earth. Thus, a three-dimensional space as shown in Fig. 4 can be uniquely determined where the opposite of gravity is denoted as *Z-axis*, geomagnetic field as *X-axis* and the joystick as the origin. Obviously, plane *XOY* is parallel to the surface of earth. α (0°~360°), the azimuth, is the angle between the projection of *PD* on *XOY* and *X-axis*. β (-90°~90°), the inclination, is the angle between vector *PD* and plane *XOY*. Therefore, the direction of vector *PD* can be uniquely specified by α and β which are taken as the state variable of joystick. The calibrating axes, *X_{mc}-axis* and *Y_{mc}-axis*, of magnetic sensor correspond to vector *MX* and *MY* respectively. Similarly, *X_{ac}-axis* and *Y_{ac}-axis* corresponds to *AX* and *AY* respectively. In the light of placement of sensors, vector *PD*, *AX* and *MY* are collinear, while vector *AX*, *MY* and *Z-axis* are coplanar. Besides, β equals to β' which can be calculated according to *AX* and *AY*. Likewise, vector *MY_{xoy}*. *MX* and *X-axis* are coplanar. α equals to α' which can be calculated according to *MX* and *MY_{xoy}* as well. *MY_{xoy}* is defined as below

$$\left|MY_{xoy}\right| = \left|MY\right| / \cos\beta$$

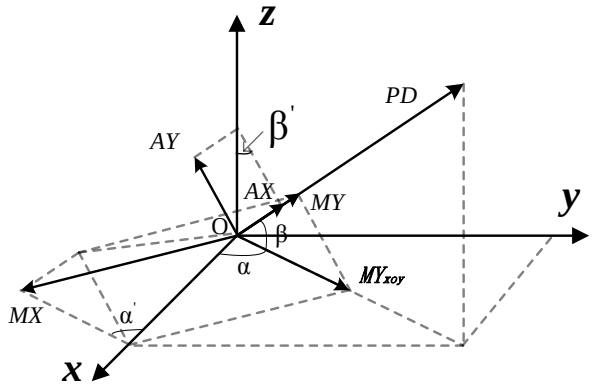


Figure 4. Scheme of sensors' calibrating axes in three-dimensional space

B. Calculation of State Variable

Because α and β can uniquely determine the direction of *PD* which represents the state of joystick, it is naturally to choose the following vector to describe the state of joystick.

$$\boldsymbol{x} = \begin{bmatrix} \alpha & \beta \end{bmatrix}^{\mathrm{T}} \tag{1}$$

The value of the first variable in (1), α, depends on the value of β and β can be calculated depending on which quadrant *PD* is in. The projection of aforementioned three-dimensional space to *YOZ* plane is presented graphically in Fig. 5 where *PD_{yoz}* is in 1st quadrant and 4th quadrant respectively. The calibrating outputs of accelerate sensor are G_x and G_y using which β can be worked out by (2).

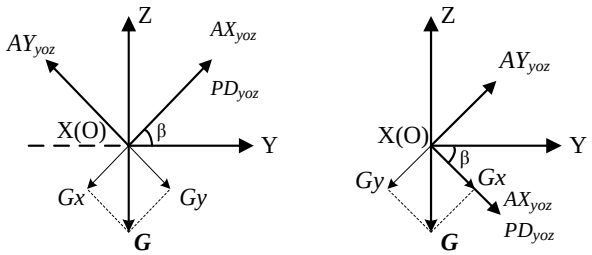


Figure 5. Schematic diagram of inclination calculation

$$\beta=\begin{cases}\arcsin G_x/|G| & G_x>0\,\&\,G_x<G_y\\ \arccos G_y/|G| & G_x>0\,\&\,G_x>G_y\\ \arccos G_x/|G| & G_x<0\,\&\,G_x<G_y\\ \arcsin G_y/|G| & G_x<0\,\&\,G_x>G_y\end{cases} \quad (2)$$

Where $|G|=\sqrt{G_x{}^2+G_y{}^2}$

The calibrating outputs of magnetic sensor are B_x and B_y, corresponding to *MX* and *MY* calibrating axes respectively. Since $BY_{xoy}=B_y/\cos\beta$, the component of geomagnetic field on plane *XOY*, denoted as B_{xoy}, is given by the following equation

$$B_{xoy}=\sqrt{B_x^2+BY_{xoy}^2}$$

Then α can be acquired by the following equations under different conditions.

When $B_x>0$ and $BY_{xoy}>0$, $\alpha\in[0,\pi/2]$ which can be calculated from (3)

$$\alpha=\begin{cases}\arccos(|BY_{xoy}|/B_{xoy}) & |BY_{xoy}|>|B_x|\\ \arcsin(|B_x|/B_{xoy}) & |BY_{xoy}|<|B_x|\end{cases} \quad (3)$$

When $B_x>0$ and $BY_{xoy}<0$, $\alpha\in[\pi/2,\ \pi]$ which can be calculated from (4)

$$\alpha=\begin{cases}\pi-\arcsin(|BY_{xoy}|/B_{xoy}) & |BY_{xoy}|>|B_x|\\ \pi-\arccos(|B_x|/B_{xoy}) & |BY_{xoy}|<|B_x|\end{cases} \quad (4)$$

When $B_x<0$ and $BY_{xoy}<0$, $\alpha\in[\pi,\ 3\pi/2]$ which can be calculated from (5)

$$\alpha=\begin{cases}\pi+\arccos(|BY_{xoy}|/B_{xoy}) & |BY_{xoy}|>|B_x|\\ \pi+\arccos(|B_x|/B_{xoy}) & |BY_{xoy}|<|B_x|\end{cases} \quad (5)$$

When $B_x<0$ and $BY_{xoy}<0$, $\alpha\in[3\pi/2,\ 2\pi]$ which can be calculated from (6)

$$\alpha=\begin{cases}2\pi-\arcsin(|BY_{xoy}|/B_{xoy}) & |BY_{xoy}|>|B_x|\\ 2\pi-\arccos(|B_x|/B_{xoy}) & |BY_{xoy}|<|B_x|\end{cases} \quad (6)$$

C. Determination of Action Command

To minimize the amount of data to be transmitted, only the action command generated by MCU is transmitted to the Host, including Pitch up, Pitch down, Turn left, Turn right, Move forward and Move backward. The first four actions are determined by checking the state of joystick and the latter two actions are determined by whether buttons are pressing on the joystick. The principle of encoding action command is given in Fig. 6

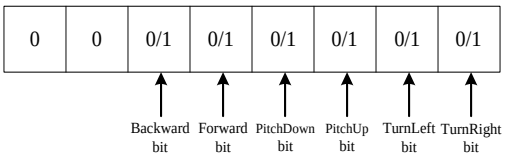


Figure 6. Principle of encoding action command

In Fig. 6, each bit being 1 indicates a specific action is done, otherwise, means not.

After initiation of joystick, the initial state of joystick denoted as $\boldsymbol{x}_0=[\alpha_0\quad\beta_0]^{\mathrm{T}}$ is automatically recorded

Then, the program comes to a loop. In each loop, the current state variable of joystick $\boldsymbol{x}_{\mathrm{i}}=[\alpha_t\quad\beta_t]^{\mathrm{T}}$ is acquired and whether button being pressed is checked. The variation between the current state variable and the initial state variable is given as following

$$\Delta\boldsymbol{x}_{\mathrm{i}}=[\Delta\alpha_t\quad\Delta\beta_t]^{\mathrm{T}}$$

where $\Delta\alpha_t=\alpha_t-\alpha_0$, $\Delta\beta_t=\beta_t-\beta_0$.

Then the action commands are generated according to the principles shown in TABLE I.

TABLE I. PRINCIPLE OF GENERATING ACTION COMMANDS

Content	Action	Code(hex)
Δα>+8 °	Turn left	0x01
Δα<-8 °	Turn right	0x02
Δβ>+7 °	Pitch up	0x04
Δβ<-7 °	Pitch down	0x08
Forward Button	Forward	0x10
Backward Button	Backward	0x20

Multiple actions may happen simultaneously (e.g., Turn left and Pitch up can happen at the same and the final action command code is 00010101b). However, such irregular action coincidence as Turn left and Turn right, Pitch up and Pitch down and Forward and Backward are illegal.

IV. DESIGN OF HOST MODULE

The Host in this system consists of Communicative Driver Program and Virtual World Platform. Depending on the received action command Communicative Driver Program will manage to change the viewpoint in the virtual world. To ensure the above function works properly, two requirements for the Virtual World Platform are given below[6]:

- Editable, the virtual world established by the platform should be able to be reconfigured by user
- Interface, the change of viewpoint in the virtual world should be able to be manipulated by Operating System or Communicative Driver Program

As a result, Vega is selected to implement the construction of virtual world [7]. By simulating the press of certain keys on the keyboard, the viewpoint in the virtual world is changed. To simulate the action of press key on the keyboard, WinIo.dll is used to write bytes to and read bytes from the 0x60 and 0x64 registers belonging to the 8042 keyboard/mouse controller.

A. *Program Flow*

The MCU in joystick and the Host work asynchronously. Joystick transmits action command to Host circularly. Each time a command is transmitted, a timer is started in the MCU which is waiting for a reply. The Host will send a reply indicating a successful receipt to the joystick after receiving the command. If the timer in joystick times out without receiving any reply, the joystick will transmit another action command to the Host. Fig.7 and Fig. 8 illustrate the program flow of joystick and Host respectively.

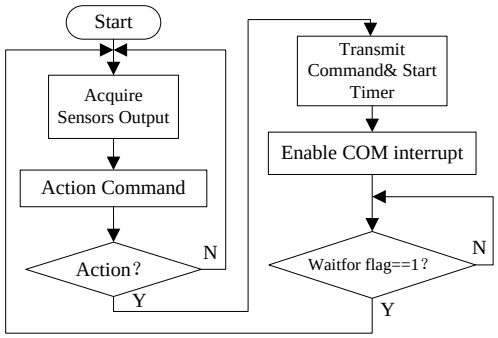
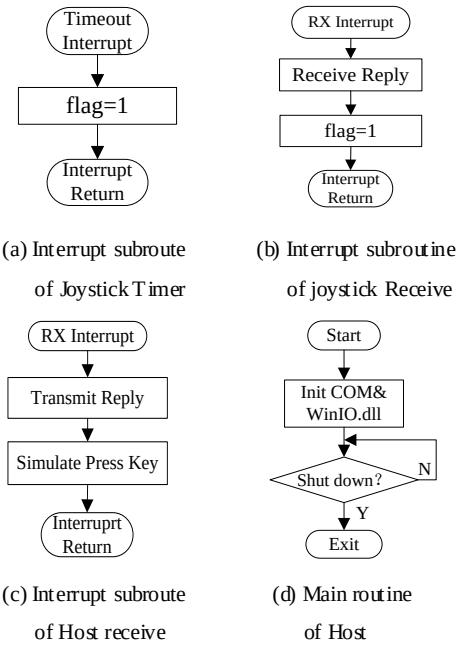


Figure 7. Program flowchart of joystick

Either a Timeout or Receive Interrupt in joystick will assign 1 to the variable flag which equals to 0 initially and cause the program on joystick jupming the next loop.



(a) Interrupt subroute of Joystick Timer

(b) Interrupt subroutine of joystick Receive

(c) Interrupt subroute of Host receive

(d) Main routine of Host

Figure 8. System program flowchart

V. TEST & ANALYSIS

We assume the magnitude of vector PD in Fig. 4 equals to 1

$$|\boldsymbol{PD}|=1$$

First of all, turn the joystick left and right and then pitch up and down independently and then rotate the joystick randomly during the previous operation, record all the critical states where action commands are generated. These recorded critical states will form a sequence, denoted as below

$$\left\{\boldsymbol{x}_i \,|\, i=1,2\cdots N\right\}$$

where $\boldsymbol{x}_i=\left[\alpha_i \quad \beta_i\right]^T$

Therefore, according to the sequence a new sequence described by spherical coordinate is given as following.

$$\left\{\boldsymbol{P}_i \,\middle|\, \boldsymbol{P}_i=\left[|\boldsymbol{PD}| \quad \boldsymbol{x}_i^T\right]^T\right\} \tag{7}$$

The geometrical interpretation of sequence (7) is a series of points in the three-dimensional space illustrated in Fig. 4.

Then the continuous curves, denoted as Theoretical and Measured Boundary in Fig. 9 are the fitting to a series of points which are the projection of sequence (7) on plane *YOZ*. The Boundary represents the critical directions which the vector PD points to.

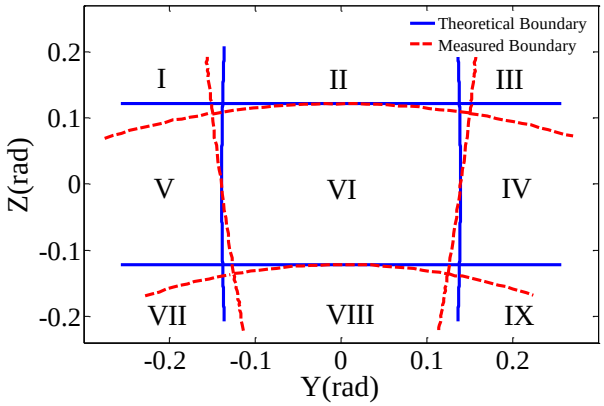


Figure 9. The command boundary projected on plane *YOZ*

In Fig. 9 plane *YOZ* is divided into 9 sectors by Theoretical and Measured Boundary. The relationships between action commands and sectors are given in TABLE II.

TABLE II. RELATIONSHIP BETWEEN SECTORS AND ACTION COMMANDS

Sector	Action Command	Code(Hex)
I	Left & Up	0x05
II	Up	0x04
III	Right & Up	0x06
IV	Left	0x01
V	Unmoved	0x00
VI	Right	0x02

VII	Left & Down	0x09
VIII	Down	0x08
IX	Right & Down	0x0a

The Measured Boundary in Fig. 9 demonstrates that by turning the joystick appropriate degrees definite action commands will be generated which proves the feasibility of the proposed system. However, there is an error between Theoretical Boundary and Measured Boundary which, however, is negligible when $\Delta\alpha$ and $\Delta\beta$ are small. The reason for this error is that when user turns the joystick, especially when turning left and right, roll angle is introduced causing the distortion of upper and lower boundary.

REFERENCES

[1] K. Russ and A. Wetherelt, "Large-scale mine visualization using VRML" , *IEEE Computer Graphics and Applications*, vol. 19, pp. 39-44, Mar.-Apr. 1999

[2] R. Kalawsk, "The Science of Virtual Reality and Virtual Environments". Addison Wesley ,1993

[3] Di Wu, Wenqian Huang, "The Development and Current Research on Virtual Reality Technique", *Hydrographic Surveying and Charting*, Vol. 22 No. 6, pp. 15-17, Nov. 2002. (in Chinese)

[4] J. Bartholomeycz, S. Zimmermann, U. Breng, W. Gutmann, M. Hafen, E. Handrich, et al."MEMS based inertial measurement unit for attitude and heading reference systems", *MOMS,MOES,ICS and Electronic Components (SSI), 2008 2nd European Conference & Exhibition on Integration Issues of Miniaturized Systems*, pp. 1-8, Apr. 2008.

[5] Jian Kang, BoXiong Wang, ZhongXiang Hu, Rui Wang, Tao Liu, "Study of drill measuring system based on MEMS accelerative and magnetoresistive sensor", *The Ninth International Conference on Electronic Measurement & Instruments*, pp. 2-112-2-116, Aug. 2009

[6] Bing Liu, Rufei Liu, Xiushan Lu, Yongqiang Xie, Xiaomei Wang, "Study of the virtual reality smart simulation campus Based on Vega", *International Conference on Mechanic Automatic and Control Engineering (MEAC)*, pp. 6864 – 6867, Jul. 2011.

[7] Sourin Alexei. "Nanyang technological university virtual campus". *IEEE Computer Graphics and Applications*, vol. 24(6), pp.6-8, Nov.-Dec.2004