Using the eSense Wearable Earbud as a Light-Weight Robot Arm Controller

Henry Odoemelem
University of Siegen
henry.odoemelem@student.uni-siegen.de

Alexander Hölzemann
University of Siegen
alexander.hoelzemann@uni-siegen.de

Kristof Van Laerhoven
University of Siegen
kvl@eti.uni-siegen.de

ABSTRACT

Head motion-based interfaces for controlling robot arms in real time have been presented in both medical-oriented research as well as human-robot interaction. We present an especially minimal and low-cost solution that uses the eSense [1] ear-worn prototype as a small head-worn controller, enabling direct control of an inexpensive robot arm in the environment. We report on the hardware and software setup, as well as the experiment design and early results.

CCS CONCEPTS

• **Computer systems organization** → **Robotics**; • **Human-centered computing** → *Ubiquitous and mobile computing*.

ACM Reference Format:
Henry Odoemelem, Alexander Hölzemann, and Kristof Van Laerhoven. 2019. Using the eSense Wearable Earbud as a Light-Weight Robot Arm Controller. In *Proceedings of 1st International Workshop on Earable Computing (EarComp'19).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3345615.3361138

1 INTRODUCTION

The past decade has seen an increasing interest in developing and investigating the interface between humans and robots. For many scenarios in this research, precision and speed are crucial, with cost factors and size and mobility of such systems being less of a focus. In contrast, we present here a human-robot interface that aims at being minimal in terms of size and costs, and intend to investigate the trade-offs that are caused by this minimalism in terms of accuracy and speed. Our application domain is head motion-based robot control, as it is for instance required to enable tetraplegics to control a multi-degree of freedom robot arm in real-time using solely head motion (as for instance motivated in [4]).

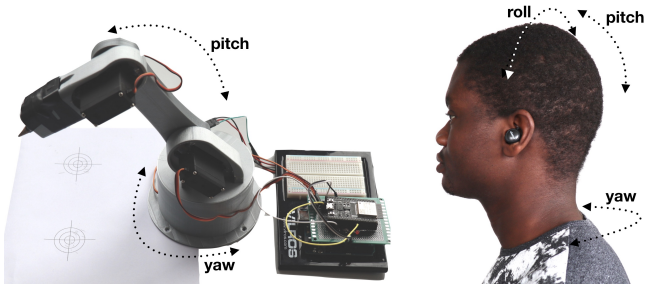

Figure 1: This work presents an affordable and light-weight system that uses an in-ear IMU to control a robot arm's yaw and pitch motions by moving the head.

2 DESIGN OF HARDWARE AND SOFTWARE

For our design, we use an open-source design 3-DOF robot arm, using STL Files available on [1]. This design uses 3 low-cost mg996r servo motors capable of 180 deg rotation and is powered by an Arduino (connected to a 9V dc power adapter). This design results in an extensible, re-programmable, and low-cost robot arm that costs about 70$ for all components. We use this model as this design explicitly represents an entry model for interactive control applications, where users need to adapt to the speed and accuracy of the servo motors.

In order to control the robot arm, IMU (Inertial Measurement Unit) data are sent via Bluetooth Low Energy(BLE) from an eSense ear-worn unit to a ESP32-WROOM-32D-equipped robot for local processing. eSense[2] is a multi-sensory earable platform for personal-scale behavioural analytics research. It is a True Wireless Stereo (TWS) earbud, composed of a Qualcomm CSR8670, with dual mode Bluetooth (Bluetooth Classic and Bluetooth Low Energy), three-axis accelerometer, a three-axis gyroscope,etc. The left earbud is the one containing the IMU sensor accessible through the BLE interface. The ESP32-WROOM-32D module[3] at the robot arm. This module is a generic Wi-Fi+BT+BLE MCU system-on-chip,

EarComp'19, September 9, 2019, London, United Kingdom

ACM ISBN 978-1-4503-6902-2/19/09…$15.00
https://doi.org/10.1145/3345615.3361138

[1] https://howtomechatronics.com/download/arduino-robot-arm-stl-files/
[2] eSense User Documentation: http://www.esense.io/share/eSense-User-Documentation.pdf
[3] ESP32 Datasheet: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32d_esp32-wroom-32u_datasheet_en.pdf

equipped with an Xtensa single-/dual-core 32-bit LX6 microprocessor. With up to 600 MIPS, it is designed for flexible mobile, wearable, and networked sensor applications.

Connection: The ESP32 is programmed using the Arduino IDE, and the BLE connection between eSense and ESP32 was accomplished with the help of the Arduino BLEDevice library [4]. To access the IMU data using the appropriate service and characteristics UUID[5], we first have to register for notification and enable IMU sampling, here we are using 50Hz sampling rate. Next, on each notification we convert the accelerometer readings to g using the 8192 LSB/g Scale factor (+/- 4 g range is used) and gyroscope reading to deg/s using 65.5 LSB/(deg/s) Scale factor(+/-500 deg/s range is used).

Frame transformation: The acceleration in the earth's reference frame has to be transformed to the IMU body frame, the orientation of the eSense IMU is shown in the figure 3. Appendix A contains the details on how we achieve this.

When the eSense is worn, the orientation of the earbud is as shown in the figure 4, where it is 90 deg (approximately) rotated about the z-axis from the orientation we have used in our calculations, to account for this and be able to make use of (14), we must first make a transformation of our IMU readings from the worn orientation to the original orientation used in calculation as shown in (1), where variables with subscript w represent values from the worn orientation.

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \cos(-90) & -\sin(-90) & 0 \\ \sin(-90) & \cos(-90) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_w \\ y_w \\ z_w \end{pmatrix} = \begin{pmatrix} y_w \\ -x_w \\ z_w \end{pmatrix} \quad (1)$$

The simple substitution necessary is as shown in (2) and (3).

$$G_x = G_{yw}, G_y = -G_{xw}, G_z = G_{zw} \quad (2)$$

$$gyro_x = gyro_{yw}, gyro_y = -gyro_{xw}, gyro_z = gyro_{zw} \quad (3)$$

Filtering: Accelerometer is susceptible to high-frequency noise and needs a low pass filter, while the gyroscope drifts with time due to integration and needs a high pass filter. The accelerometer readings from (14) (has to be converted to degrees) and the gyroscope rate readings in deg/sec are passed through a complementary filter to mitigate noise and drift (see figure 2). (15) shows the complementary filter for pitch and (16) for roll (see Appendix B).

Calibration: Since different users will have different neutral/natural head orientation, it makes sense to calibrate the readings for each user. After the system setup, with the user's head upright and facing forward the micro-controller receives on notification, N IMU readings and takes the sample mean of these values which we call offset readings (see Appendix C) as shown in (18). With (19) and (20) we get the

[4]BLEDevice Library: https://github.com/nkolban/ESP32_BLE_Arduino/blob/master/src/BLEDevice.h

[5]eSense-BLE-Specification: http://www.esense.io/share/eSense-BLE-Specification.pdf

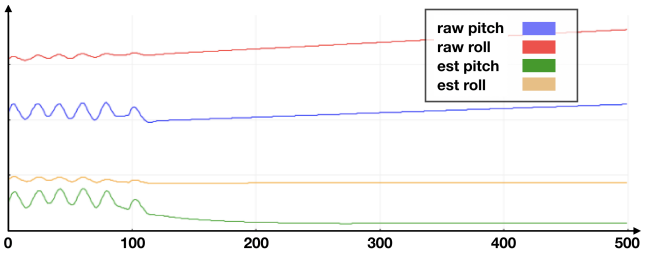

Figure 2: Raw pitch and roll angles calculated from the raw eSense IMU readings, showing drift compared to estimated pitch and roll angles after the complementary filter, with noise and drift eliminated. Head pitch movement was present till sample 100, afterwards the head was kept still.

estimated pitch and roll angles, for which the neutral/natural head orientation gives values approximately equal to zero.

Mapping: From (8) we observe that the yaw angle is eliminated during calculation, [3] because accelerometer cannot detect yaw rotations, since the G vector does not change during yaw. Hence, we have mapped the user's head estimated roll angle to the robot arm yaw motion (base servo motor, for left and right motion) and the user's head estimated pitch angle to robot arm pitch motion (for up and down motion). However, when the user's head makes a pitch motion, there is a bit of roll motion and vice versa (see Figure 2), hence, constraint/condition statements are used to determine exactly which motion(pitch or roll) is dominate/intended by the user before mapping.

3 EXPERIMENT DESIGN AND OUTLOOK

Our experiments are currently ongoing, where novice users are wearing the eSense to control the robot to mark (with a pencil) targets (as can be seen in Figure 1) as accurately and quickly as possible. Both performance measures are then taken over multiple sessions to also assess the learning of the interface. Intermediate results show that users tend to reach the targets from 10 centimetres within 5 seconds at an accuracy of approximately 9 mm from the target centre. What remains to be investigated is (1) an analysis of the importance of accuracy versus speed, and (2) the range of applications that such performances would allow.

REFERENCES

[1] Fahim Kawsar, Chulhong Min, Akhil Mathur, and Allesandro Montanari. 2018. Earables for Personal-Scale Behavior Analytics. *IEEE Pervasive Computing* 17, 3 (jul 2018), 83–89.

[2] Hyung Gi Min and Eun Tae Jeung. 2015. Complementary filter design for angle estimation using mems accelerometer and gyroscope. *Department of Control and Instrumentation, Changwon National University, Changwon, Korea* (2015), 641–773.

[3] Mark Pedley. 2013. Tilt sensing using a three-axis accelerometer. *Freescale semiconductor application note* 1 (2013), 2012–2013.

[4] Nina Rudigkeit and Marion Gebhard. 2019. AMiCUS - A Head Motion-Based Interface for Control of an Assistive Robot. *Sensors* 19, 12 (2019), 2836. https://doi.org/10.3390/s19122836

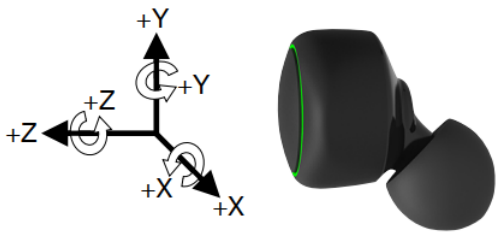

Figure 3: (Image from eSense User Documentation) The orientation of the IMU in eSense is shown above, the lower case of the axes are used in our calculations;x=+X,y =+Y,z=+Z. CCW direction on y axis represent Yaw direction, CCW on z axis represent Pitch direction and CCW direction on x axis represent Roll direction.

A EARTH REFERENCE FRAME TO BODY (ESENSE IMU) FRAME TRANSFORMATION

.

Similarly to [3], we use a transformation matrix R as shown in (4). We assume the accelerometer undergoes no linear acceleration and only experiences a gravitational field g, also the IMU y-axis is aligned to direction of g hence the unit vector (0,1,0) as shown in (5). The three-axis accelerometer reading is given by the vector G. We make use of the Euler angles; roll (about x-axis), pitch (about z-axis), and yaw (about y-axis) order for the transformation matrix;

$$R = R_x(-\phi).R_z(-\theta).R_y(-\varphi)$$

In (6), the angles are negative because we make a transformation from earth's reference frame to the IMU body frame.

$$\mathrm{G} = \begin{pmatrix} G_x \\ G_y \\ G_z \end{pmatrix} = R(g-a) \tag{4}$$

$$\mathrm{G} = \begin{pmatrix} G_x \\ G_y \\ G_z \end{pmatrix} = Rg = R\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \tag{5}$$

$$\mathrm{R}\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = R_x(-\phi).R_z(-\theta).R_y(-\varphi).\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \tag{6}$$

(7) shows the elementary rotations from earth's (inertial) reference frame to the body (IMU) frame.

$$\begin{aligned} R_x(-\phi) &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(-\phi) & -\sin(-\phi) \\ 0 & \sin(-\phi) & \cos(-\phi) \end{pmatrix} \\ R_z(-\theta) &= \begin{pmatrix} \cos(-\theta) & -\sin(-\theta) & 0 \\ \sin(-\theta) & \cos(-\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ R_y(-\varphi) &= \begin{pmatrix} \cos(-\varphi) & 0 & \sin(-\varphi) \\ 0 & 1 & 0 \\ -\sin(-\varphi) & 0 & \cos(-\varphi) \end{pmatrix} \end{aligned} \tag{7}$$

After applying (7) in equation (6), we arrive at (8) and (9) below,

$$\mathrm{R}\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -\sin(-\theta) \\ \cos(-\phi)\cos(-\theta) \\ \sin(-\phi)\cos(-\theta) \end{pmatrix} \tag{8}$$

which relates the transformed acceleration vector to accelerometer acceleration measurement in normalized form.

$$\hat{G} = Accelerometer\ measurements\ in\ normalized form$$

$$\hat{G} = \frac{G}{\sqrt{G_x^2+G_y^2+G_z^2}} = \begin{pmatrix} \hat{G_x} \\ \hat{G_y} \\ \hat{G_z} \end{pmatrix} = \begin{pmatrix} -\sin(-\theta) \\ \cos(-\phi)\cos(-\theta) \\ \sin(-\phi)\cos(-\theta) \end{pmatrix} \tag{9}$$

To get the pitch angle, we make use of the method in (10) that allows us to use arctan instead of arcsin , to avoid multiple angles for same value of acceleration measurement.

$$\begin{aligned} \frac{\hat{G_x}}{\sqrt{\hat{G_y}^2+\hat{G_z}^2}} &= \frac{-\sin(-\theta)}{\sqrt{\cos^2(-\theta)\left(\left(\cos^2(-\phi)+\sin^2(-\phi)\right)\right.}} \\ &= -\tan(-\theta) \end{aligned} \tag{10}$$

The pitch angle can then be calculated using (11).

$$\theta = -\tan^{-1}\left(\frac{-\hat{G_x}}{\sqrt{\hat{G_y}^2+\hat{G_z}^2}}\right) \tag{11}$$

Similarly, for roll, we get the relationship in (12) and can calculate roll angle using (13).

$$\begin{aligned} \frac{\hat{G_z}}{\hat{G_y}} &= \frac{\sin(-\phi)\cos(-\theta)}{\cos(-\phi)\cos(-\theta)} \\ &= \tan(-\phi) \end{aligned} \tag{12}$$

$$\phi = -\tan^{-1}\left(\frac{\hat{G_z}}{\hat{G_y}}\right) \tag{13}$$

To avoid issues that result with arctan, such as division by zero and inability to distinguish quadrants, we make use of atan2 in actual programming as shown in (14). Also, since the norm of G cancels out in (11) and (13), we can use the values of G non-normalized. The angles here are in radians.

$$\theta = -\mathrm{atan2}\left(-G_x, \sqrt{G_y^2+G_z^2}\right), \phi = -atan2\left(G_z, G_y\right) \tag{14}$$

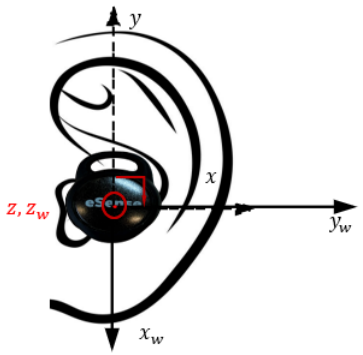

Figure 4: (Image from eSense User Documentation) eSense orientation when worn is approximately 90 degrees CW rotated about the z-axis from original orientation (see figure 3) used in calculation. the new axis orientation are in lower case *w*.

B COMPLIMENTARY FILTER

.

$$\theta_f(k) = \left(\theta_f(k-1) + gyro_z * \Delta t\right) * \alpha + \theta * (1-\alpha) \quad (15)$$

$$\phi_f(k) = \left(\phi_f(k-1) + gyro_x * \Delta t\right) * \alpha + \phi * (1-\alpha) \quad (16)$$

$$\alpha = \frac{T_s}{T_s + \Delta t}, T_s = \frac{1}{2\pi f_c} \quad (17)$$

$\theta_f(k)$, $\phi_f(k)$ *are current filtered pitch and roll angles*

$\theta_f(k-1)$, $\phi_f(k-1)$, *filtered pitch and roll angles* 1 Δt *in the past*

α , $(1-\alpha)$ *are the filter coefficients(weights) and sum up to* 1

$$T_s = \ filter \ \ time \ \ constant$$

$\Delta t = \ sampling \ \ time$, *determined by the IMU sampling rate*

$$f_c = \ filter \ \ cutoff \ \ frequency$$

The filter cutoff frequency is usually chosen such that the weight/coefficient of the gyroscope reading are favoured (greater) more than the weight of the accelerometer readings (since gyroscope readings are more accurate than accelerometer readings in short time intervals e.g. sampling time), and then tuned for best performance. The methods in [2] can be used to determine the filter coefficients.

C CALIBRATION EQUATIONS

.

$$Pitch\_offset = \frac{\sum_{k=1}^{N} \theta_f(k)}{N}$$
$$Roll\_offset = \frac{\sum_{k=1}^{N} \phi_f(k)}{N} \quad (18)$$

$$\text{Estimated\_Pitch\_Angle} = \theta_f(k) - Pitch\_offset \quad (19)$$

$$\text{Estimated\_Roll\_Angle} = \phi_f(k) - Roll\_offset \quad (20)$$